Evolving Neural Networks Using Behavioural Genetic Principles

Maitrei Kohli

A dissertation submitted in partial fulfilment of the requirements for the degree of

Doctor of Philosophy

Department of Computer Science & Information Systems

Birkbeck, University of London

United Kingdom

March 2017

Declaration

This thesis is the result of my own work, except where explicitly acknowledged in the text.

Maitrei Kohli …….

ABSTRACT

Neuroevolution is a nature-inspired approach for creating artificial intelligence. Its main objective is to evolve artificial neural networks (ANNs) that are capable of exhibiting intelligent behaviours. It is a widely researched field with numerous successful methods and applications. However, despite its success, there are still open research questions and notable limitations. These include the challenge of scaling neuroevolution to evolve cognitive behaviours, evolving ANNs capable of adapting online and learning from previously acquired knowledge, as well as understanding and synthesising the evolutionary pressures that lead to high-level intelligence.

This thesis presents a new perspective on the evolution of ANNs that exhibit intelligent behaviours. The novel neuroevolutionary approach presented in this thesis is based on the principles of *behavioural genetics (BG)*. It evolves ANNs' 'general ability to learn', combining evolution and ontogenetic adaptation within a single framework. The 'general ability to learn' was modelled by the interaction of *artificial genes*, encoding the intrinsic properties of the ANNs, and the *environment*, captured by a combination of filtered training datasets and stochastic initialisation weights of the ANNs. Genes shape and constrain learning whereas the environment provides the learning bias; together, they provide the ability of the ANN to acquire a particular task. *Ontogenetic adaptation* was implemented via a local gradient-based search method, while *phylogenetic evolution* was implemented via a Darwinian, fitness-based selection approach.

The project was structured as follows. Chapter 2 presents the novel neuro-evolutionary approach for evolving populations of neural networks, inspired from BG principles. In chapter 3, the framework is applied to an exemplar problem domain drawn from psychology, that of English past-tense acquisition within the field of child language development. This domain is notable because it is a quasi-regular or dual-natured task. Chapter 3 also introduces the analytical technique of assessing the 'heritability' of performance in a population of ANNs. Populations are created comprising identical and non-identical 'twins', so specified by the similarity of their artificial genomes. Heritability provides a scalable summary statistic of the net effect of all internal parameters on learning. However, it can only be quantified when variable also exists in the quality of the environment. The findings from the experimental evaluations demonstrated the effectiveness of the model and provided a basis to extend it to capture population-level differences within developmental settings.

In the second half of the project, the framework was extended to model *transfer learning*, with a special focus on *heterogeneous tasks*. This simulated the neuroevolutionary scenario wherein population members can be required to learn tasks different from those for which they were selected. Chapter 4 lays out the theoretical issues in this field. Large-scale simulations, involving over 200,000 networks, then identified and tested two key factors that modulated the performance of the transfer model – the type of *selection operator* (chapter 5) and the *nature of source task* (chapter 6). By transferring the 'general ability to learn', the transfer model enabled a population of ANNs to acquire successfully five different heterogeneous tasks. Analyses of heritability and environmentability were utilised to reveal which factors were most responsible for variation in performance, and its improvement across generations. Crucially, these large-scale experiments demonstrated that it is possible for a population of ANNs to acquire multiple heterogeneous tasks, and that the heritability metric can be utilised to identify when negative transfer effects may occur. As discussed in chapter 7, the BG-inspired method therefore presents concrete progress in optimising those neurocomputational properties of ANNs relevant to enhance learning across multiple problem domains.

Acknowledgements

I am truly grateful to all the people who supported me through this incredible journey.

My sincerest thanks to

My parents, you are my biggest strength and inspiration. I am so grateful for your unconditional love, prayers, and unwavering belief in me. Most of all, I am extremely grateful for everything you do for me. I dedicate this work to you.

My brother, Jivak, for your constant support, encouragement, funny banter and for always being there for me. Thanks for being the best brother ever.

My grandma, for loving me so much and for endless blessings.

My best friend, Areej, through good times and bad you have been there for me always. Thank you for everything.

Mrs. La Young Jackson, for guiding and helping me through visa processes numerous times. Ms. Kristina Freris, for advising and encouraging me when things got tough and overwhelming. I am really thankful.

Oriental bank of commerce, for giving me education loan and thereby making finances easier for me.

Dept. of Computer Science, Birkbeck, for awarding me with 3-year BEI studentship. Systems Group, for helping me numerous times with Matlab and Condor.

Members of Developmental Neuro-cognition Lab, for giving me several opportunities to present my work and for all your constructive feedback and suggestions. Thanks to Prof. Michael Thomas for making me a member of DNL.

My co-supervisor, Prof. Michael Thomas, for your guidance, encouragement, valuable feedback, hard-work and patience with me. You have taught me to always strive for perfection and to keep the goals and standards high. Working with you has been a great privilege and I will always be grateful to you.

Finally, my supervisor and mentor, Prof. George Magoulas. Words cannot describe how grateful I am to you for all the invaluable advise, support, guidance, encouragement, feedback, hard-work and patience with me. I have learned so much from you and it has been a great honour to work with you. Thank you for not only being my supervisor but also the perfect role-model. I will always be earnestly grateful to you.

Table of Contents

List of Tables

List of Figures

Chapter 1 Introduction

An important aspect of human cognition is the capability to learn multiple skills/behaviours and to store and reuse the acquired knowledge to modify behaviour when necessary (Greve et al., 2016). Research in the field of cognition suggests that evolution and intelligence are interconnected (Fogel, 2006), and this raised an interesting possibility for researchers in AI, machine learning and cognitive computing – could we create entities capable of generating intelligent behaviours by modelling evolutionary processes? The result was a research field called neuroevolution (NE). Neuroevolution is a nature inspired approach for creating artificial intelligence. The main objective is to evolve artificial neural networks (ANNs) capable of exhibiting intelligent behaviours. Neuroevolution, therefore, acts as a means to investigate the evolution of intelligence in humans and also as a useful method for engineering ANNs to perform chosen tasks. Similar to evolution via selection in nature, which is driven by feedback from reproductive success, neuroevolution is guided by some measure of overall performance. It therefore makes it possible to find a neural network that optimises behaviour given only sparse feedback, without exact information about what exactly needs to be done. Further, neuroevolution generalises to a wide variety of network architectures and neural models (Floreano et al., 2008; Lehman and Miikkulainen, 2013).

Although the field of neuroevolution has been a widely researched discipline, the last decade has witnessed a resurgence in interest. It has been driven by a number of breakthroughs that occurred during the last decade or so namely, the availability of massive datasets (i.e. Big data) for instance via Google, Facebook, Amazon and many more; powerful and cheaper computational facilities viz. GPUs (Nickolls and Dally, 2010), OpenCL/CUDA (Bourd, 2016; Howes and Munshi, 2015; Nvidia, 2008); and more affordable data storage. Additionally a big breakthrough came about in 2006 called deep learning or deep neural networks. It combines advances in computing power and a special type of neural networks endowed with multiple hidden layers in order to learn extremely complicated patterns in large amounts of data (Schmidhuber, 2015). These resulted in significant advancements in the field of neuroevolution as well, both in terms of methodology and practical, real-world cases. For instance, combining neuroevolution with deep learning architectures is a new trend. Examples include evolving Compositional Pattern Producing Networks (CPPNs) to design convolutional nets (Fernando et al., 2016), evolving Neural Turing Machines to express memory (Greve et al., 2016), and

extreme mini-batching to scale up to large datasets and networks. Additionally, there has been substantive success in using neuroevolution for real-world problems like protein folding (Nielsen et al., 2016) and power plant control (Khadka et al., 2016). Another emerging trend involves the use of neuroevolutionary approach to optimise, configure and correct algorithms (Blum et al., 2016; Martins et al., 2016). The intent is that humans can take care of approximate, high-level algorithm design, while the details are better left for automated optimisation.

Despite the huge success and growing popularity of the research field, there are still some open research questions such as understanding and synthesising the evolutionary pressures leading to high-level intelligence; scaling neuroevolution to evolve cognitive behaviours such as multimodal behaviour, communication, and lifetime learning (Lehman and Miikkulainen, 2013; Lehman and Miikkulainen, 2014); evolving neural networks that learn different skills/behaviours and are capable of storing and reusing the acquired knowledge to modify behaviour online i.e. networks that can learn and adapt ontogenetically as well (de Castro, 2007; Greve et al., 2016; Risi et al., 2010a; Yao et al., 2006).

Considerable research efforts have been made to enable evolving ANNs to adapt online and learn from previously acquired knowledge (Blynel et al., 2003; Greve et al., 2016; Risi et al., 2010b). Many approaches for adaptive ANNs involve using local learning rules for evolving connection weights based on neural activation (Greve et al., 2016; Risi et al., 2015; Stanley et al., 2003). Other researchers have suggested evolving local synaptic plasticity parameters that determine how the weights of ANNs should change during lifetime depending on incoming activation (Greve et al., 2016; Tonelli et al., 2013). However neither of these methods so far have been scaled up to solve more realistic and difficult tasks.

Additionally, neuroevolutionary approaches have not been applied extensively for incremental learning, i.e. learning new skills without forgetting current skills. A similar issue concerns using evolutionary computation for avoiding catastrophic forgetting. This problem occurs when, in order to learn new skills, the learning algorithm changes the weights of neural connections. This results in loss of old skills, when there is inconsistency between old and new skills since the weights that encoded old skills/knowledge have now been changed (Ellefsen et al., 2015; Haykin, 2009). Some solutions have been proposed for this issue; for instance, researchers (Seipone et al., 2005) have used evolution to optimise certain ANN parameters like patterns of connectivity, initial weights, and output error tolerances amongst others to mitigate the effect of catastrophic forgetting. Neural modularity has also been proposed as a method to

tackle this issue (Ellefsen et al., 2015). Use of this approach resulted in networks exhibiting higher performance, learning and retention. Despite these advantages, some issues remain, such as the need to investigate the generality of the neural modularity approach. Further investigations are also needed to examine the effect of more complex learning tasks, experimental parameters such as number of tasks that can be handled, as well as different neural sizes and architectures (Clune et al., 2013; Ellefsen et al., 2015).

This PhD thesis proposes a new neuroevolutionary approach based on behavioural genetic (BG**)** principles. The approach combines evolution with ontogenetic learning (or adaptation) within a single framework. It aims to evolve the general 'ability to learn' or learning predisposition of ANNs and ergo it enables ANN population(s) to acquire any number of learning tasks which are different from evolutionary tasks. The rest of the chapter is organised as follows: the next section, presents the research questions of this PhD, Section 1.2 describes the methodology and finally Section 1.3 explains the structure and the contributions of this thesis.

1.1 Research Questions

The main aim of this PhD thesis is to develop a neuroevolutionary approach which allows: a) evolving the 'ability to learn' and b) combining evolution and adaptation of a given ANN population within a single framework. In order to accomplish this aim, the following research questions have to be addressed.

1. What constitutes the 'ability to learn' and how can the 'ability to learn' be represented such that it is evolvable?
2. How to develop a mechanism for evolving a population of ANNs based on their general 'ability to learn'?
3. How to maintain evolvability and ontogenetic adaptability in the same population in a neuroevolutionary scenario?
4. How to make the neuroevolutionary framework domain relevant or extrapolatable/reusable for any task in any domain?
5. How to avoid catastrophic interference/forgetting whilst maintaining ontogenetic adaptability in a population?

1.2 Methodology

To address the aforementioned research questions, the following methodology is adopted in this thesis. The neuroevolutionary framework/approach draws inspiration from BG principles. Behavioural Genetics is a field of study that examines the role of genetics in individual differences in human behaviour. Behaviour is the most complex phenotype as it reflects the functioning of the complete organism; it is dynamic and changes in response to the environment (Plomin, 1990). This field is concerned with the study of individual differences, i.e. knowing what factors make individuals within a group differ from one another. It also estimates the relative importance of genetic and environmental factors in causing individual differences. Thus, the behaviour or phenotype is the result of genetic factors together with environmental factors.

In the terminology of BG, environmental influences are defined as being of two types, shared (or between-family) and non-shared (or unique and within-family). Shared, or between-family, environmental influences are those which are shared amongst family members and serve to make members of a family similar to each other and different from members of other families. Shared environmental influences often tend to include family structure, socioeconomic status, and parental education to name a few (Plomin and DeFries, 1980). By contrast, non-shared, or within-family, environmental influences are factors that are not common amongst family members, serving to make individuals different from one another. These environmental influences often do not operate on a family-by-family basis but rather on an individual-by-individual basis. Examples include peer groups, perinatal traumas, and parental treatment (Plomin and DeFries, 1980; Plomin et al., 2008). In BG, twin studies are widely employed to untangle genetic and environment effects on behaviour. Heritability is an important concept in BG. It is a statistic that describes the effect size of genetic influence and refers to the proportion of observed or phenotypic variance in a group or population that can be explained by genetic variance; or in simpler terms, it is the amount of population variability explained by genetic similarity (Plomin et al., 2008). There has been increasing acceptance that in humans, many high-level behaviours show marked heritability (Plomin et al., 2008). One of the important recent findings from quantitative behavioural genetic research is that the same set of genes is largely responsible for genetic influence across various cognitive domains. These genes are known as the "generalist genes" to highlight their pervasive influence (Kovas and Plomin, 2007).

The following BG principles, are used/simulated/emulated specifically in the methodology:

– Research in this multidisciplinary field shows that variance in performance is produced by variance in both genes and environment.

– The same set of genes, known as generalist genes, are largely responsible for predicting variance across different cognitive domains.

– Heritability is a useful summary statistic depicting the collective contribution of all genetic variation.

Based on these three principles, this thesis proposes and presents a neuroevolutionary approach that evolves a 'learning predisposition' or the general 'ability to learn', both phylogenetically and ontogenetically, wherein the former refers to evolutionary development whereas the latter implies development via learning (i.e. adaptation) during an individual's lifetime. This work draws an analogy between the influence of genes on neurocomputation and the intrinsic parameters of ANNs, and between the training dataset and unique weights for ANNs and the environment – shared and non-shared respectively. In this approach, the concepts of BG are combined with the idea of a parametrically diverse populations of learning systems, used in the context of a hybrid genetic algorithm, where genes (representing intrinsic factors) and environment (expressed via training datasets and unique weights) interact throughout learning to shape differences in individual classifier behaviours (performance). Within BG, it is well known that the quality of environment can modulate the influence of genetic variation, so called gene-environment interactions (Plomin et al., 2008). Following the analogy, one can similarly observe that training datasets affect the influence of intrinsic parameters.

The approach focuses on evolving two main aspects of ANNs, the architecture (e.g. number of hidden units), including node transfer function (steepness/slope of logistic activation), and the learning rule's algorithmic parameter (initial learning rate). In this approach, these are considered formational parameters of the ANNs which either increase or decrease their ability to acquire a new task and have no specific relation to the problem domain that ANNs need to acquire, in line with the 'generalist genes' hypothesis. In order to constrain learning, these properties are encoded into a genome using standard binary representation. This allows the individuals in a population to have a different genotype, that is, different values of each of the free parameters but from within the same fixed range. It thus leads to variability in a population by giving each network a different ability/capacity to learn new tasks. The Darwinian-based

approach is employed to evolve the genetic/neurocomputational parameters encoded in the artificial genome. Further, the process of generating next generation of ANNs utilises constraints of meiosis and fertilisation. This is more biological than usual in genetic algorithms, but required to maintain genetic relatedness between ANNs needed to measure heritability (Plomin et al., 2008).

Shared environmental variability is implemented as a filter applied to the training tasks, inspired by research on how socio-economic-status (SES) affects cognitive development. A body of research suggests that individuals in lower SES families experience substantially less quality and quantity of information (Thomas et al., 2013). The filter creates a unique subsample of the training set for each simulated individual, based on a parameter determining the quality of the environment. This gives a probability that any given pattern in the full training set would be included in that individual's training set (Thomas et al., 2013). This filter is applied at each generation to create unique training subsets for all members of the population in that generation. The learning speed and fast convergence of many feed-forward neural networks depend to some extent on their initial values of weights and biases. For this reason, in this approach, the initial values of weights are used as a way to capture unique environmental effects. It is worth mentioning that in the proposed approach, the network's weights are not encoded in genome to be evolved. Instead these are continuously modified during the lifetime via learning process in which genetically inherited information interacts with information coming from external environment.

One of the challenges in ensuring the maintenance of evolvability and ontogenetic adaptability in the same population is avoiding catastrophic interference/forgetting or negative transfer. Various attempts have been made to mitigate the effects of catastrophic forgetting/interference such as, using novelty vectors to modify backpropagation algorithm. However, this technique is only applicable for auto encoders thereby limiting its essence as a general solution to catastrophic forgetting (French, 1999). Orthogonalisation based methods mitigate the interference effect between tasks by reducing their representational overlap in input neurons, albeit through manually designed preprocessing (Lewandowsky and Li, 1995). Interleaved learning is another approach which involves training on both old and new data. However this technique is not scalable and does not work for real world environments (Robins, 1995). Other techniques for countering the effects of catastrophic forgetting include neuromodularity (Clune et al., 2013; Ellefsen at al., 2015), multi-objective learning (Jin and Sendhoff, 2006), and conservation training (Albesano et al., 2006) to name a few. However, these methods also have

their limitations as pointed out by respective authors. Therefore it is evident that there is a need for a more scalable technique for avoiding catastrophic forgetting/interference that is applicable for any set of tasks/domains. To this end, in this work the notion of heritability is exploited for assessing task relatedness and thus avoiding any catastrophic interference or negative transfer. The approach uses a population of twins (ANNs with some degree of similarity in their neuro-computational parameters) to disentangle these genetic and environmental influences on performance. Additionally, twin studies provide a valuable tool for exploring environmental influences, especially family or shared environment, against a background of heritability.

Finally, the approach uses a combination of fitness based selection(s) and sexual reproduction to model the interaction between learning and evolution within a single framework.

1.3 Thesis Structure and Contribution

This thesis is organised as follows:

Chapter 2 presents a novel neuro-evolutionary approach for evolving populations of neural networks inspired from BG principles. The chapter discusses the numerous research efforts made in the field of neuro-evolution, their scope and limitations. The literature review highlights the need for a more generic and systematic neuro-evolutionary framework which is not bound by task specifics and is applicable and adaptable to various tasks belonging to any domain. Based on the observations collected from previous research efforts, a framework is proposed which, first enables a population of artificial neural networks to get fitter at a given evolutionary task over generations at a population level (i.e. the evolutionary task is same as the learning task). Second the evolving populations are able to adapt to changes in the environment. Third the members of the population have to learn task(s) which are different from those that they have been selected for, at an individual level. Since the approach draws inspiration from BG principles, the relevant concepts of BG are discussed such as twin studies, genes-environment, heritability, generalist genes, evolution and selection. These concepts are then combined with the idea of a parametrically diverse populations of learning systems, used in the context of a hybrid genetic algorithm, where genes (representing intrinsic factors) and environment (expressed via training datasets and unique weights) interact throughout development to shape differences in individual classifier behaviours (performance). This

approach for combining learning and evolution is systematic and is not dependent on problem domain. It can be easily applied to any given set of learning and/or evolutionary tasks.

In **Chapter 3**, the neuro-evolutionary framework is applied to model the sample domain of children's past tense formation and thereby capture population variability across language development. The work summarised in this chapter models the neuro-evolutionary scenario wherein the evolutionary task is same as the learning task. The chapter includes a comprehensive literature review of the language acquisition field focusing on English past tense verbs. Literature in the field of BG views variability in children's learning in terms of genetic and environmental influences. Although many connectionist models exist for capturing language development, very few consider individual differences. This chapter discusses why acquisition of English past tense is an interesting candidate task to test the framework. It is mainly because this task belongs to quasi-regular domain. Quasi-regular domains are interesting because of the presence of systematic input-output mappings along with the presence of a minority of exceptions. One of the main aims of this chapter is to discover how evolution and learning interact in a dual natured problem domain and whether this interaction lead to potentially divergent overt behaviours? To address this question, the framework was applied to the same problem but with two very different selection (or evolutionary) mechanisms – stochastic (roulette-wheel selection) and deterministic (truncation selection). In the past tense model, the effects of genetic influences are simulated through variations in the neuro-computational properties of ANNs, and the effects of environmental influences are simulated via a filter applied to the training set. The approach uses a population of twins to disentangle genetic and environmental influences on past tense performance and to capture the wide range of variability exhibited by children as they learn English past tenses. This approach allows modelling of both individual differences and development (within the lifespan of an individual) in a single framework. Finally, the approach permits the application of Selection on developmental performance on the quasi-regular task across generations. This is an important aspect that distinguishes the current work from others reported in literature for the past tense formation problem, setting individual differences within an evolutionary framework. An experimental evaluation of this model focusing on individual differences in performance is then presented. The experiments led to some interesting findings such as: applying selection on the individual's performance level in a quasi-regular task such as past tense acquisition resulted in the emergence of divergent behaviours depending on initial conditions – both genetic and environmental; once selection started targeting a particular aspect of the task

domain, it behaved similarly to a traverse of Waddington's epigenetic landscape; and selection based on a stochastic method such as roulette-wheel, when combined with sexual reproduction method for population generation, had a limiting effect on the final behavioural (or performance) levels achieved. The findings validate the effectiveness of the method within an evolutionary setting and provide the basis for future work to capture population-level differences within a developmental setting.

Chapter 4 extends the BG-inspired model to transfer learning. The focus of this chapter, from a neuro-evolutionary perspective, is to evolve individuals (ANNs) capable of learning task(s) different from those for which they have been selected for. In such a situation, the members of the population have to evolve (or become fitter) at the population level on the evolutionary (or source) task and also learn various other (target) tasks. In this chapter the basic concepts of transfer learning have been discussed and then the literature review is presented, categorised according to the four key issues in transfer learning – what to transfer, how to transfer, when to transfer and how to assess task relatedness. Analysis shows that transfer learning, especially when used in conjunction with computational intelligence methods has been successful in nearly all kinds of applications. However, there are still several research challenges in the field, some of which include: most methods of transfer learning implicitly assume that the source and target tasks are somehow related to each other; there is a lack of a reliable theory of task relatedness that could be used as a benchmark and successfully applied in every scenario; most current methods developed for heterogeneous transfer focus only on improving performance on the principal (or target) tasks and risk of negative transfer has not yet been eliminated. This chapter addresses some of the aforementioned issues through BG inspired framework for transfer learning. The model uses ANNs as computational models capable of learning various heterogeneous tasks in an evolutionary framework. The proposed method spans transfer learning systems and multi-task learning systems, incorporating "good/useful" features of both, and then combines them with principles of BG. This work draws an analogy between genes and intrinsic parameters of ANNs, and the training dataset and the environment. This method therefore, imitates more closely learning as it happens in human beings – taking into account both structure and environment where the learning system is placed. By using same genetic range and environmental proportion for all tasks, our approach transfers the ***ability to learn*** across heterogeneous tasks. The interaction between quality of environment (i.e. filtered training set) and good (or not-so-good) genes (i.e. encoded ANN parameters) gives networks the *ability to learn* a given task. Thus using the same quality of training set and same neuro-

computational parameters leads to transfer of *ability to learn* across different tasks rendering it general. Two key factors were identified that could potentially modulate the performance of this model – selection operator and nature of source task. The transfer approach uses population of ANN twins and exploits the notion of heritability to assess task relatedness. Heritability is a useful statistic because it is scalable across potentially very large numbers of computational parameters (and their interactions) that contribute to the variation in learned high-level behaviours, or in this case, the outcome of learning for a set of ANNs. Twin studies provide an exact computation of heritability and this leads to an interesting finding that the direction of change in heritability has the potential to act as a mechanism for identifying task relatedness, which extrapolates to different task domains, and consequently avoids negative transfer.

Chapters 5 and 6 present the experimental evaluation of the BG-inspired transfer framework. The transfer approach has been applied to heterogeneous tasks. In chapter 5, the different heterogeneous tasks and their respective dataset descriptions are discussed. In chapter 4 it was established that the behaviour of the transfer model is potentially modulated by – type of selection operator and nature of source task. In order to test this hypothesis, the performance of the model was explored in different lineages, i.e. combinations of genetic and environmental influences. Overall ten replications of the model were tested, each with a twenty-generation duration. Each scenario was characterised by its own initial population (produced with random binary genomes) and unique values for the other heuristics involved, such as initial weights. The evolutionary methodology was then applied to each of these model instantiations, such that they all shared the same range of variation for genetic and shared environmental influences. At the same time, however, they were unique, for each of them began with a different initial population created from random binary genomes. Thus, having ten replications (*r1*, *r*2… *r10*) of the model aided in evaluating the robustness of the method. The first six replications were dedicated to investigating the effects of selection operator (roulette wheel selection for replications 1, 2 and 3 and truncation selection for replications 4, 5 and 6; the source task, was kept same for all 6 replications) on the behaviour of the transfer model whilst the remaining four replications were used for probing the modulatory effects of nature of source tasks. In the experiments reported in these two chapters, populations with over **200,000** neural networks in total were trained on five different tasks. The experiment results uncovered some interesting corollaries such as - evolution (via selection) and learning (i.e. ANN training) interact throughout lineage and result in different overt behaviours. The aforementioned interaction is of circular nature, wherein selection provides ANN populations with capacity and ability to

learn and thus constrains the behavioural outcome i.e. accuracy levels. By contrast, the performance levels attained after training (i.e. learning) determine fitness which in turn regulates what type of networks get chosen for breeding the next generation and thus in a way indirectly limit what type of intrinsic factors future generations will inherit. Further, the type of selection operator being used, namely stochastic or deterministic, modulates the accuracy levels achieved by ANN populations. Next, results confirmed that heritability acts as an identifier of task relatedness. Heritability informs us whether given tasks are targeting the same neurocomputational parameters varying within similar ranges. Consequently, the chances of improvement in accuracy are enhanced if selection is acting on one of these tasks. Thus it is easier to predict if transfer will be successful and thereby avoid negative transfer. This ascertains that heritability could be used as an identifier of task relatedness irrespective of the nature of tasks. Additionally, the trends emerging from the results demonstrate that the effect of selection (owing to shift in range of intrinsic properties) on different tasks is consistent throughout the replication, similar to Waddington's epigenetic landscape discussed in Chapter 3. This behaviour is not necessarily desired in machine learning, especially if performance starts worsening. This is where the analysis of proportion of variance due to genetic and environmental factors becomes more relevant. This analysis revealed which of these neurocomputational or environmental factors caused most behavioural variance and consequently informs us which of them is exploited most by ANNs for acquiring certain task. Thus training could be biased towards the more important/contributing factor to boost performance accuracy.

Finally, **Chapter 7** concludes this thesis with a summary of research and findings, and an outline of the thesis contribution. It also identifies directions for future work and ways in which they could be addressed.

Chapter 2 Behavioural Genetics inspired framework for evolving populations of neural networks: combining learning & evolution

2.1 Overview

In biological evolution, learning and evolution are two principal forms of adaptation that differ in time and space. Evolution is a process involving selective reproduction and substitution based on presence of population of individuals displaying some variability. Learning, on the other hand, is a set of adjustments taking place within each individual in the population during its own lifetime. Over the last decades, researchers in the field of neuroevolution have used artificial evolution techniques, i.e. genetic algorithms and learning techniques viz. artificial neural networks to study the interaction of learning and evolution with the intent of looking at the advantages, in terms of performance, that this interaction leads to. This chapter is organised as follows: first the basics of evolution and learning are discussed in Section 2.2. This is followed by a review of the various frameworks for combining evolution with learning in Section 2.3. The field of behavioural genetics is then introduced in Section 2.4 and the proposed framework is presented in Section 2.5. Finally the summary and chapter contribution is given in Section 2.6.

2.2 Evolution and Learning and interactions therein

Evolution is a type of adaptation that captures relatively slow environmental changes that involves several generations, i.e. evolution operates at phylogenetic level. Learning includes various set of mechanisms that lead to adaptive changes in an individual during its lifetime, i.e. learning operates on ontogenetic level. However learning also has costs. It increases the unreliability of the evolved individuals and it involves a delay in the ability to acquire fitness and thus during the learning phase, the individuals will have sub optimal behaviours (Nolfi and Floreano, 1999 and references therein). In addition, evolution operates on the genotype (set of alleles constituting the genetic makeup of individual) whereas learning affects the phenotype (observed characteristics of the individual) and

phenotypic changes cannot directly modify the genotype (Nolfi and Floreano, 1999; Nolfi and Parisi, 2002). An important distinction here is between the genetic code inherited from parents, i.e. the genotype, and the complete individual formed according to information contained in genotype along with other developmental factors i.e. the phenotype.

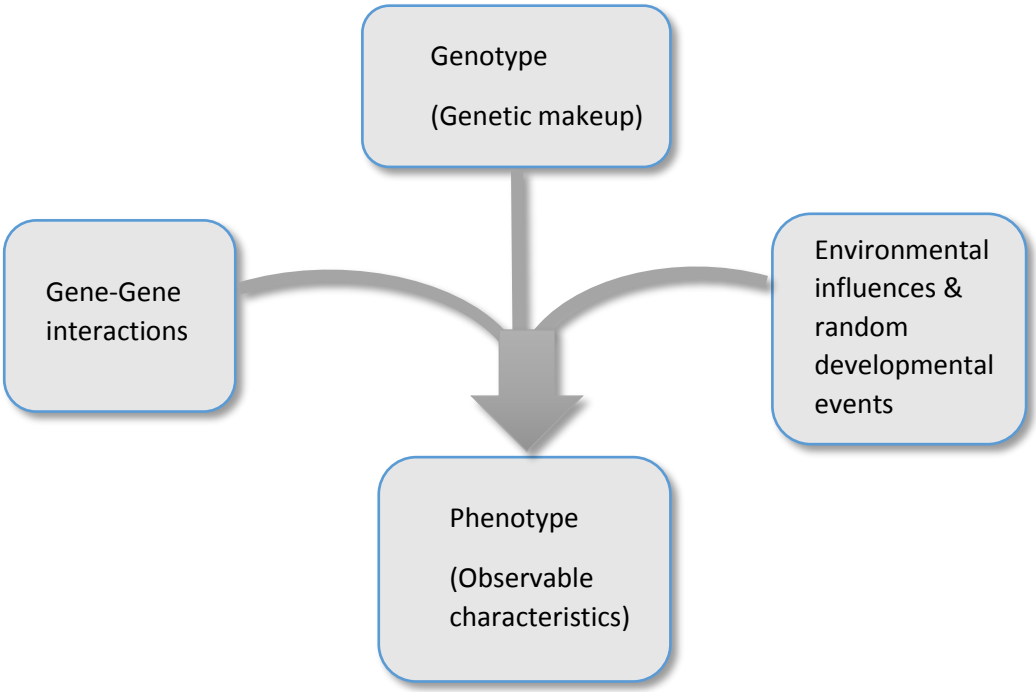


Figure 2.1: Difference between Genotype and Phenotype

Figure 2.1 illustrates that there are three contributing factors to the formation of phenotype. These are the genotype, environmental contributions to each gene, gene to gene interactions, and each behaviour might depend on multiple genes and finally random developmental events. Research in the field of neuroevolution suggests that, within an evolutionary perspective, learning can have numerous different adaptive roles (Nolfi and Floreano, 1999 and references therein):

- It lets individuals adapt to changes in the environment that occur during the lifetime of that individual.
- It lets evolution use the information extracted from the environment ergo channelling evolutionary search.
- It can help and guide evolution.

How can learning help and guide evolution?

The idea of interaction between learning and evolution was first proposed by Baldwin (1896) and Lloyd Morgan (1896) and is commonly referred to as the Baldwin Effect. Waddington (1942) also proposed a similar kind of interaction which is called canalisation or genetic assimilation. The key concept in all the aforementioned theories is that what a species must initially learn during each individual's lifetime, can overtime become part of the genetic makeup of that species, i.e. what is initially learned eventually becomes innate (Munroe and Cangelosi, 2002). This effect can also be interpreted as a two-step process:

- step1 – individuals capable of adapting their behaviour/trait according to the environment through lifelong learning occupy the reproductive population;
- step2 – evolution finds innate solutions that could replace the learned trait due to cost of learning. This step is also known as genetic assimilation.

The structure of all cognitive abilities that we possess like language acquisition, reasoning and likewise arise from the interactions between two complex adaptive systems – learning and evolution. However, there is a third block – the acquired trait/performance level that also plays an important role in turning the learnable to innate. Figure 2.2 explains the interactions between these three.

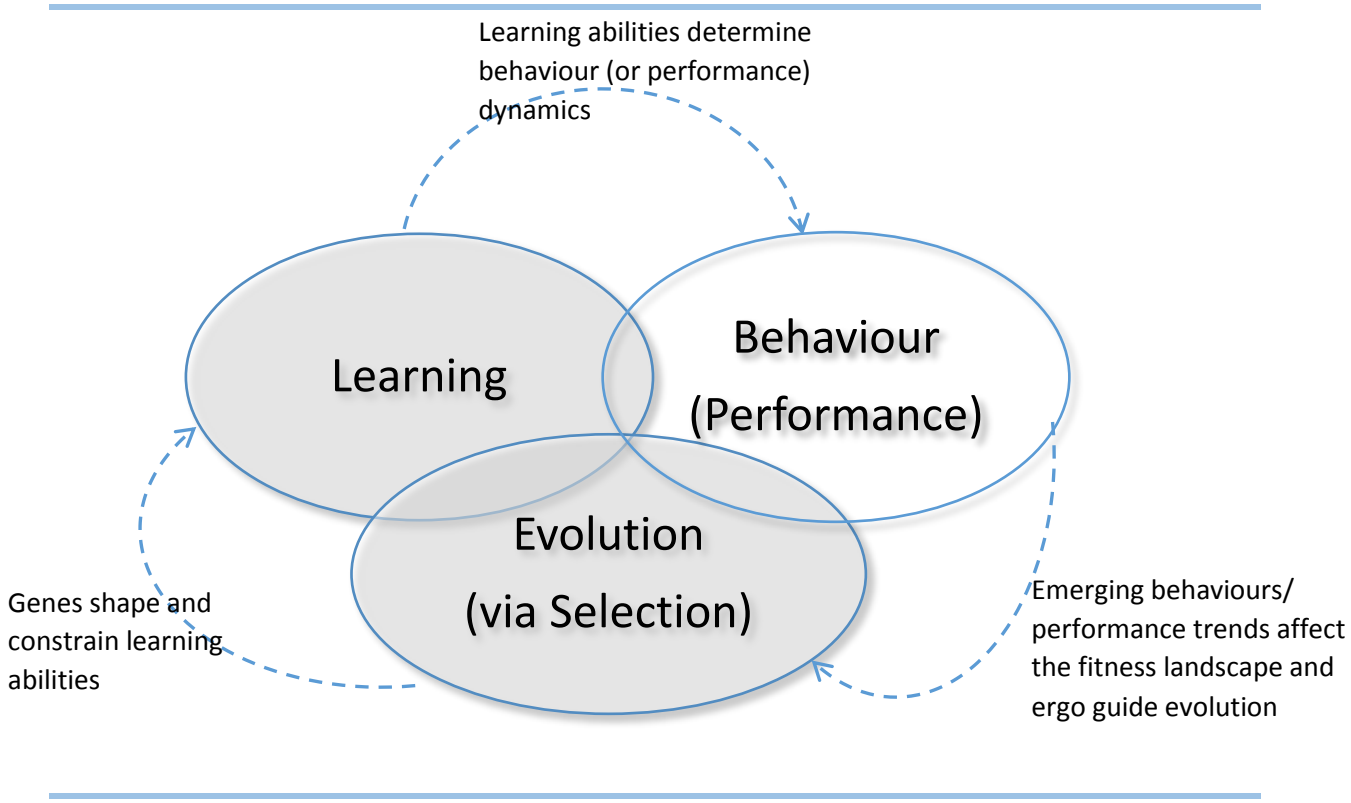


Figure 2.2: Interactions between learning and evolution & role of behaviour therein

The acquisition of complex cognitive abilities begins with evolution/nature providing random genotypes to population members. The genotype displays some plasticity in its interaction with environment, i.e. learning (or nurture). The degree of plasticity, however, varies from individual to individual which implies that genes both shape and constrain learning abilities. This degree of plasticity governs resulting behaviour or, in other words, how successfully a trait/skill/task is acquired. This behaviour in turn determines the fitness landscape, i.e. selective reproduction (evolution) is dependent on behaviour. Thus a circular relationship exists wherein each factor depends on the other. Another vital point is that evolution, in terms of selective reproduction, acts directly on genotypes. However, the fitness landscape which guides evolution is determined from the phenotype. Figure 2.3 explains this relationship.

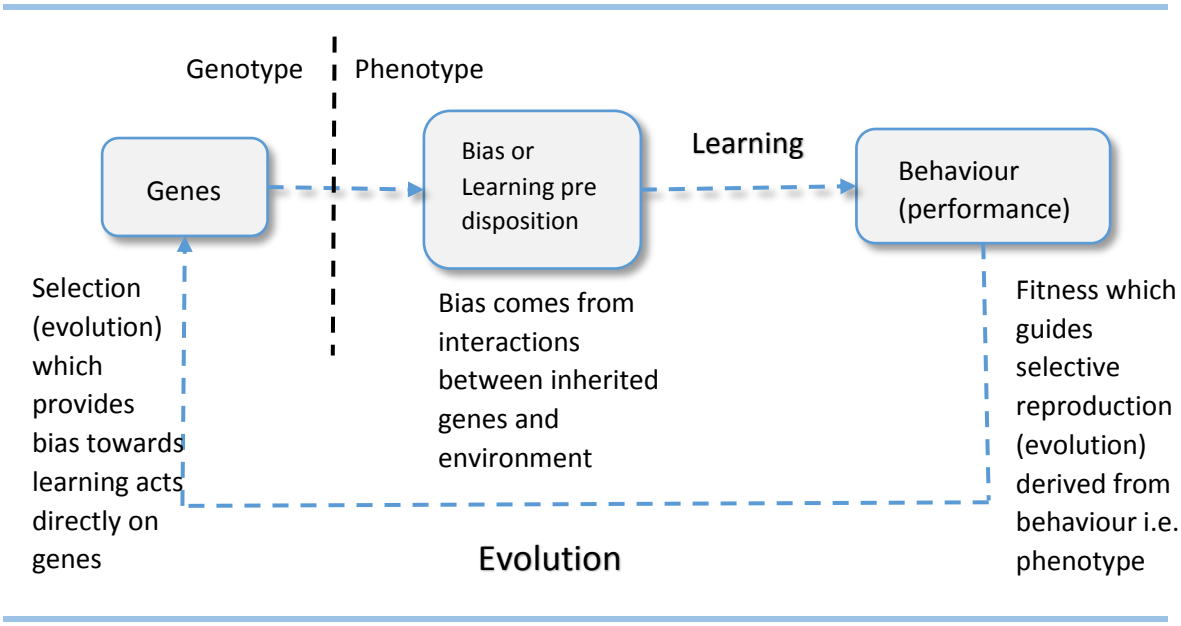


Figure 2.3: Relation between genes, learning bias and acquired behaviour

Figure 2.3 shows that the fitness of the individual, which affects selective reproduction, acts on the phenotype whereas what is inherited from parents is the genotype. Also, it shows that learning makes the fitness landscape smoother, which in turn simplifies the search for reproductive selection performed by evolution (Nolfi and Floreano, 1999). Behavioural traits, or performance levels, which initial generation(s) acquires through learning gradually become part of learning bias or predisposition for subsequent generations. This happens when genes responsible for desired trait get fixed on the correct values. Thus more

and more individuals with part of their genes set on right values and a remaining part of plastic genes get selected because fitness is inversely proportional to number of learnable/plastic genes.

Hence, the important conclusions that can be drawn are - learned behaviours might affect the direction and rate of evolutionary change via selection. Additionally, the probability that an individual (and thus population) can acquire a particular trait (or learn a task) largely depends on the traits (or performance on that given task) acquired by the learners in preceding generations.

2.3 Combining Evolution and Learning using ANNs

Over last decades, many researchers have used artificial evolution techniques, i.e. genetic algorithms and learning techniques viz. artificial neural networks to study the interaction of learning and evolution with the intent of looking at the advantages, in terms of performance, that this interaction leads to (Hinton and Nowlan, 1987; Nolfi and Floreano, 1999). This interest spawns from many different perspectives. The first perspective is artificial intelligence, where the aim is to let intelligent systems solve problems and learn new tasks by itself without expert (human) intervention. Another view is that of artificial life, wherein the idea is to create intelligent artificial lifeforms capable of lifelong learning and surviving in potentially dynamic environments, based on nature's principles of evolution.

Artificial neural networks (ANNs) are computational abstractions of the biological information processing system. A special class of ANNs, where evolution is another form of adaptation along with learning, are referred to as Evolutionary artificial neural networks (EANNs) (Yao, 1999). These neural networks are evolved using evolutionary algorithms, which are a class of population based stochastic search methods inspired from Darwinian evolution (Floreano et al., 2008). These algorithms complement the standard learning algorithms such as backpropagation. In these methods, the characteristics of artificial neural networks are encoded in an artificial genome and then evolved to a performance benchmark (Floreano et al., 2008). The generic phases in evolving artificial neural networks using evolutionary algorithms are explained below:

1. Randomly create an initial population of different artificial genotypes, each of which encodes free parameters e.g. connection strengths and/or architecture and/or learning rules
2. Train (i.e. learning) and Evaluate each individual of the population of networks to determine the fitness (based on performance).
3. Based on chosen ***Selection*** criterion, the ***selected*** networks reproduce (sexually or asexually) by creating copies of their genotypes with addition of changes introduced by genetic operators like cross over.
4. Repeat steps 1-3 for number of generations till the networks satisfy performance/termination criterion set by researcher

One of the main benefits of using evolutionary algorithms for design and tuning of artificial neural networks is that evolution can be pooled with learning. The combination of evolution and supervised learning provides a powerful synergy between complementary search algorithms (Belew et al., 1990; Floreano et al., 2008). For instance, gradient-based learning algorithms such as backpropagation are sensitive to the initial weight values, which may considerably affect the quality of the trained network. In these situations, evolutionary algorithms can be used to find suitable initial weight values of networks to be trained with backpropagation. The fitness function is computed using the residual error of the network after training with backpropagation on a given task. Experimental results reliably show that networks with evolved initial weights can be trained significantly faster and better than networks with random initial weights (Floreano et al., 2008). Another benefit is that along with initial weights, the genome can also encode the values of the learning rate and of other learning parameters of gradient based search algorithms, such as the momentum, slope of activation function to name a few. Another important feature is that the evolved parameters like initial weights, learning rate and likewise are not directly coded back into the genotype, because these methods follow the Darwinian approach, i.e. phenotypic changes cannot directly modify the genotype.

Thus it is well established that the relation between learning and evolution is highly complex. Over the years, many models have been proposed but most of them are either concerned with how learning can guide evolution (Belew, 1990; Hinton and Nowlan, 1987; Nolfi et al., 1994; Smith, 1986) or how weights and/or architectures can be evolved (Muhlenbein and Kindermann, 1989; Paredis, 1991). From Baldwin's theory (Baldwin, 1896) it is now long known that learning affects natural evolution. Empirical evidence also

suggests that the same is true in case of artificial evolution and learning (Floreano et al., 2008; Nolfi and Floreano, 1999). Many evolutionary methods/frameworks have been developed over the years for neuro-evolution. These have been discussed and summarised in the following subsection.

2.3.1 Frameworks for combining Evolution with Learning

One of the first computational model to show that learning facilitates evolution was proposed by (Hinton and Nowlan, 1987). Their results proposed that addition of learning results in smoothing of the fitness surface area around the optimal combination of genes (wherein genes encoded weights), which can be found by genetic algorithms (Hinton and Nowlan, 1987; Floreano et al., 2008). However, this model had one limitation and that is they assumed learning space and evolutionary space to be completely correlated. These two spaces are completely correlated if genotypes which are close in evolutionary space correspond to phenotypes which are close in phenotypic space (Floreano et al., 2008). Further extending the research of (Hinton and Nowlan, 1987), researchers like (Mayley, 1996), showed that by varying the cost of learning (i.e. loss of fitness during initial part of lifetime when an individual has sub-optimal performance) and correlation between learning and evolutionary space, led to the finding that adaptive benefits of learning are proportional to correlation between the two search spaces; the incorporation of traits initially acquired through learning is proportional to the correlation between the two search spaces and to the cost of learning; and finally that in some scenarios learning cost outweighs learning benefits.

Some researchers have also proposed that instead of using a Darwinian approach to evolution (i.e. an approach where learned or acquired traits are not encoded back into the genotype directly or wherein phenotypic changes cannot directly affect genotype), a more plausible and efficient approach to evolution can be achieved by following Lamarckian evolutionary theory. As per this theory, acquired traits are directly coded back into genotypes and thus transmitted to the offspring. Some authors like (Ackley and Littman, 1991) suggested that performing Lamarckian evolution computationally is easy and straightforward and they also showed that this approach is far more effective in stationary environments (Floreano et al., 2008). However, other research (Sasaki and Tokoro, 1997) has showed that combining learning and evolution by following Darwinian evolutionary

theory yields much better results than Lamarckian evolution when the environments are dynamic or when different individuals are exposed to different learning experiences – a scenario that is more true to real world cases in machine learning.

Generally speaking, evolution has been introduced into ANNs at roughly three key levels, evolution of connection weights, evolution of network architectures and finally evolution of learning rules. The evolution of connection weights introduces an adaptive, global search approach to training that has been used mainly in the reinforcement learning and recurrent network learning paradigm where gradient-based training algorithms often experience many difficulties. The evolution of architectures allows ANN's to adapt their topologies to different tasks without human interference and thus provides an approach to automatically design ANNs as both the network's connection weights and structures can be evolved. The evolution of learning rules can be considered as a procedure of "learning to learn" in ANN's wherein adaptation of learning rules is attained through evolution. It can also be viewed as an adaptive process of automatic detection of novel learning rules (Yao, 1999). These three broad categories are discussed and summarised below.

2.3.1.1 Evolving ANN connection weights

Weight training in ANN's is typically expressed as minimisation of an error function, such as the mean square error between target and actual outputs averaged over all samples, by iteratively fine-tuning connection weights. Many ANN training algorithms are based on gradient descent, which although has been greatly successful but still has a drawback. It often tends to get trapped in local minima of the error function and has difficulty finding global minima when the error function is multimodal and/or non-differentiable (Yao, 1999).

In such scenarios, evolution can be coupled with ANN learning and a training process can be formulated which evolves the connection weights in the environment determined by architecture and learning tasks (Ding et al., 2013; Miikkulainen, 2015; Schrum and Miikkulainen, 2014; Cardona et al., 2013). Such a neuro-evolutionary algorithm can then be used efficiently in the evolution to find a sub-optimal set of connection weights without computing gradient information. The evolutionary approach to weight training consists of two stages – the first phase involves deciding the type of representation of connection weights, i.e. binary or real valued and the second phase involves simulating evolutionary

processes wherein genetic operators being used like crossover and mutation have to be chosen (Yao, 1999). The main steps involved in evolving connection weights for ANNs are:

1. Generate a population wherein each individual (genotype) represents a set of connection weights.
2. Construct corresponding ANNs using these weights
3. Evaluate fitness of these ANNs according to mean squared error between actual and desired output, or some other suitable cost function; the higher the error the lower the fitness
4. Select parents for reproducing next generation based on their fitness
5. Apply genetic operators like crossover, mutation to selected parents and create offspring which form next generation
6. Repeat till termination criterion is met

The connection weights can be represented using either binary or real-valued representation. In binary representation, each connection weight is represented by number of bits of certain length and an ANN is encoded by concatenating all connection weights of the network in the genotype (Yao, 1999; Whiteley et al., 1990; Srinivas and Patnaik, 1991). The main advantage of using this representation is the simplicity in applying classical genetic operators to binary strings. The drawback however is in trade-off between representation precision and chromosome length. If the bits used to represent connection weights are too few training might suffer because certain combinations of real valued weights cannot be estimated accurately by discrete values. On the other hand, if the chromosome is too long then representing bigger and complex ANNs becomes difficult and inefficient (Yao, 1999). The real-values representation overcomes some of the drawbacks faced by binary representation. This representation scheme uses one real number for each connection. Thus, an individual is represented by real-valued vector and number of genes in chromosome is same as total number of connections between neurons (Yao, 1999; Montana and Davis, 1989; Fogel et al., 1990; Fogel et al., 1995; Yan et al., 1997; Porto et al., 1995). The drawback of this scheme is difficulty in use of traditional genetic operators like crossover and mutation, associated primarily with binary representation (Yao, 1999).

As discussed previously, the evolutionary training methods are attractive because they are capable of handling global search problems better in complex, multimodal and non-differentiable surface without depending on gradient information. However, a good body of research (Lee, 1996; Kinnebrock, 1994; Hung and Adeli, 1994; Likartsis et al., 1997) has also shown that using evolutionary algorithms, such as GAs, to search for near optimal set of initial weights and then applying gradient based algorithms, such as backpropagation, to perform local search from these initial weights is quite effective. Their results showed that this hybrid evolutionary/gradient based approach is more efficient than either of the two algorithms used alone.

2.3.1.2 Evolving ANN Architectures

The methods for evolving ANN connection weights, discussed in the previous subsection, mostly assume that the architecture of the ANN is predefined and fixed. The architecture includes the connectivity, or topology, and transfer function information for each node. Architectural design is very vital in effective application of ANNs because architecture has direct effect on networks information processing capabilities (Yao, 1999).

Given a learning task, a network with too few connections and linear nodes might not be able to learn the task due to limited capability. On the other hand, a network with too many connections and nonlinear nodes will overfit and thus fail to have good generalisation ability (Yao, 1999). Generally, designing ANN architecture requires expertise and there is no systematic way to determine the appropriate architecture automatically. However, there have been some attempts to design network topology automatically by means of either constructive and/or destructive algorithms. A constructive algorithm starts with minimum topology, i.e. with minimum number of hidden layers, nodes and connections and adds new layers, nodes and connections as and when needed during training. On the other end, destructive algorithms begin with maximum architecture and gradually removes unnecessary layers, nodes and connections. Nevertheless, research has indicated that such structural hill climbing methods are prone to becoming stuck in structural local minima and these are capable of investigating limited topological subsets instead of complete class of network architectures (Angeline et al., 1994).

An alternative way to design optimum network architecture and avoiding the aforementioned issues is to evolve the network architectures (Ding et al., 2013; Risi and

Togelius, 2015; Floreano et al., 2008; Ahmadizar et al., 2015; Fister et al., 2015). This process for evolving the optimum architecture design of ANNs can be articulated as a search problem in the space of possible architectures wherein each point represents an architecture. Given some performance optimisation criteria viz. lowest training error or lowest ANN complexity, etc. the performance level of all architectures forms a discrete surface in the space and thus finding an optimum architecture design becomes equivalent to finding the highest point on this surface (Yao, 1999).

Like in case of evolution of connection weights, the evolution of architecture also involves two phases – the first is determining the genotype representation scheme of architecture and the second involves determining what type of evolutionary algorithm must be used to evolve architectures. The main concern in the first phase is to decide how much information about architectures should be encoded in the genotype. Depending on the amount of information chosen to be specified, there are two main encoding schemes most commonly used. The first one is direct encoding in which all details about ANN topology like all nodes and connections etc. are specified using binary representation (Schaffer et al., 1990; Marin and Sandoval, 1993; Alba et al., 1993). Evolved neural networks with direct encoding have been applied to various problems like data classification (Chandra and Yao, 2006), game playing (Chellapilla and Fogel, 2001) and control of robot swarms (Trianni et al., 2007) amongst others. Although popular, one potential issue with direct encoding scheme is scalability. A large network will need big connectivity matrix and this will in turn increase the computation time of evolution. A solution to this problem is to use domain knowledge in order to decrease search space dimensionality. For instance, if the network is fully connected feed forward then its architecture can be encoded by providing number of hidden layers and number of hidden nodes in each layer, thereby reducing the length of chromosome (Yao, 1999; Schaffer et al., 1990). Though, this requires adequate domain knowledge and expertise which is quite difficult to get hold of. In addition there is also a risk of missing some good solutions if the search space is restricted manually based on expertise.

In order to tackle the problem of scalability and also avoid the aforementioned issues, yet another encoding scheme is used, called the indirect encoding. In this approach only the most important aspects of neural architecture are described like number of hidden layers, number of hidden neurons (Kitano, 1990; Harp et al., 1990; Yao and Shi, (1995). There are two ways for indirect encoding. The first is parametric representation which incorporates

information about crucial parameters of network structure like number of hidden layers, number of neurons and connectivity between layers etc. into genotype (Yao, 1999; Harp et al., 1990). Though this method enables compact genotype representation, however, an evolutionary algorithm cannot perform global search due to access to only limited information contained in genotype and thus may not be very efficient in finding a compact network with good generalisation ability. The second indirect encoding approach includes developmental rule representation (Yao, 1999; Yao and Shi, 1995). In this scenario, developmental rules used to build a network architecture are encoded in genotype. This representation scheme too has limitations such as, it might lead to huge ANN topologies, cannot evolve architecture and weights at the same time amongst others.

After choosing a suitable representation scheme, most evolutionary methods follow the following generic steps in order to evolve network architectures:

1. Generate a population where each individual is decoded into possible ANN architecture
2. Train each ANN with decoded architecture by predefined learning rule starting from different random initial weights and learning rule parameters
3. Evaluate fitness of each individual as per training performance result and some other criterion like complexity of network architecture
4. Select members from current population for breeding based on fitness
5. Generate offspring by applying genetic operators on chosen parents and these offspring constitute next generation

This process stops when a network with optimum architecture and performance has been found. This approach for evolving the architecture is applicable not only to the topological structure of networks but also to the transfer function of the nodes (Yao, 1999). The evolutionary approach described above is mostly employed for evolving network architectures only- the weights are learned after a suitable architecture has been found. However, this has one main drawback that is evolution of architectures without evolving weights often leads to noisy fitness estimation (Yao and Liu, 1997). The noise gets introduced through two main reasons: First being random initialisation of weights wherein

different weights often lead to different training results. Ergo, same genotype might result in quite dissimilar fitness due to different random initial weights. Second cause is the training algorithm. Diverse training algorithms lead to different results even if they have started from the same set of initial weights. Simultaneously evolving both the architecture and connection weights can alleviate this problem (Marin and Sandoval, 1993; Alba et al., 1993; Srinivas and Patnaik, 1991; Bornholdt and Graudenz, 1992; Angeline et al., 1994; Stanley and Miikkulainen, 2002). Evolving them simultaneously makes it possible to start evolution with simple solutions and then gradually make them more complex, a process inspired from biology and powerful approach in machine learning in general. This is why these methods like (Stanley and Miikkulainen, 2002) have been widely applied to many problems such as pole balancing (Stanley and Miikkulainen, 2002), robot control (Stanley and Miikkulainen, 2004), and computer games (Reisinger et al., 2007) to name a few. The simultaneous evolution of weights and architecture requires each individual to be fully represented network with complete weight information as well in the chromosome and therefore having no difference between genotype and phenotype fitness. This makes fitness evaluation more accurate (Yao, 1999).

2.3.1.3 Evolving ANN Learning Rules

The training algorithm of ANNs perform differently when applied to different architectures, and thus the choice of suitable training algorithm, or learning rules used to modify weights, depends on the network architecture (Yao, 1999). Selecting the appropriate learning rule proves to be hard especially when there is lack of prior knowledge about the type of network architecture. Therefore, there is a need to develop automatic and methodical techniques to adapt the learning rule to the network architecture and learning task, e.g. using evolution to evolve dynamically the most suited learning rules (Ding et al., 2013; Mouret and Tonelli, 2014; Moriarty and Miikkulainen, 2016; Floreano et al., 2008).

Evolution has been applied to learning rules in two ways. The first approach involves adaptive fine-tuning of learning algorithm's parameters such as learning rate and momentum. Researchers (Belew et al., 1990; Kim et al., 1996) have used evolutionary procedures to find learning parameters for gradient based learning algorithms where a network architecture was pre-defined. However, the drawback in this case was that parameters evolved were optimised towards architecture instead of being relevant to

learning. Efforts have also been made to encode parameters of gradient based learning algorithms in genotype along with network architecture (Harp et al., 1989). This simultaneous evolution of learning parameters and architecture expedites exploration of interactions between network architecture and learning algorithm so that an ideal amalgamation of learning algorithm and architecture can be established (Yao, 1999; Harp et al., 1989).

The second approach to evolve learning rules involves adapting the actual learning rule itself through evolutionary process. This promises to enhance network's capability to successfully adapt in dynamic environment scenarios. Unlike any other kind of previously discussed evolutions, the evolution of learning rule has to deal with dynamic behaviour of network. Therefore the key issue is to find a way to encode dynamic behaviour of learning rule into static genotype. Some research has been done in this area, however it is beyond the scope of this work and (Yao, 1999) can be referred for more details on this.

The general steps involved in evolving learning rules for ANNs are summarised below. This process continues till specified number of iterations or until population converges.

1. Generate an initial population where each individual can be decoded into learning rule/ learning rule parameter
2. Train a set of ANNs with random architectures and weights using the decoded learning rules
3. Evaluate fitness of each individual i.e. learning rule according to training performance
4. Based on fitness measure select members to breed next generation
5. Apply genetic operators on selected parents to create offspring which constitute next generation

Researchers like (Chalmers, 1990) have used mean squared error as fitness function and trained neural network with single layer of connections on different linearly separable tasks. The initial weights were set to small random values close to zero. The evolutionary algorithm evolved a learning rule similar to Widrow and Hoff rule. Similar work was also done by (Fontanari and Meir, 1991). On the other hand, (Dasdan and Oflazer, 1993) employed similar strategy but instead evolved unsupervised learning rules for classification

tasks. The authors reported that evolved learning rules were more powerful than corresponding human-expert designed rules (Floreano et al., 2008). Some researchers like (Baxter, 1993) tried evolving complete network i.e. weights, architecture and learning rule in single level of evolution. They only considered ANN's with binary threshold nodes, so the weights could only be 0 or 1. The number of nodes in ANN's were also fixed. The learning rule only considered two Boolean variables. Although their experiments were simple, their efforts confirmed that complex behaviours could be learned and the ANN's learning ability could be improved through evolution (Yao, 1999).

Research about evolving learning rules is still going strong (Baxter, 1993; Chalmers, 1990; Bengio et al., 1992; Fontanari and Meir, 1991; Floreano et al., 2008; Mouret and Tonelli, 2014; Moriarty and Miikkulainen, 2016). This is an important direction because it provides an automatic way of optimising learning rules and modelling interactions between learning and evolution. Additionally, this will also help in exhibiting creative process since evolved learning rules will be able to operate within complex and dynamic environment (Yao, 1999).

2.3.2 What is next?

It can be inferred from the discussion above that it has now been well established that neither pure evolutionary algorithms nor purely local search techniques like those based on gradient information are well suited to fine tune search in complex combinatorial spaces. However, hybridising the two said techniques can greatly improve the efficiency of search (Krasnogor and Smith, 2005). This combination of learning and evolution by means of hybrid algorithms has been very successful and has been applied in number of different areas like robot learning, automatic programming, game playing, operational research and optimisation amongst others. These have also been used to study and enhance models of population genetics, economics, immune systems, and the interactions of evolution and learning and many more application areas (Krasnogor and Smith, 2005). From an optimisation perspective, these hybrid approaches have fared much better both in terms of efficiency i.e. needing much fewer evaluations to find optima and more effective i.e. being able to find better or higher quality solution compared to more traditional approaches (Krasnogor and Smith, 2005). However, despite all these benefits, the process of designing effective and efficient neuro-evolutionary approaches is still fairly ad-hoc and is masked

behind problem-specific particulars (Krasnogor and Smith, 2005). Also most of the methods have been developed and tested in relation to work on/for only single task. However, during their lifetime, individuals of any species acquire more than one behavioural traits, some of which are evolved, i.e. selected for and the others are learned. Therefore there is a need for a more generic and systematic neuro-evolutionary framework/approach which is not bound by problem/task specifics and is applicable and adaptable to various tasks belonging to any domain (Lehman and Miikkulainen, 2013; Miikkulainen, 2015; Risi and Togelius, 2015).

Given the observations collected from previous research efforts, in this thesis a neuro-evolutionary approach/framework is proposed which enables a population of artificial neural networks to:

1. Optimise: this implies that the population should evolve or get fitter at a given evolutionary (or main) task over generations at population level;
2. Adapt: this implies that the evolving populations should be able to adapt to changes in the environment. These changes might be slow and subtle as in concept drift or they might occur abruptly as in concept shift. This basically involves getting fitter at more than one task or scenarios wherein learning tasks are different than evolutionary task;
3. Model interactions between learning and evolution – thereby exemplifying how learning can shape and constrain evolution

The proposed approach is systematic and is not dependent on problem domain. It can be easily applied to any given set of learning and/or evolutionary tasks. Evolutionary task is the task or the behavioural trait that is being chosen for or in simpler terms, the members of population have to become fitter at this task at a population level for evolution to work. For example, ability to gather food in birds might be an evolutionary trait and only those individuals who are good at this get selected for breeding. For species to survive, they've to keep becoming better at this particular task i.e. food gathering. On the other hand there are some trait(s)/task(s) that the individuals of the population have to acquire/learn during their lifetime at an individual level. These task(s) or trait(s), however are different from what they've been selected for. For instance, continuing with the previous example, food gathering is the evolutionary task, however the species members also have to learn to avoid being hunted. So although they are being selected for because of their food gathering skills,

but still it is important that they individually become fitter at being able to avoid being hunted. Hence avoiding being hunted is the learning task in this example.

The proposed method involves evolving two main aspects of ANNs – the architecture (e.g. number of hidden units, node transfer function- steepness/slope of logistic activation), and learning rule's algorithmic parameter (initial learning rate). These are the formational parameters of ANNs which either increase or decrease their ability to acquire a new task. These parameters correspond to how a network is built and thus relates to network's capacity to learn (via number of hidden units) and it governs how the network adapts and hence provides a network with the ability to learn (via initial learning rate). The steepness of the activation function corresponds to the activation dynamics acting within each network. Research has shown that transfer function is an important part of ANN architecture and has significant impact on its performance (Yao, 1999). Also in the proposed approach the network's initial weights were not encoded in genome to be evolved. Instead these were continuously modified during the lifetime via learning process in which genetically inherited information interacts with information coming from external environment (Nolfi and Floreano, 1999).

The neuro-evolutionary approach proposed in this work draws inspiration from the multi-disciplinary field of behavioural genetics (BG). This is a field of study that examines the role of genetics and environmental influences on behaviour. Research in the field shows that genes and environment interact throughout development to shape differences in behaviour (Plomin et al., 2013). This makes this field an apt choice for using to develop new method for neuro-evolution since the main building blocks in both BG and combination of evolution and learning are the same – genotype (genes which shape and constrain learning); environment (which provides learning bias) and finally the interactions between genes and environment. All three factors together lead to phenotype (behaviour). Figure 2.4 demonstrates this relation. In the following section the concepts of BG related to this thesis will be discussed in detail.

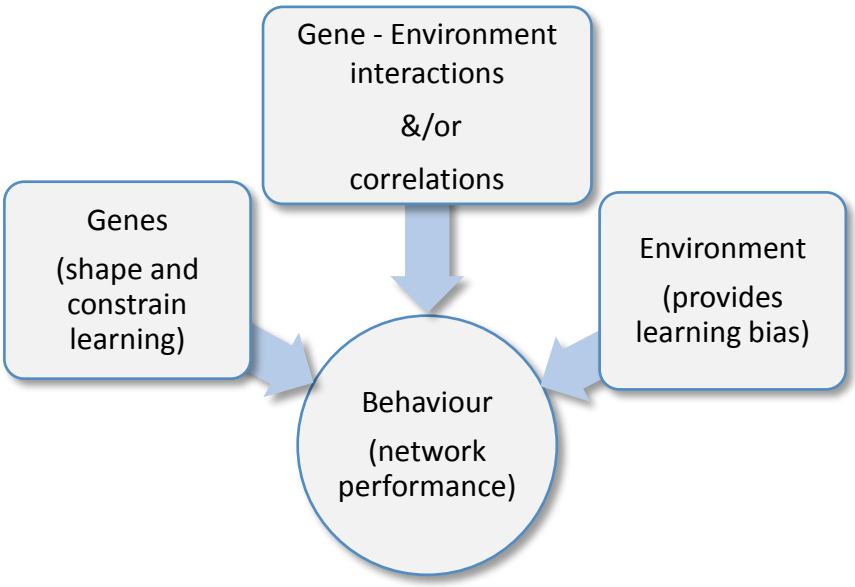


Figure 2.4: Basic components of BG

2.4 Behavioural Genetics

Behavioural Genetics is a field of study that examines the role of genetics in individual differences in human behaviour. Behaviour is the most complex phenotype as it reflects the functioning of the complete organism; it is dynamic and changes in response to the environment (Plomin et al., 2013). This field is concerned with the study of individual differences, i.e. knowing what factors make individuals within a group differ from one another. It also estimates the relative importance of genetic and environmental factors in causing individual differences. Thus, the behaviour or phenotype is the result of genetic factors together with environmental factors. Some of the important concepts and findings of the field are discussed below.

2.4.1 Genotype, Phenotype and Environment

A gene is the basic unit of genetic information that determines the inherited characteristics for example eye colour, hair colour, left or right handedness to name a few (Pearson, 2006). Each gene has two or more alternate forms that arise due to mutations and are present at the same location (or locus) on a chromosome. These alternate forms are called alleles.

These alleles can either be dominant (in notation, represented in upper case always) or recessive (in notation, represented in lower case). An individual with even one dominant allele will display the said trait whereas in latter case both alleles should be recessive for trait to be expressed.

The genome or genotype is the complete genetic information of an individual, i.e. an individual's combination of alleles (Plomin et al., 2013). It is the measure of the base composition of an individual, the representation of a species (Plomin et al., 2013). In other words, it serves as a set of instructions about how to form an organism of a particular species or group. Figure 2.5 explains this further.

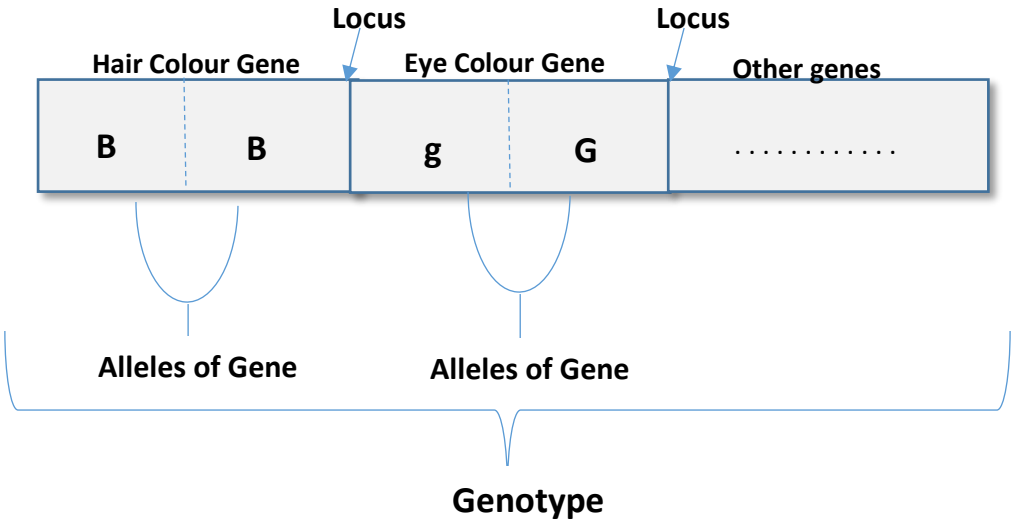


Figure 2.5: Schematic Genotype and its constituents

The environment is understood very broadly. It basically encompasses everything that influences an individual's phenotype that isn't part of their genotype (Plomin et al., 2013). This will be discussed in more detail in coming subsection.

The phenotype is the set of observable characteristics of an individual which result from an interaction between the individual's genotype and environment. For simplicity consider an example with a genetic contribution to the phenotype's chosen aspect, based on Figure 2.5, if B → brown, g → green and G → grey then the phenotype of this individual will have brown hair and grey eyes.

2.4.2 Methods employed in BG research – twin studies & GCTA

The most widely employed way to study the contribution of genetic and environmental factors towards behaviour is observational studies. These involve assessing and comparing relatives such as twins or siblings, families and adopted children. This category of research is called quantitative genetics and it aims to inspect the extent to which variation in a particular behavioural trait is influenced by genetic factors in a population. This approach relies on statistical methods to examine and compare groups of individuals without focusing on specific genes.

Twin studies are the workhorse for quantitative genetics. This design capitalises on the quasi-experimental scenarios triggered by twinning to measure comparative contribution of nature and nurture (Plomin et al., 2013). It is a method very extensively used to disentangle genetic from environmental sources of influences between relatives (twins) (Plomin et al., 2013). In Behavioural Genetics, twin studies are widely employed to untangle genetic and environment effects on behaviour. Twin pairs are matched for age, family and other social influences. They are either genetically identical (genetic relatedness of 1.0 for *monozygotic,* MZ, or identical twins) or as similar as siblings (genetic relatedness of 0.5 on average for *dizygotic,* DZ, or fraternal twins) and, to an approximation, share the same environment (applicable for both MZ and DZ twins based on the *Equal Environments Assumption* – it assumes that environmentally caused similarity is roughly the same for both types of twins raised in the same family) (Plomin et al., 2013). The difference in the similarity in performance between MZ or DZ twin pairs, along with assumptions about their similarity of environment, allows inferences to be drawn about the influence of genetic relatedness on behaviour (Plomin and Spinath, 2004). Environment plays a vital role in twin studies and is therefore discussed in detail in Section 2.4.3.

In addition to twin study method, a new method called GCTA i.e. Genome-Wide Complex Trait Analysis (Yang et al., 2011), for estimating genetic influence using DNA is a fresh addition to the armamentarium of quantitative genetics. The significance of this method is that it estimates the net effect of genetic influence using DNA of unrelated individuals rather than relying on familial resemblance in groups of family members such as MZ and DZ twins. Like twin design, GCTA uses genetic similarity to predict phenotypic similarity. However, GCTA uses genetic similarity for each pair of unrelated individuals based on that pair's overall similarity across hundreds of thousands of single nucleotide polymorphisms

(SNPs) for thousands of individuals; each pair's genetic similarity is then used to predict their phenotypic similarity. In contrast to the twin design, which only requires a few hundred pairs of twins to estimate moderate heritability, GCTA requires samples of thousands of individuals because the method attempts to extract a small signal of genetic similarity from the noise of hundreds of thousands of SNPs (Plomin and Deary, 2015). The advantage of this method over twin study method is that, in addition of being more robust to violations of the twin study assumptions, SNP data can be easier to collect since it does not require rare twins and also heritability for rare traits can be estimated.

2.4.3 Environment

It has been established that environmental influences contribute towards the phenotype. The twin design enables to estimate how much environmental influences contribute to individual differences alongside genetic influences. Environmental influences are defined as being of two types, shared (or between-family) and non-shared (or unique and within-family).

Shared environmental influences –Shared, or between-family, environmental influences are those which are shared amongst family members and serve to make members of a family (in this case, twins) similar to each other and different from members of other families. Shared environmental influences often tend to include family structure, socioeconomic status, and parental education to name a few (Plomin and DeFries, 1980). Research shows that there is little evidence of shared environmental influences on many commonly studied behaviours such as personality and cognitive abilities (Plomin et al., 2013). The modest shared environmental influences that have been found are only often significant early on during an individual's lifetime and gradually become less important for explaining similarity between family members (or twins) in long run (Burt, 2009).

Non-shared environmental influences - non-shared, or within-family, environmental influences are factors that are not common amongst family members, serving to make individuals different from one another. These environmental influences often do not operate on a family-by-family basis but rather on an individual-by-individual basis and includes measurement error. Examples include peer groups, perinatal traumas, and parental treatment (Plomin et al., 2013; Plomin and DeFries, 1980). Also research has shown that

in the long run, environmental variance in behaviour is largely non-shared (Plomin et al., 2013).

Environmental variation – the nature of environmental effects is diverse and unsettled compared to the underlying nature of genetic influences. Therefore it is hard to find the mechanism by which environment might influence a trait. In BG there are two main ways to investigate environmental influences on behavioural trait. The first method involves the use of family-based studies like twin design which allows environmental influences to be segregated into shared (influences which make twins more similar to each-other) and into non-shared (which make twins unique/different from one-another). The second way is to actually measure a particular aspect of environment like socio-economic-status (SES) of parents, stress, nutrition and likewise, and use it directly in genetic analysis. In other words, some aspects of environment might affect the expression of certain genetic influences. For example, considering that stress brings out genetic vulnerabilities towards depression, ergo depression might be anticipated to show more genetic influence for individuals experiencing stress (Plomin et al., 2013).

2.4.4 Genetic and Environmental Influences

One possible statistical model is that genetic and environmental influences act independently and additively to shape up given behavioural trait. However, this is not the case and behaviour does not follow simple (gene + environment = behaviour) rule. Also, in addition to simple additive/independent genetic and environmental influences, these two influences at times are correlated or they might interact non-additively.

Gene-Environment correlation - it refers to the experiences that are correlated with genetic propensities (Plomin et al., 2013). This can be explained as, what looks like an environmental effect can actually reflect genetic influence because these experiences are in-fact influenced by genetic differences between individuals (Plomin et al., 2013). As an example, friendly and extrovert parents not only pass these genes to their children but might also provide them with an environment that promotes development of friendly and extrovert nature in their offspring. This is also known as passive gene-environment correlation. Additionally, an extrovert individual might actively seek out situations that serve to further enhance their sociable skills. This is known as active gene-environment correlation. Finally the third type of gene-environment correlation is called the evocative

or reactive type. This occurs when individuals on the basis of their genetic propensities evoke reactions from other people on the basis of their genetic propensities. For example extrovert children might be picked up at school and given special opportunities (participation in extracurricular activities) (Plomin et al., 2013). This generally means that the effect of environment on phenotype/behaviour depends on the genotype and/or conversely the effect of genotype on phenotype/behaviour depends on the environment (Plomin et al., 2013).

Gene-Environment interactions - In quantitative genetics, the term gene-environment interaction usually implies that the effect of the environment on the phenotype depends on genotype or equally, and similarly, the effect of genotype on the phenotype also depends on environment (Plomin et al., 2013). In other words, it refers to genetic sensitivity to environments. For example, using the twin method, researchers found that the effect of stressful life experiences on depression was greater for individuals at genetic risk for depression (Plomin et al., 2013; Kendler, 1996). Another similar example is about the effect of physical maltreatment on conduct problems. This effect was also greater for children with high genetic risk (Plomin et al., 2013; Jaffee et al., 2005).

Thus it has been established that variance in the phenotype (behaviour) stems from four possible factors: i) genes might affect phenotype independent of environment; ii) the environment can affect the phenotype independent of genetic effects; iii) genes and environment may interact to affect the phenotype beyond independent prediction of genes and environments; and finally iv) genes and environment might be correlated and thus mutually affect/shape phenotype (Plomin et al., 2013).

So far it has been discussed that both genetic and environmental influences affect the phenotype and are therefore relevant. The next important question is *how much* do genetics and environment contribute to the trait in question? In other words, the statistical significance or reliability of the effect needs to be ascertained as well. This is also known as the effect size – i.e. the extent to which individual differences for a particular trait in the population can be accounted for by genetic differences among individuals. This is a group statistic, which refers to the individual differences for a trait in the entire population and not to some individual member. It is possible to quantify genetic and environmental influences and the metric used for this is called heritability and environmentability

respectively. It is a very important concept within the field of BG and is discussed below in detail.

2.4.5 Heritability

The statistic that estimates the genetic effect size is called ***heritability***. The Heritability statistic is defined as the proportion of observed or phenotypic variance that can be explained by genetic variance. In simpler terms, heritability is the amount of population variability explained by genetic similarity (Plomin, 1990). There has been increasing acceptance that in humans, many high-level behaviours show marked heritability (Plomin et al., 2013), a finding that would have been surprising to many researchers in the latter part of the 20th Century.

There are two types of heritability – first is the broad sense heritability and it refers to all sources of genetic variance, irrespective of how the genes operate i.e. additive or dominant. The second type of heritability is called narrow sense heritability and it gives an indication of the extent to which a trait will 'breed-true' i.e. it takes into account only the additive genetic effects so that the effect of genetic variation at one locus does not depend on variation at other locus (Plomin et al., 2013).

The heritability of a trait is estimated from correlations between relatives. For simplicity, the assumption is made that the only influences acting on the trait are additive genetic and environmental, which in turn are shared and non-shared. It is also expected that correlation between full siblings (or DZ twins) will represent half the additive genetic variance, and due to equal environment assumption, all the shared environmental variance but none of the non-shared environmental variance. In case of twins, since DZ twins are half as genetically similar (on average) as MZ twins, the difference in the correlation between MZ and DZ twins shows about half the genetic influence on behaviour; doubling the difference in correlations between MZ and DZ twins gives an estimate of heritability. In other words, if MZ twins correlate 1.0 and DZ twins correlate 0.5, a heritability of 100% is implied thereby indicating that genetic differences among individuals completely account for their phenotypic differences (Plomin et al., 2013).

Thus, the narrow sense heritability can be computed as twice the difference between correlations observed for MZ and DZ twin pairs. Falconer's equations (Falconer and Mackay, 1995) describe this as:

$$h^2 = 2(r_{MZ} - r_{DZ})$$

where h^2 represents the additive genetic variance or the narrow sense heritability, r_{MZ} is the MZ correlation and r_{DZ} is the DZ correlation.

In quantitative genetics, environmental variance is the variance not explained by genetics. Shared environment is estimated as family resemblance not explained by genetics whereas non-shared environment is the remaining variance, i.e. variance due to neither genetic nor shared environmental influences. Based on this, the proportion of variance due to shared environmental influences can be estimated as the difference between MZ correlation and heritability.

$$c^2 = r_{MZ} - h^2$$

where, c^2 represents the proportion of variance due to shared environmental influences.

Finally, since MZ twins are genetically identical, they provide a direct test of non-shared environment. Since they are genetically identical and have been raised together, any differences within a pair of MZ twins can only be due to non-shared environmental factors (Plomin et al., 2013).

$$e^2 = 1 - r_{MZ}$$

where e^2 is the proportion of variance due to non-shared environmental influences.

The heritability estimate also includes the error of estimation, which is a function of effect size and sample size. The following example helps better understand the aforementioned equations. Suppose the observed correlation in MZ twins is 0.64 and between DZ twins is 0.44. Therefore, heritability or the variance attributed to additive genetic influences becomes $0.4\ (= 2 * (0.64 - 0.44))$, which implies that 40% of variation in the population phenotype is caused by additive genetic influences. The shared environment therefore accounts for 24% variance $(0.64 - 0.4 = 0.24)$ and finally, the variance caused due to non-shared environmental influences is 36% $(1 - 0.64 = 0.36)$. It is important to note that two vital assumptions have been made in order to estimate these genetic and

environmental influences. The first is that the genetic influences are assumed to be additive. Dominance and any other interactions such as epistasis are not considered. The second assumption is that the only difference between DZ and MZ twin pairs is genetic and they have the same shared environment (or the equal environment assumption). If these two assumptions are violated, the prominence of genetic effects will get overestimated or underestimated depending on direction of violation (Plomin et al., 2013).

Interpreting heritability – a noteworthy point is that the heritability refers to the genetic contribution to individual differences and not to the phenotype of single individual. A good example is height – the heritability of height is about 90%. This implies that most of the height differences among individuals in a given population are due to genetic differences among them. Another important point is that the heritability statistic describes contribution of genetic differences to observed differences among individuals in a given population at a given time. The genetic and environmental influences might be different for different populations or even for same population but at different times, and thus heritability estimate for these would differ. Another related issue is average difference between groups, such as between male-female, between ethnic groups wherein heritability is not relevant to informing the origin of differences in group means. Heritability only refers to genetic contribution to differences among individuals within a group. Finally, heritability does not imply genetic determinism. Simply because a trait displays high genetic influences does not imply that this cannot be changed (Plomin et al., 2013). Environmental intervention has been shown to able to change this at times, for instance, environmental interventions for genetically influenced disorders such as myopia (eye glasses) and diabetes (insulin supplements) often resolve these disorders (Brady et al., 2011).

2.4.6 Evolution and Selection

At the core of Charles Darwin's theory of evolution is natural selection (Darwin, 2009), a process that occurs over successive generations and is described as the differential reproduction of (individuals) genotypes. During the twentieth century, genetics was integrated with Darwin's mechanism, thereby allowing the evaluation of natural selection as the differential survival and reproduction of genotypes, corresponding to particular phenotypes. The process of natural selection requires four key components –

Variation – Individuals (members of population) should exhibit some variation in behaviour and appearance.

Inheritance – There must be some traits that are passed on from parent to offspring, i.e. they should be heritable. The other traits could be influenced more by environmental conditions.

Significant degree of population growth – Having more offspring than the available local resources leads to struggle for resources. This is what selection targets/requires most because it results in elimination of substantial number of individuals.

Differential survival and reproduction – Individuals having traits better suited for the struggle for local resources will contribute more offspring to the next generation.

From one generation to the next, the struggle for resources or existence favours individuals with some variations over others and thus changes the frequency of traits within the population. This process is natural selection. The traits that confer an advantage to individuals who have more offspring are known as adaptations (Darwin, 2009; Dawkins, 2006).

In order for natural selection to operate on a trait, the said trait must possess heritable variation and must lead to an advantage in the competition for resources. If one of these requirements does not happen, then that trait will not experience natural selection. Therefore it can be deduced that natural selection operates by comparative advantage, not an absolute standard of design. Selection acts on the range of variation or frequency of traits, and can take the form of stabilising, directional, or diversifying selection (Darwin, 2009; Plomin et al., 2013).

Stabilising selection - extreme varieties from both ends of the frequency distribution/range of variation of the trait are eliminated. The frequency distribution/range of variation appears precisely as it did in the generation before, e.g. birth weight of human babies.

Directional selection - individuals at one end of the range of variation/frequency distribution of chosen trait(s) do especially well, and thus this range of variation/distribution of the trait in the subsequent generation keeps shifting/skewing from where it was in the parental generation. This is the commonly understood mode of operation of natural selection.

Diversifying (disruptive) selection - both extremes of distribution/range of variation are preferred at the expense of intermediate varieties. This is uncommon, mostly triggered by some sort of environmental change/disruption and often leads to speciation.

Thus, it can be deduced that selection has a major role in evolutionary design models as it shapes the evolution-learning interactions and therefore moderating the various differential performance/behavioural trends that emerge from this interaction.

2.4.7 Generalist Genes, Pleiotropy and Polygenicity

One of the most important recent findings from quantitative behavioural genetic research such as twin studies, is that the same set of genes is largely responsible for genetic influence across these domains. These genes are known as the "generalist genes" to highlight their pervasive influence (Kovas and Plomin, 2006; Plomin et al., 2007). Research in the field of BG suggests that:

- Most genes found to be associated with a particular learning ability or disability (such as reading) are also associated with other learning abilities and disabilities (such as mathematics).
- In addition, most (but not all) of these generalist genes for learning abilities (such as reading and mathematics) are also associated with other cognitive abilities (such as memory and spatial).

The two key concepts that underlie the generalist genes hypothesis are – pleiotropy and polygenicity. Pleiotropy means that one gene might affect multiple traits and polygenicity implies that each trait might be affected/influenced by multiple genes, each of which have a small effect size (Kovas and Plomin, 2006; Plomin et al., 2007).

These three concepts provide very useful hints for an evolutionary design scenario wherein the evolutionary task is different from learning task(s). In such cases, having knowledge of these concepts allows identification of suitable gene(s), (varying within suited range of variation/frequency distribution) which enables a population to acquire multiple tasks more effectively i.e. a gene or intrinsic parameter helps in acquisition of more than one task (Pleiotropy) or accuracy on given task might be dependent on combination of multiple intrinsic parameters (Polygenicity).

2.5 BG as a framework for neuro-evolution

Inspired from the research efforts made in the fields of neuro-evolution (Section 2.3.1) and behavioural genetics (Section 2.4), in this thesis a novel framework for neuro-evolution is presented which is based on behavioural genetic principles. This work draws an analogy between genes and the intrinsic parameters of ANNs, and between the training dataset and unique weights for ANNs and the environment – shared and non-shared respectively. Therefore, the proposed approach combines concepts of Behavioural Genetics with the idea of a parametrically diverse populations of learning systems, used in the context of a hybrid genetic algorithm, where genes (representing intrinsic factors) and environment (expressed via training datasets and unique initial weights) interact throughout development to shape differences in individual classifier behaviours (performance). The approach uses a population of twins (ANNs with a quantified degree of similarity in their neuro-computational parameters) to disentangle these genetic and environmental influences on performance.

As we saw, within Behavioural Genetics, it is well known that the quality of environment can modulate the influence of genetic variation. Following the analogy, one can similarly observe that training datasets affect the influence of intrinsic parameters. Thus, for ANNs, a certain number of hidden units may be highly beneficial for a specific condition of the dataset (say, for the number of training examples available) but if these conditions were to change drastically (similar to concept shift or concept drift in machine learning), the same number of hidden units may no longer be optimal. Thus, the system's performance will alter. Therefore, the proposed approach enables the population of ANN twins to acquire not just the evolutionary task/trait but also provides them with a learning bias/predisposition. This learning bias makes these ANNs more capable of learning to solve new/different task(s)/trait(s) in case of concept drift or concept shift. It also helps in analysing the role of selection in evolution and learning interactions. Applying different types of selection operators for same tasks might tend to target different aspects of given task/trait and thus result in completely different behavioural/performance patterns.

Thus by combining BG principles with evolution and learning, the aim is to build a framework that can provide the population of ANNs with learning predisposition in a dynamic environment. Figure 2.6 depicts the three broad levels of this framework.

Level 1	Phenotypic Plasticity (encoded neuro computational parameters within range)	⟺	Developmental Process (via learning)
	Shaped by		
Level 2	Selection (based on reproductive fitness)	⟺	Evolutionary Mechanism
	Leads to		
Level 3	Genetic Assimilation (Learning predisposition)	⟺	Evolutionary Outcome

Figure 2.6: Different levels in neuro-evolutionary framework

The proposed neuro-evolutionary framework is described in Table 2.1. The various phases involved in this framework are discussed in detail in subsequent subsections.

1. Identify evolutionary and (if needed) learning task(s)
2. Simulate variations in genetic influences
3. Simulate variations in environmental influences
4. Generate initial population of ANN twins, $G(0)$ such that each individual is an ANN characterised by its own genetic and environmental influences. Set $i = 0$
5. REPEAT
   (a) *Train* each individual (ANN twin) using some local search mechanism
   (b) Evaluate *Fitness* of each individual ANN according to training performance result. Also calculate heritability to quantify the net effect of all intrinsic parameters on learning
   (c) *Select* parents from $G(i)$ based on their fitness on evolutionary task
   (d) Apply search operators to parents to produce offspring which form $G(i+1)$
6. UNTIL, termination criterion is met

Table 2.1: High level description of the proposed Neuro-evolution framework

2.5.1 Evolutionary and Learning Task(s)

One of the main aims of this framework is to be able to evolve individuals which can learn task(s) different from those for which they have been selected. This means that there should be a clear distinction between evolutionary and learning tasks. This is in line with the nature, wherein living beings are capable of learning tasks which are different from those for which they have been selected. Human beings, for example, acquire lots of different cognitive and learning abilities over time usually in sequential manner and they are able to stack this new knowledge over the already existing knowledge, for instance humans can learn to read without that being a target of evolution thus far (since reading is a recent cultural invention).

The learning processes in neural network, or connectionist modelling is often achieved via gradient-based adaptive methods. However, when a gradient-based local search method is applied to sequential learning it suffers from one major drawback, called catastrophic forgetting or catastrophic interference. This means that a network having been trained on a task, if later is retrained on a different task, the newly acquired information might completely destroy the previously acquired task knowledge (Ans and Rousset, 1997; Ans and Rousset, 2000).

However, this behaviour is psychologically implausible and is therefore not an acceptable model of learning and a number of research efforts have been made to reduce this retroactive interference (refer Ans and Rousset, 1997; Ans and Rousset, 2000). Completely resolving this problem is still a challenge due to the distributed nature of represented information, principally required within a network to achieve good generalisation, is seemingly incompatible with weak interference levels. In ANNs, knowledge acquired about various learned items share the same connection weights. So when a new set of patterns/items (belonging to a different task) are learned, these connections weights, which have been adjusted with respect to the first task, will need to be modified again. Doing this re-modification might completely abolish all knowledge related to the previous task, resulting in what is called the 'stability-plasticity dilemma' (Ans and Rousset, 1997; Ans and Rousset, 2000; Grossberg, 1987).

The concept of multi-task (Caruana, 1997) learning addresses a similar issue of learning more than one task. However, in this case the aim is to learn multiple tasks simultaneously by exploiting common features in their training signals. Although this method has been

hugely successful, it has a limitation – there needs to be prior knowledge of all tasks that need to be learned and the training is done in parallel. In real world applications, however there are scenarios wherein there is no prior knowledge of new task or the environment within which the system (or ANN model) is placed might change gradually or abruptly (concept drift and concept shift). In those cases as well, the model would become less accurate and unreliable.

Therefore there is a need for a framework that is capable of evolving a population based on a given task or trait and also able to learn new tasks which might be completely different from what it has been selected for. In the proposed framework, therefore the first step is to identify the evolutionary task and learning tasks which are different from one other in terms of degree of similarity between input-output patterns, the presence of structure and regularity in mappings and overall complexity.

2.5.2 Simulating variations in genetic influences

This phase can further be divided into three steps, each of which is explained below.

2.5.2.1 Encoding structural and learning parameters into genome

Artificial neural networks depend on a range of parameters that increase or decrease their ability to acquire a new task. In this section we will illustrate the operation of the method by providing an example that considers evolving two main aspects of ANNs – the architecture (e.g. number of hidden units), including node transfer function-(steepness/slope of logistic activation), and the learning rule's algorithmic parameter (initial learning rate). These are the formational parameters of ANNs which either increase or decrease their ability to acquire a new task. Hidden units affect how a network is built and thus relate to network's capacity to learn. The steepness of the activation function corresponds to the activation dynamics acting within each network. Research has shown that transfer function is an important part of ANN architecture and has significant impact on its performance (Yao, 1999). Modulation of activation function leads to steeper or shallower slopes in the threshold function. A shallow slope negates the opportunity of a processing unit to make large output changes in response to small changes in input; a steep slope ultimately leads to very sensitive but binary response characteristics subject to

entrenchment effects. Therefore, too shallow or too steep values of this parameter will hinder the learning process (Plagianakos et al., 2006; Thomas et al., 2009). Heuristic learning parameters, such as the learning rate, govern how the network adapts and hence provide a network with the ability to learn. It is worth mentioning that in the proposed approach network's weights are not encoded in genome to be evolved. Instead these are continuously modified during the lifetime via learning process in which genetically inherited information interacts with information coming from external environment (Nolfi and Floreano, 1999).

In order to constrain learning, these properties are encoded into a genome using standard binary representation. The genome or genotype is the measure of the base composition of an individual. In other words, it serves as a set of instructions about how to form an organism of a particular species or group. Encoding parameters in the genome allows the individuals in a population to have a different genotype, that is, different values of each of the free parameters but from within the same fixed range. It thus leads to variability in a population by giving each network a different ability/capacity to learn new tasks. The genotypes are constructed as concatenated binary strings of given lengths. For the encoding, binary representation is used (Whitley et al., 1990), whereby each gene has two variants or alleles, with 10 bits per parameter, split into two chromosomes. Multiple bits are used per parameter (so-called polygenic coding) to allow gradual increase in parameter value under the pressure of selection. Figure 2.7 gives an example of genome in line with the previous discussion.

Genome	**1 1 1 0 0 1 0 1 0 1**	**0 0 1 1 0 1 1 0 0 0**	**1 0 1 1 0 0 1 1 0 0**
	HU	*LR*	*Slope*

Figure 2.7: An example genome

2.5.2.2 Calibrate the range of variation in genome

In the next step, the range of variation of each of these parameters is calibrated to avoid the presence of genes in the population that produce networks with no learning ability. This range is chosen to help acquire the evolutionary task. To this end, we begin with random

values for all parameters and train 100 neural networks for 1000 epochs while varying the values, in steps of 5 for hidden units and 0.01 otherwise, for each of these parameters individually. The calibration process is carried out for all parameters, until values are identified beyond which the learning fails, as well as the values which result in successful learning. This method provides a range of parameter values from poor up to very good performance. These values are then encoded in the artificial genome. Encoding the parameters within a fixed range allows variation in the genome between members of population, which then produces variations in computational properties. The range of variation of the parameter values serves as the upper and the lower bound used for converting the genotype (encoded values) into its corresponding phenotype (real values). For the encoding, binary representation is used, whereby each gene has two variants or alleles, with 10 bits per parameter, split into two chromosomes. The parameters and their range of variation are given in Table 2.2.

Neuro-computational Parameters	Type of parameter	Range of Variation
No. of hidden units	Structural	10 - 500
Initial learning rate	Learning	0.07 – 0.1
Slope of logistic activation	Node transfer	0.0625 – 4.0

Table 2.2: neuro-computational parameters and their range of variation

2.5.2.3 Genotype – Phenotype Mappings

The final step during this phase involves decoding the binary representation of the population into vectors of real values. The genotypes are the concatenated binary strings of given length and are decoded into real valued phenotypes over a specified interval using standard binary coding (Whitley et al., 1990). There are number of ways in which binary to real conversions can be done, in this work we make use of Matlab genetic Algorithm Toolbox (http://codem.group.shef.ac.uk/index.php/ga-toolbox) library function called $bs2rv$ which has a decoding matrix. This matrix has the following parameters to

accomplish this binary to real conversion – length of each binary string (len); lower and upper bounds for each encoded gene (neuro-computational parameter) (lb and ub); type of encoding –binary or grey ($code$); type of scaling to be used for each string – arithmetic or logarithmic ($scale$); and finally whether or not to include the lower and/or upper bound in the representation range ($lbin$ and $ubin$).

As an example consider a population consisting of three ANNs with their neuro-computational properties of number of hidden units (HU), learning rate (LR) and the slope of logistic activation ($Slope$) encoded in genotype (G), with each gene having four bits.

$$G = \begin{matrix} \boldsymbol{HU} & \boldsymbol{LR} & \boldsymbol{Slope} \end{matrix} \begin{bmatrix} 0\,0\,1\,1 & 0\,1\,1\,1 & 1\,1\,0\,0 \\ 1\,0\,0\,0 & 1\,0\,0\,1 & 0\,0\,0\,1 \\ 0\,0\,1\,0 & 1\,0\,0\,0 & 0\,0\,1\,1 \end{bmatrix}$$

Let the decoding matrix (known as $FieldD$) based on the range given in Table 2.2 be specified as follows –

$$\boldsymbol{fieldD} = \begin{matrix} \textit{Len} & \textit{Lb} & \textit{Ub} & \textit{Code} & \textit{Scale} & \textit{Lbin} & \textit{Ubin} \end{matrix} \begin{bmatrix} \mathbf{4} & \mathbf{4} & \mathbf{4} & \mathbf{10} & \mathbf{0.07} & \mathbf{0.0625} & \mathbf{500} & \mathbf{0.1} & \mathbf{4.0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} & \mathbf{1} \end{bmatrix}$$

This decoding matrix specifies the length of each gene, the lower and upper bounds for the range, it uses binary coding (0 represents binary and 1 grey), arithmetic scaling (1 represents arithmetic and 0 logarithmic) and finally that the lower and upper bounds are included in representation range as depicted by ones (otherwise use zero to exclude them from range). Therefore, the resulting phenotype, $Phen$ will be a function of G and $fieldD$, i.e.

$$Phen = f(G, fieldD)$$

And the resulting three ANNs will have the following parameter values.

$\boldsymbol{Phen} =$	$\boldsymbol{HU}$	$\boldsymbol{LR}$	$\boldsymbol{Slope}$
	22	0.082	1.74
	81	0.086	0.08
	17	0.084	0.14

2.5.3 Simulating variations in environmental influences

Environmental influences are defined as being of two types, shared (or between-family) and non-shared (or unique and within-family). Shared, or between-family, environmental influences are those which are shared amongst family members and serve to make members of a family (in this case, twins) similar to each other and different from members of other families.

Broadly speaking, in terms of ANNs, environment can be anything that is not the network itself. Usually it is the context or the setting within which the network is placed. The task or the problem that is needed to be solved by these ANNs are perfect representation of this context or setting. Each task corresponds to a particular context, and this context is a representative of shared environment. This is because this context is identical for all individual ANNs in the population and variations can be introduced by means of filtering the training set for tasks. Similarly in order to learn the task or problem at hand, ANNs need good initial weights. These weights are unique for each network and are often modified throughout lifetime according to new information being acquired. Hence the connection weights of ANNs are used as representative of non-shared environment. Each of these environmental components have been discussed in their respective sub sections.

2.5.3.1 Simulating shared environmental influences

The effects of shared environmental influences are simulated via a filter applied to the training set. This filter alters the quality of information available to the learning system. One factor identified to correlate with variations in language and cognitive development in

children is SES, usually measured by parent income and education levels. Although this measure is a proxy for the potentially multiple causal pathways by which environmental variation influences development, one line of evidence supports the view that SES modulates levels of cognitive stimulation: individuals in lower SES families experience substantially less quality and quantity of information (Thomas et al., 2009; Thomas et al., 2013). When implemented as a filter, the result is the creation of a unique subsample of the training set for each simulated family (i.e. twin pair) based on their SES.

An individual's environmental quality is modelled by a number selected at random from the range 0.6-1.0. This gives a probability that any given pattern in the full training set would be included in that individual's training set. This filter is applied at each generation to create unique training subsets for all members of the population in that generation. The range 0.6-1.0 defines the range of variation of environmental quality, and ensures that all individuals are exposed to more than half of the training dataset i.e. had a decent view of the problem domain. Due to the *equal environment assumption*, twin pairs have the same training subset.

2.5.3.2 Simulating non-shared environmental influences

The variance in performance that cannot be inferred from shared environment is representative of effects of unique or non-shared environmental influences. It includes any measurement error, as well as stochastic factors such as the initial weights of ANNs.

The learning speed and fast convergence of many feed forward neural networks depend to some extent on their initial values of weights and biases (Thimm and Fiesler, 1995; Yam and Chow, 2000). For this reason, in this approach, initial values of weights are used as a way to capture unique environments. The initialisation method used in this work is similar to that proposed by (Bottou, 1988) and uses the interval:$[-\frac{a}{\sqrt{din}}, +\frac{a}{\sqrt{din}}]$; wherein a is chosen in a way that weight variance corresponds to the points of maximum curvature of activation function. This value is 2.38 for standard sigmoid function (Thimm and Fiesler, 1995); and d_{in} is fan-in of neuron or the total number of inputs of a neuron in the network.

2.5.4 Generating populations of ANNs

The next step involves breeding the population of ANN twins using the genome. In this phase the biological processes of meiosis and fertilisation are simulated to create 50 pairs of MZ and 50 pairs of DZ twins. This method is chosen because it is the closest simulation of actual biological processes (refer to (Cooper and Hausman, 2000) for details about biological meiosis and fertilisation). Table 2.3 lists the steps involved in this breeding mechanism which are then explained with the help of an example.

1. Generate initial population $G(0)$ of n members at random
2. Split the population members into two groups of size $\frac{n}{2}$ representing fathers and mothers
3. REPEAT
   (a) For each parent, split genome into two equal halves resulting in two chromosomes per individual such that each chromosome carries half the information for each encoded parameter
   (b) Apply crossover m times on each chromosome pair, every crossover resulting in either two sperms or two eggs
   (c) Combine the sperms and eggs using positional recombination such that half of the encoded genetic information comes from sperm and other half from egg resulting $2m$ possible offspring
   (d) Verify the genetic similarity between twin pairs and accordingly choose MZ and DZ twins pairs, picking only one offspring per crossover
4. UNTIL population of desired size n is obtained

Table 2.3: Meiosis and fertilisation based method for creating population of ANN twins

To better understand each step of this method, consider the following example with population of two members.

$$Random_{pop} = \begin{matrix} 0\,1\,1\,1\,0\,1\,0\,0\,0\,1\,1\,0\,1\,1 \\ 0\,0\,0\,0\,1\,1\,1\,1\,0\,1\,0\,0\,0\,1 \end{matrix}$$

Splitting this population into a pair (of parents) we get: $P1 = 0\,1\,1\,1\,0\,1\,0\,0\,0\,1\,1\,0\,1\,1$ and $P2 = 0\,0\,0\,0\,1\,1\,1\,1\,0\,1\,0\,0\,0\,1$

Next each parent's genome is split into two chromosomes such that each chromosome contains half the information to code for each parameter (or gene).

$P1 \rightarrow \begin{matrix} 0\,1\,1\,1\,0\,1\,0 \\ 0\,0\,1\,1\,0\,1\,1 \end{matrix}$ $P2 \rightarrow \begin{matrix} 0\,0\,0\,0\,1\,1\,1 \\ 1\,0\,1\,0\,0\,0\,1 \end{matrix}$

Assume $P1$ is father and generates sperm. These have a single chromosome, created by crossover operation over father's two copies. This occurs independently in each parameter encoding region. We can create as many of these as needed. Example,

$Sperm1 \rightarrow 0\,1\,1\,1\,0\,1\,0$

$Sperm2 \rightarrow 0\,0\,1\,1\,0\,1\,0$

$Sperm3 \rightarrow 0\,0\,1\,1\,0\,1\,1$

$Sperm4 \rightarrow 0\,0\,1\,1\,0\,1\,1$

…..

Similarly, assume $P2$ is the mother and generates eggs. These also have a single chromosome, created by a crossover operation from mother's two copies. We can also create as many of these as needed. Example,

$Egg1 \rightarrow 0\,0\,0\,0\,1\,0\,1$

$Egg2 \rightarrow 1\,0\,0\,0\,1\,1\,1$

$Egg3 \rightarrow 1\,0\,1\,0\,0\,1\,1$

$Egg4 \rightarrow 1\,0\,1\,0\,1\,1\,1$

….

To create an offspring, positional recombination is used to combine the sperms and eggs, such that for each parameter, half the encoded information came from sperm and other half from egg. Thus, every crossover and fertilisation will lead to 2 offspring and resulting in total *2m* possible offspring. Although in biology, meiosis creates two sperm/two eggs from the crossover operation, the likelihood of both of the pair ending up in organisms is very small. If this happened, the mean genetic similarity of the population would start to be affected. We therefore only select one of the pair of sperm/eggs generated by the crossover to generate offspring, while the other is discarded. Thus, the remaining offspring will be –

$$Offspring1 \rightarrow Sperm1 + Egg1 \rightarrow 0\,1\,1\,1\,0\,1\,0\,0\,0\,0\,0\,1\,0\,1$$

$$Offspring2 \rightarrow Sperm2 + Egg2 \rightarrow 0\,0\,1\,1\,0\,1\,0\,1\,0\,0\,0\,1\,1\,1$$

$$Offspring3 \rightarrow Sperm3 + Egg3 \rightarrow 0\,0\,1\,1\,0\,1\,1\,1\,0\,1\,0\,0\,1\,1$$

$$Offspring4 \rightarrow Sperm4 + Egg4 \rightarrow 0\,0\,1\,1\,0\,1\,1\,1\,0\,1\,0\,1\,1\,1$$

….

To verify the genetic similarity between twin pairs, we use the Hamming distance metric to assess the similarity amongst offspring. First, we randomly pick any one offspring out of the possible four; let us assume that is $Offspring1$. Next, the similarity of $Offspring1$ is checked with the remaining three offspring using the Hamming distance formula. The offspring that is at most fifty percent similar is chosen as $Offspring1$'s corresponding DZ twin, assume $Offspring4$. This implies that $(Offspring1, Offspring4)$ form a pair of DZ twins.

$$DZ1 \rightarrow 0\,1\,1\,1\,0\,1\,0\,0\,0\,0\,0\,1\,0\,1$$

$$DZ2 \rightarrow 0\,0\,1\,1\,0\,1\,1\,1\,0\,1\,0\,1\,1\,1$$

Now, out of the remaining two twins, any one is chosen randomly and replicated, and they comprise the MZ twin pair. For instance, consider $Offspring3$ and therefore MZ twin pair becomes,

$$MZ1 \rightarrow 0\,0\,1\,1\,0\,1\,1\,1\,0\,1\,0\,0\,1\,1$$

$$MZ2 \rightarrow 0\,0\,1\,1\,0\,1\,1\,1\,0\,1\,0\,0\,1\,1$$

The remaining twin(s) are discarded. The genotypes of these resulting offspring were converted to a phenotype using the parameters values given in Table 2.2.

This process is repeated until the desired population size is achieved. When simulating multiple generations, the internal similarity of the gene pool should not be increased by inbreeding. In other words, if related individuals were to breed with each other, the average similarity between individuals would increase over the generations. For this reason, we separate twin pairs into breeding and non-breeding populations, and only breed from the breeding twin of each pair, while the non-breeding twin is available to compute heritability. Breeding therefore always takes place between unrelated individuals, preserving the mean genetic similarity within populations across generations.

2.5.5 Training and performance assessment

The population of twin ANNs is then trained independently on the evolutionary and (if needed) learning task(s) using any local search algorithm. The training is done using the filtered training sets (which represent shared environmental influences) and unique initial weights (representing non-shared environmental influences). In this work, the Rprop algorithm (Riedmiller and Braun, 1993) was used for training. The performance is assessed on the full training set, as well as on another novel dataset that was created to test the generalisation ability of the networks (refer Chapter 5 Section 5.3 for more details on datasets). The continuous outputs produced by networks are converted to binary by applying a threshold. Finally the performance is assessed using recognition accuracy based on Hamming distance at the end of training, as explained in Chapter 5 Section 5.4.1.

2.5.5.1 Fitness Evaluation

The next step involves evaluating each individual in the population to find their fitness. In this work, classification-based fitness measure was used. The network's fitness is computed as a proportion of its classification accuracy with respect to the cumulative accuracy of the population. The algorithm in Table 2.4 describes the fitness evaluation process.

Input:	Classification performance ($\boldsymbol{CP_i}$), for each network $\boldsymbol{i}$ [i.e. total no. of correct classifications]
Output:	Fitness (F_i), for each network i
Variables:	$N \rightarrow$ total number of networks in population $PT \rightarrow$ total number of patterns in dataset $P_A \rightarrow$ Accuracy of population

1. **initialise** $\boldsymbol{P_A} = 0$
2. $\boldsymbol{for\ i = 1{:}N\ do}$
   a. $\boldsymbol{A_i = \frac{CP_i}{PT}}$;
   b. $\boldsymbol{P_A = P_A + A_i}$;
3. $\boldsymbol{end}$
4. $\boldsymbol{for\ i = 1{:}N\ do}$
   a. $\boldsymbol{F_i = \frac{A_i}{P_A}}$;
5. $\boldsymbol{end}$

Table 2.4: Fitness evaluation method

2.5.5.2 Computing Heritability

Measuring heritability involves calculating MZ and DZ correlations. This is done by using the Pearson correlation formula as explained in (Plomin et al., 2013).

$$\rho_{x,y} = \frac{cov(x,y)}{\sigma_x \sigma_y}$$

Where, $\rho_{x,y}$ is the population correlation coefficient; $cov(x, y)$ is the covariance between x and y; σ_x is the standard deviation in x; σ_y is the standard deviation in y.

Subsequently, Falconers equations (Falconer and Mackay, 1995) are used to compute heritability and the proportion of variance due to shared and non-shared environmental influences as explained in Section 2.4.5 of this chapter.

2.5.6 Selection

Based on the performance of the networks on the full training set, members are selected from the breeding population to produce offspring to populate the next generation. In this work, two different selection mechanisms – the standard roulette wheel and truncation selection were evaluated, each inspired from natural selection methods. The selection is applied at the end of ANN training. An important aspect of this approach is the combination of the selection method(s) with the sexual reproduction method. The selected members enter the breeding pool and then breed with a randomly chosen member from that pool. After selection, only the offspring form the next generation of populations – parents (or members of previous/breeding population) are discarded. Despite the use of sexual reproduction, we did not include gender effects in the method or its outcomes. The selection metrics used in this work are described below in their respective subsections.

2.5.6.1 Roulette wheel selection (Stochastic selection)

The first selection method used, the roulette wheel (Lipowski and Lipowska, 2012) is similar to stabilising selection (Darwin, 2009). The basic idea of this selection process is to stochastically select from one generation to breed the members of the next generation. According to this selection mechanism, the fittest individuals have a greater chance of survival than weaker ones. Weaker individuals are however not without a chance.

In this method, the fitness function assigns a fitness to all population members. This fitness is used to associate a probability of getting selected with each individual. Let f_i be the fitness of i^{th} individual, then the probability for this individual to get selected is $P_i = \frac{f_i}{\sum_{j=1}^{N} f_j}$, where N is the total number of individuals in the population. The following

example explains this better. Assume a population consisting of 5 individuals. Table 2.5 and Figure 2.8 depict their individual fitness and associated probability of being chosen, calculated using aforementioned formulae.

Individual	Fitness (f_i)	Probability of selection (P_i)
Ind. 1	15	14%
Ind. 2	27	24%
Ind. 3	6	5%
Ind. 4	52	47%
Ind. 5	11	10%
	$\sum f_i$ = 111	

Table 2.5: Roulette wheel example

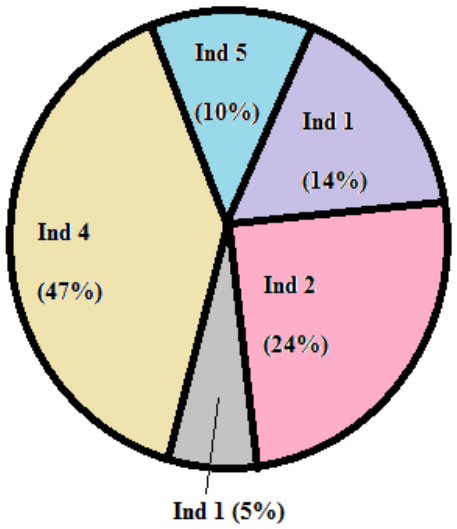


Figure 2.8: Roulette wheel example

The number of times this wheel is rotated (or a probability is generated at random) is equal to the number of individuals in the population. As can be seen from the way the wheel is divided, each time the wheel stops it gives the fitter individuals the greatest chance of being selected for the next generation and subsequent breeding pool. But the wheel can stop anywhere, so assuming that randomly generated probability (or point where wheel stops) is 21 this implies that individual 2 (with closest probability of 24%) will get chosen for becoming part of breeding pool even though it is not the fittest.

The benefit for using such a stochastic selection mechanism is that it helps to maintain genetic variability in population for longer time.

2.5.6.2 Truncation selection (Deterministic selection)

In this selection mechanism, only the fittest individuals get a chance to reproduce. This method is very similar to directional selection, wherein individuals at one end of the range of variation/frequency distribution of chosen trait(s) do especially well, and thus this range

of variation/distribution of the trait in the subsequent generation keeps shifting/skewing from where it was in the parental generation (Darwin, 2009). This is the commonly understood mode of operation of natural selection. Truncation selection is known to be the most efficient form of directional selection.

In this method, the individuals are ordered by fitness and only the fittest $X\%$ of the population are selected as parents for breeding. X can usually take any value from $50\% - 10\%$. Individuals below this fitness threshold do not get chosen for breeding at all. Considering the example in Table 2.5 of previous sub section, if the selection used was truncation instead of roulette wheel and $X = 40\%$, then the chosen members for breeding will be – individual 4 and individual 2 only, while the rest will be discarded. The main advantage of this method is rapid convergence, at the cost of possible local minima though.

2.5.6.3 Selection and sexual reproduction

In this work, the population(s) are generated using a biologically inspired sexual reproduction method (as explained in Section 2.5.4). As a result of sexual reproduction, the best properties of parents do not always get transferred to offspring. This is mainly because (i) an individual (parent) can only pass one copy of each gene (or intrinsic parameter) to its offspring. Therefore, there is an equal chance that either a maternally inherited gene or a paternally inherited gene will get transmitted to the offspring (Plomin et al., 2013; Goldberg, 2013; Sastry et al., 2014; Mehta et al., 2015). Since, after getting selected in breeding pool, the members breed randomly, the best properties do not always get transferred effectively, since the advantageous gene may not be inherited. (ii) Although some traits are inherited from parents during reproduction, these inherited traits are tendencies and offspring inherit the predisposition to exhibit that behaviour. Most traits, however, are the product of a developmental process involving interaction with the environment – usually skills and behaviours that are acquired by experience in the organism's lifetime and make it compatible with its environmental and survival needs. Environmental traits are not transferred genetically from one generation to another. It is the combination of inherited and environmental traits that make each individual unique (Griffiths, 2010).

2.5.7 Breed next generation and repeat

Finally, the selected parents become part of breeding pool and mate randomly to generate members of next generation. The procedure followed to generate offspring is the same as that explained in section 2.5.4, Table 2.3. The entire process was iterated until ANN parameters did not markedly change across generations or performance of the population at the end of training started to converge, i.e. the learning error reached a small value.

2.6 Summary and contribution of the chapter

In this chapter, a novel neuro-evolutionary approach which draws inspiration from behavioural genetics was presented. This approach uses artificial evolution techniques, i.e. genetic algorithms and learning techniques viz. artificial neural networks to study the interaction of learning and evolution with the intent of looking at the advantages, in terms of performance, that this interaction leads to. The main aim is to let populations of ANNs acquire a trait or learn a task they are being selected for and also learn new/different task(s) by themselves without expert intervention.

The combination of learning and evolution by means of hybrid algorithms has been very successful in the literature and has been applied in number of different areas like robot learning, automatic programming, game playing, operational research, and optimisation amongst others. These have also been used to study and enhance models of population genetics, economics, immune systems, and the interactions of evolution and learning and many more application areas. From an optimisation perspective, these hybrid approaches have fared much better both in terms of efficiency i.e. needing much fewer evaluations to find optima and more effective i.e. being able to find better or higher quality solution compared to more traditional approaches (Krasnogor and Smith, 2005). However, despite all these benefits, the process of designing effective and efficient neuro-evolutionary approaches is still fairly ad-hoc and is masked behind problem-specific particulars. Also most of the methods have been developed and tested in relation to work on/for only single task. However, during their lifetime individuals of any species acquire more than one behavioural trait, few of which are evolved, i.e. selected for and the most others are learned. For instance, in humans most high level behaviours are learned (i.e. their development is environmentally sensitive). Thus there are evolutionary selected behaviours such as social

status and then there are evolutionary novel (i.e. learned) behaviours such as reading, playing candy crush to name a few. There is still a need for a more generic and systematic neuro-evolutionary framework/approach which is not bound by problem/task specifics and is applicable and adaptable to various tasks belonging to any domain.

Given the observations collected from previous research efforts, the proposed framework enables a population of artificial neural networks to get fitter at a given evolutionary (or main) task over generations at a population level (i.e. the evolutionary task is same as the learning task); evolving populations are able to adapt to changes in the environment or the members of the population have to learn task(s) which are different from what they've been selected for, at an individual level.

This work draws an analogy between genes and the intrinsic parameters of ANNs, and between the training dataset and unique weights for ANNs and the environment – shared and non-shared, respectively. Therefore, the proposed approach combines concepts of Behavioural Genetics with the idea of a parametrically diverse populations of learning systems, used in the context of a hybrid genetic algorithm, where genes (representing intrinsic factors) and environment (expressed via training datasets and unique weights) interact throughout development to shape differences in individual classifier behaviours (performance). The approach uses a population of twins (ANNs with some degree of similarity in their neuro-computational parameters) to disentangle these genetic and environmental influences on performance.

The proposed approach is systematic and is not dependent on problem domain. It can be easily applied to any given set of learning and/or evolutionary tasks. The subsequent chapters will present application of this framework in varying scenarios like using evolution and learning for acquiring single trait at population level, acquiring multiple tasks and ensembles, respectively.

Chapter 3 Neuro-evolutionary framework for capturing population variability across language development: Modelling children's past tense formation

3.1 Overview

In this chapter, the BG inspired neuro-evolutionary approach presented in Chapter 2 is used to create computational models capable of capturing the population variability exhibited by 6 year old children in acquiring English past tense verbs. The work summarised in this chapter models the neuro-evolutionary scenario wherein the evolutionary task is same as the learning task. Literature in the field of behavioural genetics views variability in children's learning in terms of genetic and environmental influences. This approach uses a population of ANN twins to disentangle genetic and environmental influences on past tense performance and to capture the wide range of variability exhibited by children as they learn English past tenses. This chapter is organised as follows: first the existing literature in the field of language acquisition is discussed in Section 3.2. This is followed by a review of the fundamentals of computational modelling of the English past tense domain in Section 3.3. The English past tense task is then introduced in Section 3.4 and the experiment settings are described in Section 3.5. Subsequently we present results and analysis in Sections 3.6, 3.7 and 3.8 respectively. Finally, the summary and chapter contribution is given in Section 3.9.

3.2 An introduction to language acquisition

Language learning is considered one of the most complex tasks children face. Nevertheless, most children acquire it naturally, effortlessly, and quickly compared to other areas of cognitive development. Language is like the majority of complex systems which exist in nature and which empirically exhibit hierarchical structure (Simon, 1962).

Two opposing theories of language acquisition dominate the linguistic and psycholinguistic communities (refer to (Wintner, 2010) for a review). The nativist approach, proposed by Chomsky (see Chomsky, 1965; Chomsky, 1980), and promoted by Pinker, claims that the linguistic capability at least with respect to grammar is innate; therefore, certain linguistic universals are given to the language learners for free; only the established parameters need little tweaking in order for language to be fully acquired (Pinker, 1994).

The second view is the emergentist approach. It asserts that language emerges as a result of various challenging constraints, which are all consistent with other general cognitive abilities. No dedicated provisions for universal grammar are required. According to this view, the complexity of language emerges from the exposure of relatively simple developmental processes to a massive and complex environment (MacWhinney, 1998; MacWhinney, 2008).

Computational models provide an insight into language acquisition processes and the nativist versus emergentist debate. Artificial neural networks or connectionist networks offer an intuitive framework in which empirical phenomena in language acquisition can be explained by virtue of interactions between a language-learning system that incorporates general properties of computations in the brain and statistical properties of the linguistic environment to which it has been exposed (Karaminis et al., 2015). Computational models have been extensively applied to investigate the mechanisms of language development, including simulating early phonological development, lexical segmentation, vocabulary development, the acquisition of pronouns, the development of inflectional morphology, syntax comprehension, syntax production, metaphor comprehension, and reading (Thomas et al., 2013); (for reviews, see (Chater and Christiansen, 2008; Mareschal and Thomas, 2007).

One particular focus of research has been the field of inflectional morphology, which considers the alteration of the phonological forms of words to change their meaning (such as tense for verbs and plurals for nouns). Within this field, the acquisition of English past tense has drawn a great deal of attention, under the assumption that it taps the main cognitive processes involved in the acquisition and use of morphological knowledge (Karaminis et al., 2015). Children's acquisition of English past tense has been the focus of great deal of empirical research, mostly due to its *quasi-regular* mappings (Thomas et al.,

2013). Quasi-regular domains are interesting because of the presence of systematic input-output mappings and the presence of a minority of exceptions (Thomas et al., 2013).

The majority of English verbs, viz. regular, form their past tense by following a rule for stem suffixation, also referred to as +*ed* rule. This rule allows for three possible phonological suffixes, so called allomorphs (Karaminis and Thomas, 2010) - */d/* e.g. raise – raised; */t/* e.g. clap – clapped; */ed/* e.g. visit – visited. However, there are around 200 irregular verbs that form their past tenses by exceptions to the aforementioned rule, e.g. go – went; eat – ate; ring – rang, hit – hit. Although irregular verbs do not follow the productive rule, there are some irregular verbs that share characteristics of the regular verbs. For instance, many irregular verbs have regular endings, */d/* or */t/* but with either a reduction of the vowel, e.g. say – said; do – did, or the deletion of a stem consonant, e.g., has – had; make – made (Lupyan and McClelland, 2003). This overlap between regular and irregular verbs adds to the complexity of task domain. (See the mapping between written and spoken forms of English for another example of a quasi-regular domain within language, (Plaut et al., 1996)).

Due to this dual and fuzzy nature, there is an ongoing debate in the field of language development about the processing structures necessary to acquire the domain. (Refer to Thomas and McClelland, 2008, for a review). Is it necessary for the system to contain a prior processing assumption that the domain includes a productive rule, requiring symbolic computational structures? Or can productivity emerge from associative mechanisms exposure to quasi-regular domains?

There are two main theories. The first is a dual route account, proposed by Pinker (Pinker, 1984), according to which two separate mechanisms are involved in learning the mappings: a rule-based system for learning regular mappings, and a rote-memory system, which supports the irregular mappings. Rumelhart and McClelland (1986) challenged this dual-mode model by proposing a model based on the principles of parallel distributed processing. Their alternative model demonstrated that a two-layered feed-forward neural network can learn mappings between phonological representations of verbs and their corresponding past tense forms, both regular and irregular, as well as demonstrating productivity of the rule to novel verbs. This model, though extremely influential, had several drawbacks (refer to Karaminis and Thomas, 2010, for details).

This Backpropagation algorithm-based model inspired many subsequent connectionist models of acquisition of inflections like (Cottrell and Plunkett, 1991; Daugherty and Seidenberg, 1992; Plunkett and Marchman, 1991; Plunkett and Marchman, 1993) to name a few. Subsequent connectionist models addressed many of the drawbacks of the initial model. For example, Plunkett and Marchman (1993) took the main idea from Rumelhart model and modified it into a three-layered feed-forward architecture with more realistic phonological representations.

The line of research inspired by Rumelhart and McClelland employed artificial neural networks to simulate a wide range of past tense acquisition related phenomena. However, the majority of this work was concerned with capturing the developmental profile of the average child. Recently artificial neural network models have been extended to explore causal factors of atypical development, for example, in the cases of Specific Language Impairment and Williams syndrome (Karmiloff-Smith and Thomas, 2003; Thomas, 2005). To our knowledge, very little work has been concerned with capturing the wide range of variability that typically developing children exhibit in acquiring this aspect of language. Thomas, Forrester and Ronald (2013) modelled the effects of socio-economic status (SES) on language development, combining development and individual differences in a single framework. The key innovation of this model was that it addressed individual differences arising from variations in SES of the families in which children are raised, simulated as modulation of the structured learning environment, against a background of variation in the computational power of individuals' learning systems.

Recently, two innovations in this line of research have raised interesting questions of relevance to research in artificial life and evolutionary computation. The first innovation is the application of past tense modelling to individual differences between children with respect to their origin in *genetic and environmental factors*. For example, to some extent language delay runs in families, implying a heritable component, while differences in SES, a proxy measure of the quality of the environment, also explains some of the variance in language development (Thomas et al., 2013). The second innovation is the use of *multi-scale modelling* to reconcile data from multiple levels of description, including genetic, neural structure, cognitive processes, behaviour, and the environment, where behaviour itself is captured as the outcome of an extended development process involving interaction with a structured learning environment. This framework, using past tense as an illustrative

cognitive domain, has for example explored the relationship of statistical gene-behaviour associations (as reported in Genome Wide Association Studies) to developmental mechanisms. The specification of a genetic level in the model allows simulation of identical and fraternal twins, thereby simulating the kinds of twin study designs used to assess the heritability of high-level behaviour (Thomas et al., 2016).

In artificial life research, Genetic algorithms are usually employed for optimisation, where selection across generations aims to improve the performance of learning systems on a target task. By contrast, the existing multi-scale models took the presence of genetic variation as a starting point. This raises the following questions: where does the existing genetic variation in populations come from? How does this variation respond to the operation of selection? How do measures of heritability alter across generation through the operation of selection? What are the implications of using a quasi-regular domain as the target problem for optimisation? What parts of the problem domain are optimised across generations and what factors determine this?

To address these questions, in this work we used the neuro-evolutionary framework that combined concepts of Behavioural Genetics with the idea of parametrically diverse populations of learning systems, where genes (representing intrinsic factors) and environment (expressed via training datasets) interact throughout development to shape differences in individual classifier behaviours [presented in Chapter 2]. This framework has been applied in an evolutionary context by introducing selection in the populations' optimisation process across generations, focusing on learning a particular task: English past tense. The use of selection on performance in a quasi-regular task and the resulting findings make this English past tense acquisition model novel and different from others proposed in literature. In this context, a synergistic approach to capture population variability stemming from genetic and environmental influences and to analyse effects of selection on behavioural outcomes is presented in this chapter.

This approach not only captures the heterogeneity observed in acquiring a new ability but also helps in understanding how the quality of environment interacts with intrinsic constraints, leading to an individual's overt behaviour. It shows, for example, the different behaviours emerging due to interaction of quality of training set with good (or poor) learning rate (i.e., ability to learn, similar to neuroplasticity) and good (or poor) numbers

of hidden units (i.e., capacity to learn, somewhat similar to neurogenesis). It also highlights how applying selection results in changes in overt behaviour across generations.

3.3 Computational modelling of past tense acquisition

Computational modelling offers a method to explain theoretical proposals via implementation, to integrate empirical data with respect to common mechanisms, and to generate novel predictions. Its main drawback includes the simplification required for implementation. ANNs have been used extensively in the modeling of cognitive development (Thomas and McClelland, 2008; Thomas, 2016). Recently, these models have been used to investigate associations between levels of description, including those between genes, brain structure, brain activation, and behaviour (Thomas et al., 2016). Although the formalism of ANN employed in this work is much simplified and focuses on development within a single computational mechanism, it still offers numerous benefits. The model uses ANNs, which are computational abstractions of biological information processing systems; behaviour is acquired via an experience-dependent developmental process, which involves interaction between learning environment and genetic (or neuro-computational) properties; and the developmental trajectory and final representational states of each network are constrained by parameters with analogues in neurocomputation, such as the activation function of the neurons, the number of neurons and so on, as described in Chapter 2.

The aspects of the neuro-computational framework (discussed in Chapter2) that make it suitable for addressing the language acquisition problem are: First, the model simulates an aspect of cognitive development in populations of individuals, where variability in performance trajectories comes from intrinsic neurocomputational sources or extrinsic environmental sources (Thomas and Knowland, 2014). Second, the model includes an artificial genome that specifies the neurocomputational properties of the ANN. This allows modelling of genetic similarity between individuals, including creating identical and non-identical twin pairs. Twin study designs can then be simulated, which are the principal method to measure the heritability of individual differences. Third, the output of model can be viewed as acquired/learned behaviour, while changes in the range of intrinsic properties

of the ANNs, such as their connectivity or learning rate, can be viewed as potentially informative of mechanisms contributing to acquisition of said trait(s).

In this chapter we simulate acquisition of past tense verbs in 6 year olds using several populations, where individual differences stem from mixes of genetic and environmental variation. Our experiments use a base model taken from the field of language development, addressed to the domain of English past-tense formation. Here, the model is employed in an illustrative setting, intended only as an example of a developmental system applied to the problem of extracting the latent structure of a cognitive domain through exposure to a variable training environment. The intention is to capture qualitative characteristics of the empirical data rather than, for example, to exactly calibrate variances from genetic and environmental sources to fit empirically observed estimates of heritability in certain populations (Thomas, 2016).

With the aforementioned points in mind and based on the concepts and framework described in Chapter 2, we built a model to learn English past tenses and also to capture the individual differences in performance. The starting point of this work is to estimate the proportion of variance in performance attributed by variances in structural parameters (or genes), training set (shared environment) and initial weights (non-shared environment) and how selection subsequently alters these properties.

In Behavioural Genetics, factors affecting language development are attributed to genetic and environmental influences (Plomin et al., 2013). To model genetic influences, the variation in neurocomputational parameters of ANNs are encoded, thereby modulating their learning efficiency. These parameters relate to how a network (i.e. individual) is built (the number of hidden units), its processing dynamics (slope of logistic function within processing units), and how it adapts (learning rate), in line with the discussion presented in Chapter 2. The effects of shared environmental influences are simulated via a filter applied to the training set. This filter alters the quality of information available to the learning system. Learning systems raised in the same family, such as twins, experience the same training set. One factor identified to correlate with variations in language and cognitive development is Socio-Economic Status (SES), in terms of parent income and education levels. Although this measure is a proxy for the potentially multiple causal pathways by which environmental variation influences development, one line of evidence supports the

view that SES modulates levels of cognitive stimulation: children in lower SES families experience substantially less language input and also a narrower variety of words and sentence structure (Thomas et al., 2013). When implemented as a filter, the result is the creation of a unique subsample of the training set for each simulated family (i.e. twins) based on their SES.

The learning speed and fast convergence of many feed forward neural networks depend to some extent on their initial values of weights and biases (Thimm and Fiesler, 1995; Yam and Chow, 2000). For this reason, initial values of weights are used as one way to capture unique or non-shared environments. Apart from having genetic and environmental variation, the proposed model also incorporates "selection" and its effects.

3.4 Learning English past tense through Evolution

In the first instance, the neuro-evolutionary framework was applied in a scenario wherein the evolutionary task and learning task are the same. An interesting thing, however, is that despite the fact that ANN populations have to learn the same task they are being selected for, the chosen task (or behaviour) is an example of quasi-regular domain. This problem domain has dual nature – the majority of verbs form their past tense by following a rule for stem suffixation, also referred to as + ed rule. This rule allows for three possible suffixes - /d/ e.g. – tame – tamed; /t/ e.g. – bend – bent and /ed/ - e.g. – talk – talked. However, a significant number of verbs form their past tense by exceptions to that rule (example: go – went, hide - hid) (Plunkett and Marchman, 1991). The verbs adhering to the former rule-based approach are called regular verbs, while the verbs belonging to the second category are called irregular verbs. Also, some of the irregular verbs share the characteristics of the regular verbs. For instance, many irregular verbs have regular endings, /d/ or /t/ but with either a reduction of the vowel, example: say – said, do - did or a deletion of the stem consonant, example: has – had, make – made (Lupyan and McClelland, 2003). Thus, the networks have to learn the correct mappings between the English verb and its past tense. Given the phonological code of a verb stem presented in the input the networks have to learn to output the phonological code of its past tense form. This overlap between regular and irregular verbs is also a challenge for the model.

Additionally, as we discussed in Chapter 2, Section 2.2, the structure of all cognitive abilities that we possess like language acquisition, arises from the interaction between two complex adaptive systems – evolution and learning. The acquisition of such complex abilities begins with an initial genotype constructing an organism that displays some plasticity in its interaction with environment, thereby enabling it to learn. However, the degree of plasticity not only varies from one individual to another, it also shapes and/or constrains the achievable learning abilities. This in turn would either advance or constrain the quality of next generation population members because selective (or fitness-based) reproduction or evolution depends on acquired behaviour. Therefore, behaviours that are initially acquired through learning tend to become genetically specified later on and selection plays a key role in enabling this. Although, this follows only if, (i) selection can pursue structures sufficient to generate the target behaviour without learning and (ii) there is a fitness cost in needing to learn a behaviour (since there will be a period in which competence is not established).

Since the chosen task belongs to quasi-regular domain, the aforementioned points raised some interesting questions like - will selection operate similarly or differentially on regular and irregular aspects of the domain? If so, will those trends continue as long as populations are being evolved? Additionally, as described in Chapter 2, Section 2.4.6, do different kinds of selection mechanisms when applied to the same quasi-regular task, result in diverse overt behaviours? To address these questions, the neuroevolutionary framework was used to model the past tense acquisition task. As is shown later in the chapter, applying selection on performance on the English past tense problem leads to some novel findings, such as: (i) selection targets different aspects of a quasi-regular task depending on different initial conditions, potentially producing divergent populations. This in turn results in emergence of different and varied behavioural (performance) patterns, while still optimising on the said task; (ii) the amount of performance variation explained by genetic similarity, the so-called heritability metric (Plomin et al., 2013) plays an important role in identifying which aspect of this quasi-regular task is being targeted by selection.

Although the framework has been discussed in great detail in the previous chapter, Table 3.1 presents a high-level description of how the framework was applied for acquisition of English past tense task.

1. Simulate variations in genetic influences
    - Encode neurocomputational parameters into genome
    - Calibrate range of variation of each of these parameters
2. Simulate variations in environmental influences
    - Apply SES-based filter to dataset to generate unique training subset for each twin pair
3. Generate initial population of ANN twins, $G(0)$ such that each individual is an ANN characterised by its own genetic and environmental influences. Set $i = 0$
4. REPEAT
    (a) *Train* each individual (ANN twin) using some local search mechanism
    (b) Evaluate *Fitness* of each individual ANN according to training performance result for regular verbs, irregular verbs and combined performance. Also calculate heritability by comparing similarity of identical and fraternal twin pairs
    (c) *Select* parents from $G(i)$ based on their fitness on combined (overall) performance
    (d) Apply search operators to parents to produce offspring which form $G(i+1)$
5. UNTIL, termination criterion is met

Table 3.1: High level description of neuroevolutionary framework as applied to English past tense task

3.4.1 English past tense dataset

The dataset was based on the "phone" vocabulary from Plunkett and Marchman, (1991) past tense model. The past tense domain was modelled by an artificial language created to capture many of the important aspects of the English language, while retaining greater experimental control over the similarity structure of the domain (Plunkett and Marchman, 1991). Artificial verbs in effect were artificial monosyllabic phoneme strings that followed one of the three templates – CCV, VCC and CVC, wherein C is a consonant and V is a vowel. There were 508 verbs in the dataset. Each verb had three phonemes – initial, middle, and final. The phonemes were represented over 19 articulation binary features encoding English phonology e.g. voicing, tongue position, closed or open lips (Fromkin et al., 2013). A network thus had 3×19 = 57 input units

and 3×19 + 5 = 62 units at the output. The extra five units in the output layer were used for representing the affix for regular verbs in binary format.

In the training dataset, there were 410 regular and 98 irregular verbs. These were divided into four types: regular verbs that formed their past tense by adding */ed/* - e.g. visit – visited; regular verbs which formed their past tense by adding */d/* - e.g. tame – tamed, regular verbs which suffixed */t/* - e.g. clap – clapped, and finally the irregular verbs, which are of three types, vowel change e.g. hide – hid; no change e.g. hit-hit and arbitrary e.g. go – went. In the dataset, out of 410 regulars, there were 271 */ed/* verbs, 90 */d/* verbs, 49 */t/* verbs. As this is an imbalanced dataset, generating a classifier is challenging as the classifier tends to map/label every pattern with the majority class.

A second dataset was also created to assess the generalisation performance of the model. The main intent was to measure the degree to which an ANN could reproduce in the output layer properly inflected novel items presented in the input, according to the regular rule. The generalisation set comprised 508 novel verbs, each of which shared two phonemes with one of the regular verbs in the training set, for example *wug – wugged* (Karaminis and Thomas, 2010, Thomas et al., 2009b), i.e. generalisation set consists of novel regular verbs. Three different degrees of similarity were used to create generalisation dataset. In the first case, the first phoneme of the training set verb stem was changed. In the second case, the first two phonemes of verb stems were changed. Both of these changes were however consistent with phonotactics, i.e. a C was replaced by another C and a V by another V. In the third case, however, the first two phonemes were changed such that the conformity to phonotactics was violated. This use of novel verbs is standard practice for generalisation testing in context of tense formation (Karaminis and Thomas, 2010).

3.5 Experiment Design

In order to explore the behaviour of the model in different lineages, i.e. combinations of genetic and environmental influences, six replications of the model were tested, each

having a twenty-generation duration. The experiments were conducted on Condor, which is a platform that supports running high throughput computing on large collections of distributive owned computing resources (Thain et al., 2005). It follows a master-slave type configuration, which has proved suitable for training neural network architectures (Plagianakos et al., 2006).

Each scenario was characterised by its own initial population (produced with random binary genomes) and unique values for the other heuristics involved, such as initial weights. The evolutionary methodology was then applied to each of these model instantiations, such that they all shared the same range of variation for genetic and shared environmental influences. At the same time, however, they were unique, for each of them began with a different initial population created from random binary genomes. A major difference between replications 1 – 3 and 4 – 6 is that in the former case we used a stochastic selection metric (similar to stabilising selection sometimes occurring in nature, refer Chapter 2, Section 2.4.6), called the *roulette wheel selection,* whereas in the latter three replications a more deterministic selection measure (corresponding to more commonly occurring selection in nature – directional, as explained in Chapter 2, Section 2.4.6) – *truncation selection* was used. Using two very different selection mechanisms coupled with sexual reproduction and applied to a quasi-regular task lets us explore the effect of selection on the interactions between evolution and learning and on the performance trends emerging as a result. Thus, having six replications (*r1, r*2… *and r*6) of the model aided in evaluating the robustness of the method.

For each generation, there were 50 pairs of di-zygotic (DZ) and 50 pairs of mono-zygotic (MZ) twins with their computational parameters encoded into a genome. These were split in breeding and nonbreeding individuals, where the former is the population containing the 1st twin out of each of the twin pairs (100 networks) and the latter is the population containing the remaining 2nd twin of a twin pair (100 networks). These were instantiated as three-layered feed-forward networks and were trained using the batch version of the Rprop algorithm. The networks were trained on the filtered training sets, but performance was always assessed on the full training set and then tested on the previously unseen generalisation set. Performance was assessed using recognition accuracy based on Hamming distance (later in Section 3.5.1, Table 3.2). The filter applied was based on the

SES values of each twin pair. These values represent the probability of including a particular data point (or training pattern) of the full training set into an individual's filtered training set. This varied between 60% and 100% so that each individual would come across at least half of the training set. The range of $(0.6 - 1 = 0.4)$ represented a fixed level of environmental variation against which heritability, produced by neurocomputational parameter variation could be assessed. Twin pairs had the same filtered training set. The unique initial weights of ANNs were used to capture the effects of non-shared environmental influences. The weight initialisation method has been explained in Chapter 2 Section 2.5.3.2. In order to breed twins, different crossover operators were employed like single point, multi-point and more.

Moreover, empirical data from young children performing the past tense task (Karaminis and Thomas, 2010, Thomas et al., 2009b), were also used to benchmark the performance of the proposed model with respect to this age group, which has been the subject of considerable research in the literature.

3.5.1 How was behaviour (performance) measured?

The population of twin ANNs was trained on the filtered past tense dataset using the Rprop algorithm (Riedmiller and Braun, 1993). The performance was assessed on the full training set, as well as on another novel dataset that was created to test the generalisation ability of the networks (see Subsection 3.4.1). First, the continuous outputs produced by networks were converted to binary by applying a threshold. Then the performance was assessed using recognition accuracy based on Hamming distance as explained in Table 3.2. The behaviour or the performance was measured by accuracy on regular and irregular verbs combined. However, accuracy on regular and irregular verbs was also measured separately using the same algorithm.

Input: Actual output of network, $\boldsymbol{Y_n}$

Desired output, $\boldsymbol{Y_d}$

Output: Performance accuracy, A

Variables: $I \rightarrow$ total number of patterns in $\boldsymbol{Y_n}$

$J \rightarrow$ total number of patterns in $\boldsymbol{Y_d}$

$P_i \rightarrow$ a pattern in $\boldsymbol{Y_n}$, where $, i < I$

$P_j \rightarrow$ a pattern in $\boldsymbol{Y}_d$, where $, j < j$

$h_{dist} \rightarrow$ hamming distance between phonemes of P_i and P_j

$h_{allm} \rightarrow$ hamming distance between allomorphs (or the last 5 bits) of P_i and P_j

$Match; corr; err$

1. **initialise** $\boldsymbol{Match = false;\ corr = 0; err = 0}$
2. **for** $(\boldsymbol{i = 1: I; i < I; i++})$ **Repeat**
3. Split $\boldsymbol{P_i}$ into three phonemes and allomorph
4. **for** $(\boldsymbol{j = 1: J; j < J; j++})$ **do**
5. Split $\boldsymbol{P_j}$ into three phonemes and allomorph
6. Calculate $\boldsymbol{h_{dist}}$ between corresponding phonemes of $\boldsymbol{P_i}$ and $\boldsymbol{P_j}$
7. **If** $\boldsymbol{h_{dist} < preset\ threshold}$ (for all three phonemes) **do**
8. Calculate $\boldsymbol{h_{allm}}$ between respective allomorphs
9. **If** $\boldsymbol{h_{allm} == 0}$, **do**
10. $\boldsymbol{corr = corr + 1}$;
11. $\boldsymbol{Match = true}$;
12. Break;
13. **Else**
14. $\boldsymbol{err = err + 1}$;
15. **end**
16. **end**
17. **end**
18. **If** $(\boldsymbol{j == J\ AND\ Match == false})$ **do**
19. $\boldsymbol{err = err + 1}$;
20. **end**
21. **end**
22. $\boldsymbol{A = \left(\frac{corr}{I}\right) * 100}$;
23. **Return** $\boldsymbol{A}$

Table 3.2: Recognition accuracy based performance calculation algorithm

3.6 Roulette wheel selection based experiment results

Table 3.3 describes the experiment setting used in the first set of lineages using the roulette wheel selection operator. Three lineages, each having 20 generations duration, were tested under this setting.

Replications	R_1, R_2, R_3
No of Generations per replication	20
Size of population	Breeding = 100; Non-breeding= 100 Total R_1+ R_2+ R_3 across generations= 12,000 ANNs
Size of Datasets	Training= 508 Generalisation= 508
Training Mode	Batch
Max. training epochs	1000
Initial weight update (Rprop learning rate)	Values from genome
Hidden units. Steepness of logistic	Values from genome
Selection Operator	Roulette Wheel- applied at the end of training (1000 epochs)
Crossover	6 crossovers/chromosome; different operators used
Environmental Factor (SES)	Probability value between 60% and 100%

Table 3.3: Experimental Design for RWS based replications

3.6.1 Results and Analysis

The overall accuracy of the model on regular verbs was higher than that on irregular verbs. The mean performance on the full training set ranged between 74% and 80% for regular verbs, and between 34% and 40% for irregular verbs. The model was able to efficiently generalise the past tense rule in novel items with the mean accuracy rate of around 60%.

The performance of our model compares well with empirical data for children reported in the literature (Bishop, 2005; van der Lely and Ullman, 2001). The behavioural data in Bishop (2005) comprise of performance results of 442 6-year old children on a past tense elicitation test. They were tested on 11 regular verbs and 8 irregular verbs. The average accuracy achieved by children on regular verbs is centred around 90%, whereas the average accuracy for irregular verbs is centred on 38%. It also agrees to a large extent with the performance reported in the developmental study of van der Lely and Ullman, (2001), for 5-7 year old children: for regular verbs, accuracy rates were 60% (5 year olds), 75% (6 year

olds) and 80% (7 year olds); for irregular verbs, accuracy rates were 25% (5 year olds), 58% (6 year olds) and 50% (7 year olds).

We compare our model's performance with two other past tense models from (Thomas et al., 2009b) and (Karaminis and Thomas, 2010). In the former model, 1000 networks were trained for 1000 epochs in various degrees of environmental and genetic variation scenarios. The experimental setting that closely matched our experiments, referred to as G-wide and E-narrow, resulted in average accuracy of 80% for regular verbs and 38% for irregular verbs. In latter case, the model comprised of network trained for 400 epochs, with results averaged over 10 replications using different random seeds. The results corresponding to 6 year olds fall in the range of 60%-80% in case of regular verbs and between 20%-40% for irregular verbs, achieved in the window of 51-70 epochs. Their model also achieved over 80% generalisation accuracy.

We also analyse the results, initially using independent linear regressions to assess performance / heritability / parameter changes for each population over the generations. Individually reliable trend lines at the p=.05 level are described alongside results. Given the overall design, which combined repeated measures (e.g., regular verb performance, irregular verb performance, generalisation) and between group measures (replication population; breeding vs. non-breeding populations), trajectory analysis was used to assess overall patterns in the component linear regressions across the full experimental design (Thomas et al., 2009a). Trajectory analysis was used to assess overall patterns in the component linear regressions (Thomas et al., 2009a). This approach has been increasingly applied in psychology research, especially to the study of disorders (Annaz et al., 2008; Thomas et al., 2009a and references therein). The main focus of this approach is to construct a function linking behaviour (i.e. performance) with age (or stage (i.e. generation) in developmental scale) and then to evaluate whether this function fluctuates between various groups such as between twins (breeding vs non-breeding) or between replications and likewise. Due to the aforementioned feature, this method is well suited for the experiments reported in this chapter, even though trajectory analysis method isn't widely employed in traditional machine learning approaches/applications.

Figure 3.1 depicts the mean accuracy (calculated at the end of training) with which breeding and non-breeding twin populations formed past tenses for regular verbs across a sequence of generations, for three replications with differential initial genomes. Each of these graphs

summarises the results from 12,000 networks. Figure 3.2 shows equivalent data for irregular verbs, while Figure 3.3 represents the generalisation results. In each case, a zigzagged line indicates the mean accuracy level of the 100 networks for each population at each generation, while a straight line represents the general trend observed in that replication scenario. The trend line was derived from a linear regression line based on the least squares method, predicting mean performance level from generation number. R^2 values were relatively small, reflecting the non-monotonic changes in performance over generations. This is in line with changes in mean trait levels in animal populations following selective breeding, such as the open field behaviour of mice (DeFries et al., 1978; Plomin et al., 2008).

We initially considered performance of application of the past tense rule, comparing the measures of regular verb performance against generalisation, for the three replications and breeding versus non-breeding populations (12 trajectories). A fully factorial ANCOVA revealed no overall change in performance across the generations ($F(1,108)=2.23$, $p=.138$, $\eta_p^2=.020$). However, this masked a differential pattern between replications, with some showing rising performance and others no change ($F(2,108)=8.65$, $p<.001$, $\eta_p^2=.138$). This pattern was common across measures and breeding/non-breeding populations. Regular verb performance was reliably higher than generalisation ($F(1,108)=6288.30$, $p<.001$, $\eta_p^2=.983$).

Irregular verb performance, by contrast, showed no individual population with rising performance across generations, though the replication populations showed consistently different levels of accuracy ($F(2,108)=3.27$, $p=.042$, $\eta_p^2=.057$). Regular verb performance was also reliably higher than irregular verb performance ($F(1,108)=9958.42$, $p<.001$, $\eta_p^2=.989$). Comparison to regular verb performance indicated that the relationship between performance and generation was reliably modulated by measure ($F(2,108)=4.53$, $p=.013$, $\eta_p^2=.077$).

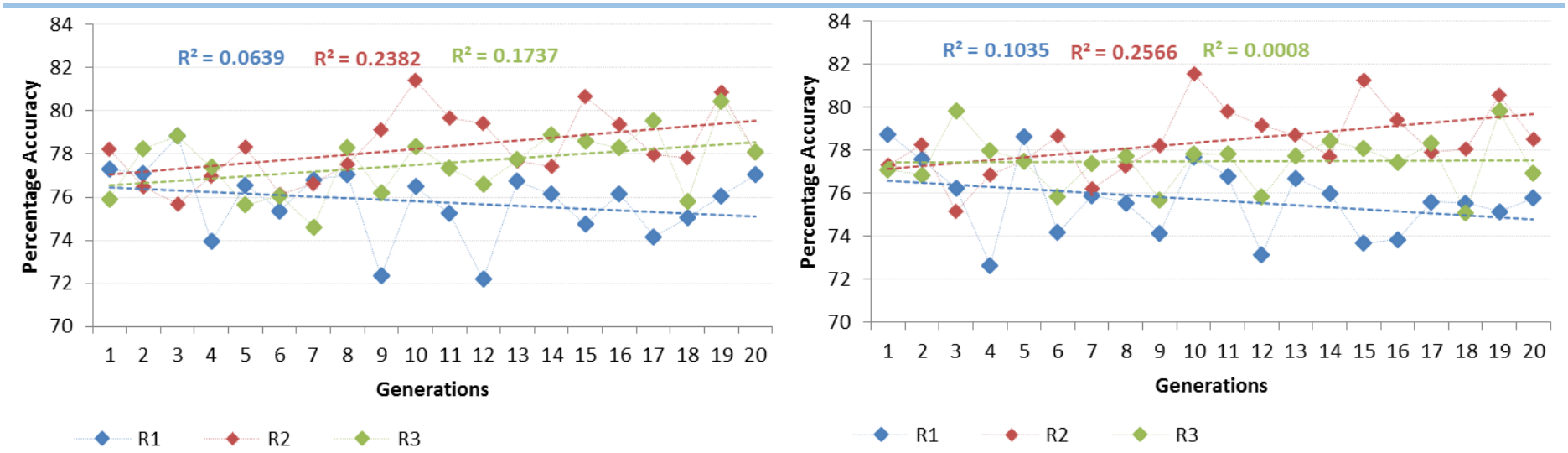


Figure 3.1: Mean performance per generation for breeding (left) and non-breeding (right) twin populations on regular verbs

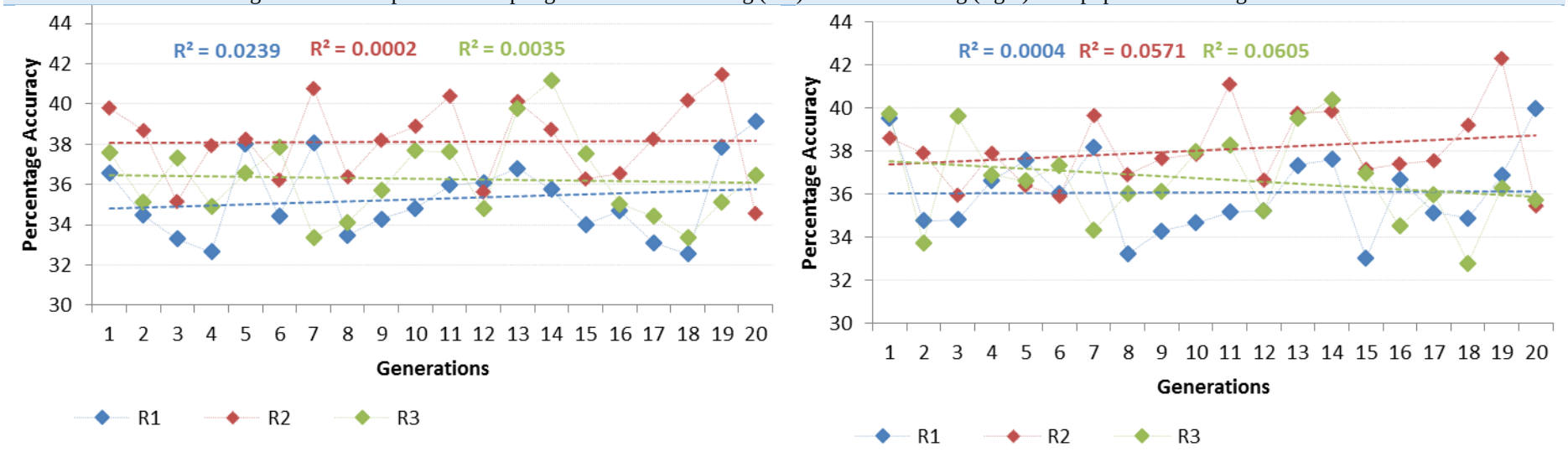


Figure 3.2: Mean performance per generation for breeding (left) and non-breeding (right) twin populations on irregular verbs

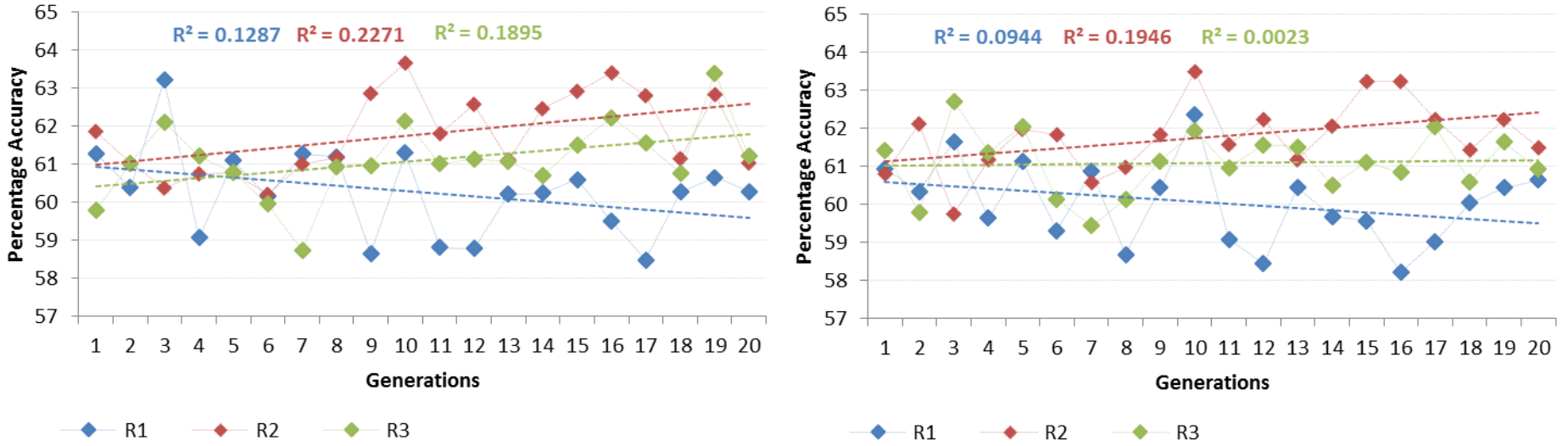


Figure 3.3: Mean generalisation accuracy per generation for breeding (left) and non-breeding (right) twin populations

Most notable in Figures 3.1 to 3.3 is the presence of some downward trends in performance over generations, despite the operation of selection. Selection should serve to improve performance over generations, since genes conveying an advantage in learning are more likely to be transmitted to the next generation. The probabilistic nature of this transmission – the mode of sexual reproduction does not guarantee that the advantageous genes of an individual selected to breed will appear in the offspring, and the selection mechanism is itself probabilistically related to final performance level – accounts for the slow change in population mean performance over generations. It does not account for why performance should become *worse* over generations on same measures.

The explanation is suggested by the fact that opposite trends are observed for regular verbs and irregulars (with generalisation patterning with regular verbs). When performance across generations is worsening for regular verbs, it is improving for irregular verbs, and vice versa. Because the learning domain of English past tense is quasi-regular, good performance across all mappings could in principle be achieved by scoring strongly on regular verbs, strongly on irregular verbs, or strongly on both (with regular verbs the more powerful driver, being in the majority). If optimising the same neuro-computational parameters enhanced both types of mapping, then selecting for either strong regular or strong irregular performance should enhance the performance of the population on the other mapping type as well. However, it is known that the two types of mappings are differentially sensitive to different parameters in ANNs, for example with regular mappings requiring steeper sigmoid functions and irregular mappings requiring more hidden units (Thomas et al., 2016). The combination of (a) selection by mean performance that could either be driven by stronger regular or irregular verb performance, and (b) parameters that favour learning of either regular or irregular mappings, together sets the stage for possible divergence of gene pools over generations. Even in the face of selection, some lineages may become specialised for regular verbs at the expense of irregular verbs, while other lineages may become specialised for irregular verbs at the expense of regular verbs. Yet others may show increased performance in both verb types across generations. Which path a given starting population follows will depend on the distribution of parameters created by the initial genomes, the set of individual environments, and stochastic factors involved in selection and breeding.

This phenomenon is similar to Waddington's epigenetic landscape, an idea proposed by Conrad Waddington (Ferrell, 2012, pp R459). In his model, Waddington associated the process of cellular differentiation to a marble, representing a pluripotent cell, on top of a hill. The hill contains many paths or valleys that the marble can roll down and each path will eventually lead to a distinct final differentiated state, such as a blood cell or a skin cell. He described each of the valleys as an individual developmental pathway or 'chreode'. As the marble moves down the hill the paths and final destinations available become more limited, representing the increased differentiation of the cell (Waddington, 1957). This is what makes an initial pluripotent cell to become a specialised cell, and reversing this process is impossible under normal circumstances.

Similarly, when selection is applied on a quasi-regular task, different aspects of the task may be optimised depending on the genetic propensities of initial populations. The trend then continues throughout the lineage because of genetic inheritance. Thus, if, as shown in lineage 1 (replication 1) in Figures 3.1 and 3.2, the first few generations improve their learning of irregular verbs at the expense of regular verb performance, the lineage is committed to this pathway. Genes for good learning of regular verbs have been lost from the gene pool. Evolution cannot go into reverse gear and find a pathway that combines good learning on both verb types. Replication 3 represents the opposite case of optimisation on regulars, while replication 2 shows improvement in both verb types across generations.

Changes in the frequency of different gene variants (here, binary values of 0 or 1) in the gene pool should alter the range of genetic variation across generations. Given that the range of environmental variation (SES of 0.6 to 1.0) remained consistent across generations, any changes in genetic variation should be reflected in changes in heritability. To explore this idea, we examined correlations in performance between MZ and DZ network twin pairs, using Falconer's equations to derive estimates of heritability (Plomin et al., 2008). Heritability was estimated as twice the difference between MZ and DZ correlations; unique environmental effects were estimated as the extent to which MZ correlations were less than 1; and shared environment effects were estimated as the remaining variance (i.e., 1-{heritability}-{unique environment}). Strictly speaking, these equations assume an additive model, which only holds for MZ correlations that are no more than twice DZ correlations. Although in our results the correlations sometimes violated this condition, we continue to plot heritability estimates according to the same formulae for

compatibility across conditions. Therefore, the plotted data should be seen as proportion to the heritability and environmentability observed in populations, rather than direct estimates under an additive model. Thus, the values sometimes range outside of the range 0 to 1, as the assumptions of the additive model become violated.

Figure 3.4 shows the estimates of heritability (variance due to genetic factors) for regular (4a) and irregular verbs (4b). These six trajectories were compared in a fully factorial ANCOVA. Heritability reliably reduced over generations ($F(1,54)=5.54$, $p=.022$, $\eta_p^2=.093$), and this pattern was not modulated by measure or replication population. Though replication 2 showed the steepest reduction in heritability, the difference in the pattern across replications was not reliable ($p=.107$).

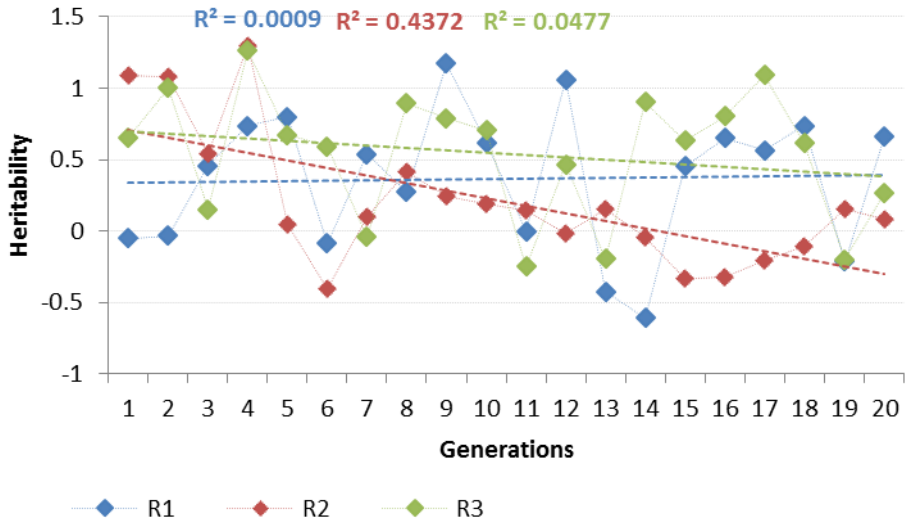


Figure 3.4(a): Heritability or proportion of variance due to genetic (or structural) factors for Regular Verbs.

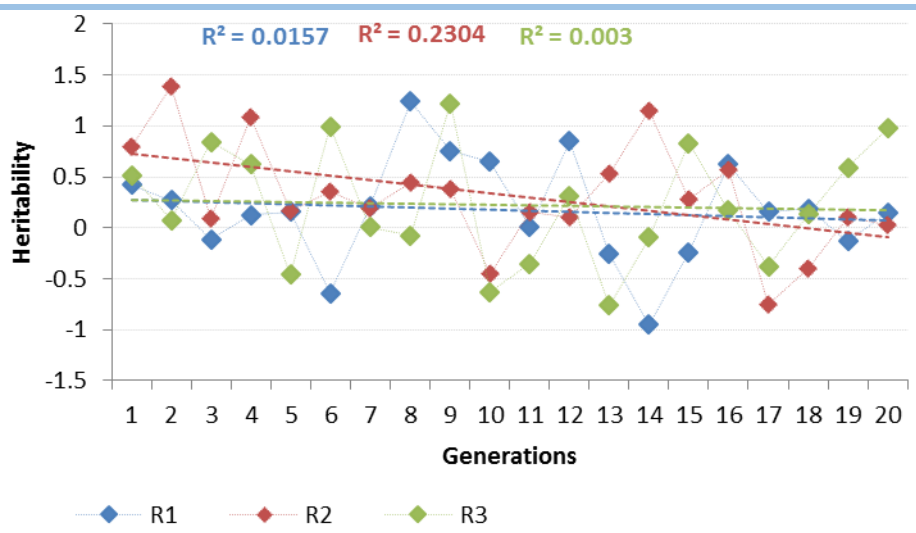


Figure 3.4(b): Heritability or proportion of variance due to genetic (or structural) factors for Irregular Verbs.

If a lineage becomes increasingly optimised on a task (or a specific aspect of the task domain), the range of its domain-relevant intrinsic parameters should decrease across generations, as only the genes producing the best parameter values are retained. For example, if populations are improving on irregular verbs, which require more capacity to hold non-systematic mappings, then across generations, networks with larger number of hidden units have a greater chance to get selected in the breeding pool. Across generations, the variability in the range of number of hidden units will reduce. By contrast, the range of variation in other less relevant parameters may be less affected. *Optimisation and heritability should therefore have an inverse relationship*.

In line with this expectation, in replication/lineage 1, regular verb performance and rule generalisation dropped across generations while irregular verb performance improved. Heritability for regular verbs was initially higher than that for irregular verbs, centred on 0.4 and it then increased across generations, implying lack of selection for parameter sets specialised for regularity. By contrast, heritability of irregular verbs was lower, centred on 0.2, and decreased with generations, implying selection for, and a narrowing of the range of, parameter sets specialised for irregularity. Note that this process of specialisation causes *overall* accuracy to drop, because irregular verbs form a minority of the dataset (there are only 98 irregular verbs compared to 410 regular verbs).

In replication/lineage 2, regular verb performance, irregular verb performance, and generalisation all increased across generations. Heritability of regular verbs dropped from high values of around 0.8 to around zero. A similar pattern was observed for irregular verbs, with heritability dropping from high values to almost nil. In this lineage, optimisation caused a narrowing of the range of genetic variation relevant to learning of both regular and irregular verbs.

In replication/lineage 3, regular verb performance and generalisation improved while irregular verb performance dropped. The heritability of regular verbs decreased from 0.6 to 0.2 while the heritability of irregulars remained stable, but at a lower value, centred on 0.2. These two observations suggest that the range of intrinsic parameters being targeted by selection works well for both regular and irregulars. But as generations progressed, there was a narrowing in this range for parameters more suited to regular verbs.

When heritability of a particular aspect of the task reduces, it implies that variance in performance is less due to genetic factors and more due to shared and non-shared environmental factors. Figures 3.5(a) and 3.5(b) display the variance due to shared environmental factors, in this case the filtered training datasets. The effect of shared environment reliably changed over generations ($F(1,54)=8.42$, $p=.005$, $\eta_p^2=.135$) though this was driven primarily by replication 2, illustrated by an interaction of population X generation ($F(2,54)=3.65$, $p=.033$, $\eta_p^2=.119$). The pattern was common across regular and irregular verbs.

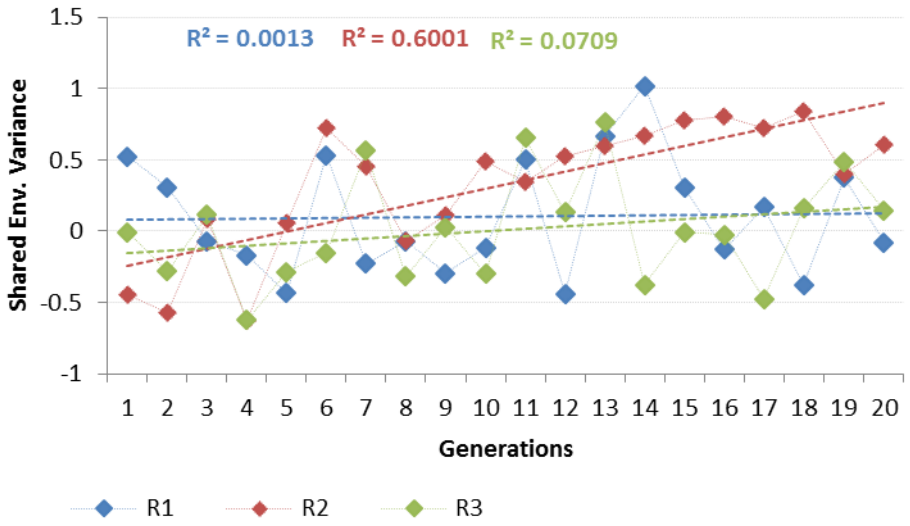


Figure 3.5(a): Proportion of variance due to shared environmental factors - Regular Verbs.

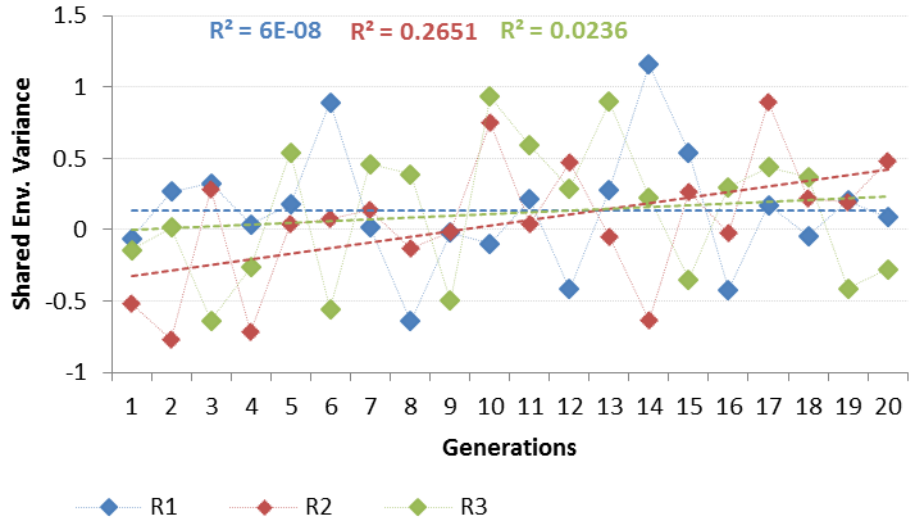


Figure 3.5(b): Proportion of variance due to shared environmental factors - Irregular Verbs.

Figures 3.6(a) and 3.6(b) represent the variance in performance due to non-shared environmental factors or initial weights in our implementation. Analyses revealed no reliable effects, with non-shared environmental effects consistent across generations and modulation neither by measure type nor by replication population. The figures show that the differences in initial weights led to large variability in behavioural outcomes. In cases when intrinsic factors were not very suitable to the task domain, having good initial weights

might give networks a fighting chance, i.e. training could be biased towards non-shared environmental factors to enhance behavioural performance.

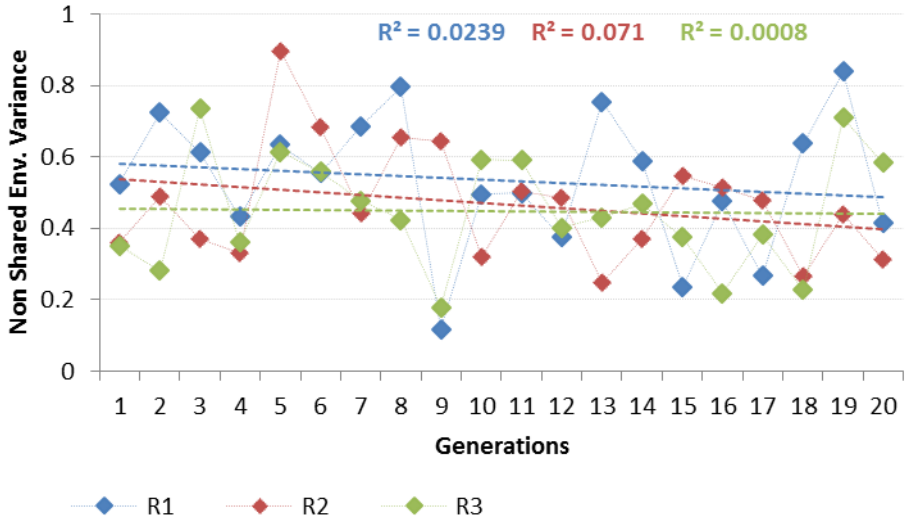


Figure 3.6(a): Proportion of variance due to Non-shared environmental factors - Regular Verbs.

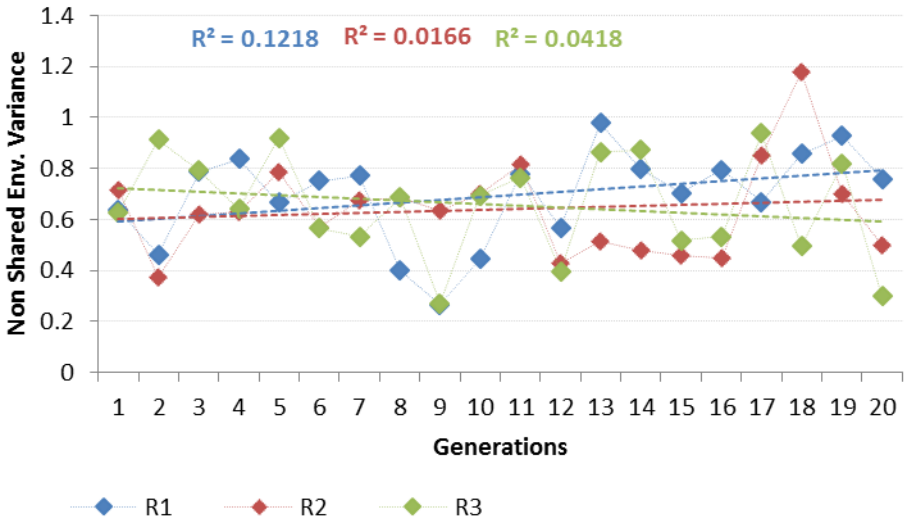


Figure 3.6(b): Proportion of variance due to Non-shared environmental factors - Irregular Verbs.

Heritability is a useful statistic because it is scalable across potentially very large numbers of computational parameters (and their interactions) that contribute to the variation in

learned high-level behaviours, or in this case, the outcome of learning for a set of ANNs. However, in the current simulations, relatively few parameters were encoded in the genome and permitted to vary across populations and between generations. Our final step of analysis, then, was to examine the change in mean parameter values for a given lineage across generations. This should reveal the domain-relevant parameters that were selected, in those cases where performance on one verb type was enhanced at the expense of the other, and therefore in turn reveal the drivers behind changes in heritability.

Figure 3.7 depicts changes in mean parameter values for number of hidden units, initial learning rate, and slope of the logistic activation function. For hidden units, there was a reliable reduction in number across generation ($F(1,54)=190.55$, $p<.001$, $\eta_p^2=.779$), with the reduction occurring at different rates across the three replication populations ($F(2,54)=33.79$, $p<.001$, $\eta_p^2=.556$).

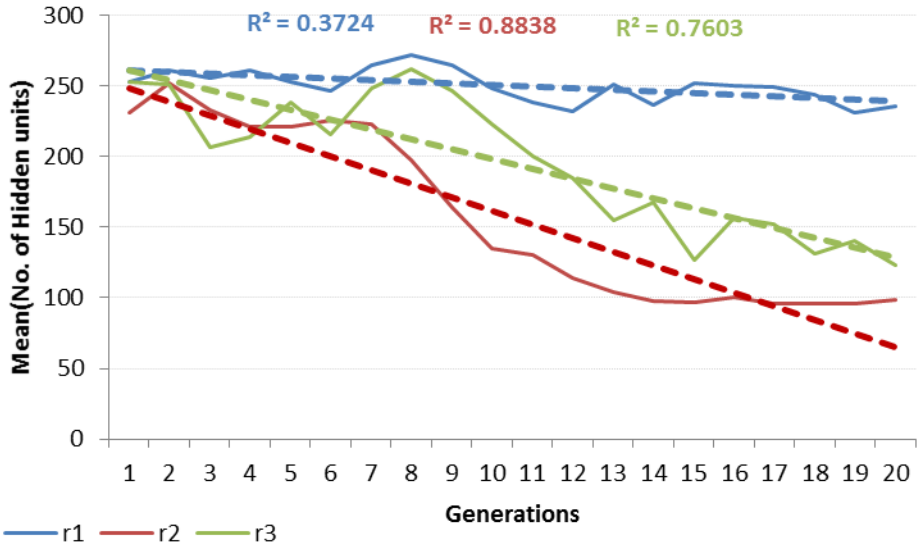


Figure 3.7(a): Change in the mean value of the number of hidden units per generation

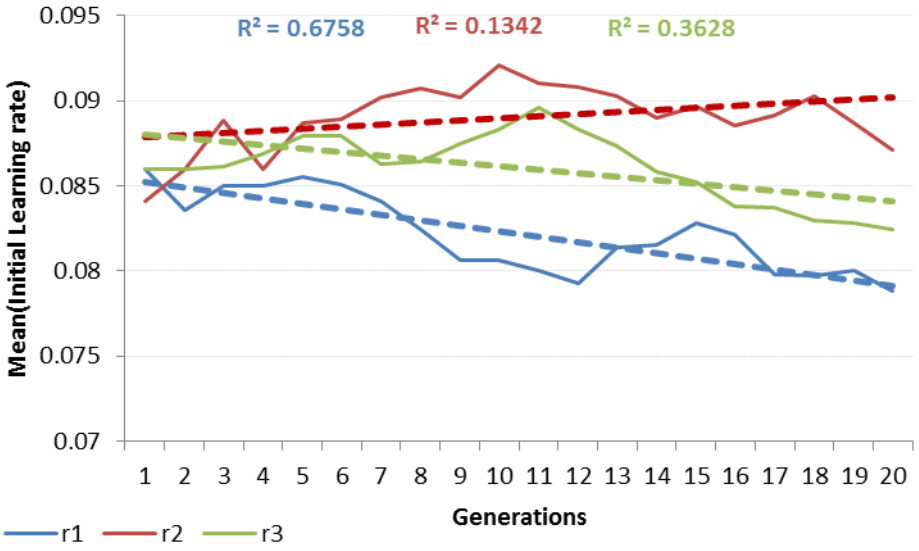


Figure 3.7(b): Change in the mean value of the initial learning rate per generation

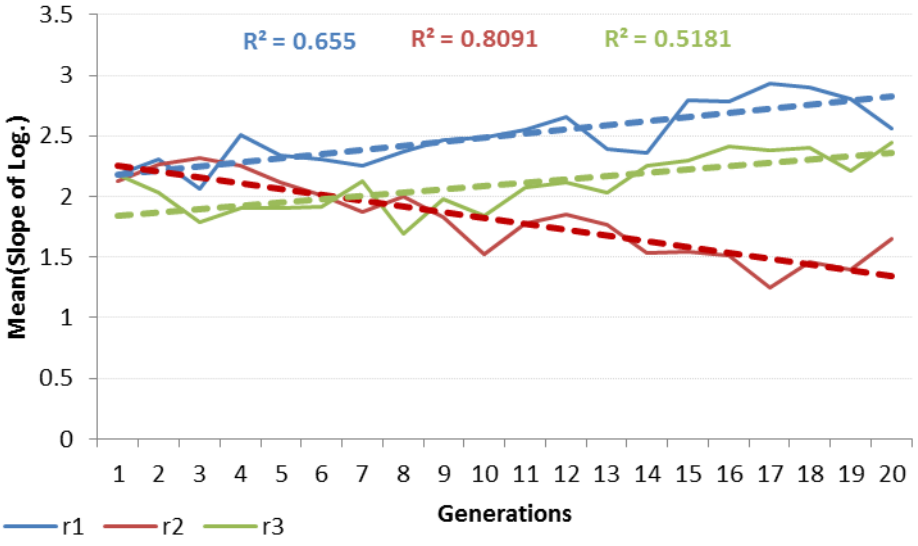


Figure 3.7(c): Change in the mean value of the slope of logistic activation per generation

For learning rate, the same pattern was observed, with an overall pattern of reduction across generations ($F(1,54)=22.69$, $p<.001$, $\eta_p^2=.296$) modulated by replication, with the reduction appearing in only two of the three replications ($F(2,54)=12.22$, $p<.001$, $\eta_p^2=.312$). Lastly, for slope of logistic activation, a differential pattern also emerged, this time with an increase in two of the populations across generations and a reduction in the other (main effect of generation: $F(1,54)=12.99$, $p<.001$, $\eta_p^2=.325$; interaction of generation*replication:

$F(2,54)=61.06$, $p<.001$, $\eta_p^2=.693$). Overall, replication 1 and 3 showed a common pattern of reduction in hidden units, reduction of learning rate, and increase in slope of logistic. For replication 1, the reduction in hidden units was milder, the learning rate fell lower, and the slope of logistic activation rose higher. Replication 2 showed a different pattern of a greater fall in hidden units, no change in learning rate, and a drop in the slope of logistic activation.

The three chosen parameters provided networks with capacity to learn (more hidden units can accommodate more input-output mappings) and/or ability to learn (optimum values of initial learning rate and steepness of logistic activation allow discovery of connection weights for those mappings). Irregular verbs belong to category of non-systematic mappings, which are more demanding on computational capacity. Figure 3.8 depicts the variation in the ranges of the three parameters across generations. It thus reflects the parameters being targeted by selection in each lineage.

Lineage/replication 1 improved irregular performance at the expense of regular, and this was reflected by maintenance of high levels of hidden units. Learning rates declined, while genes for steeper logistic slopes were selected.

Regular verbs have systematic input-output mappings, which are less demanding on computational capacity. Lineage/replication 2 improved regular performances at the expense of irregular verbs, and this was reflected by an increase in learning rate. Both hidden unit numbers and logistic slope declined.

In lineage/replication 3, the main improvement over generations was on regular verbs. As with lineage 2, there was a decline in hidden unit number, but unlike lineage 2 there was also a decline in learning rate. In contrast, the logistic slope showed an increase, which lineage 1 suggested was more sympathetic to accommodating irregular mappings.

We used roulette wheel selection as a representative of stabilising selection occurring in nature. Figure 3.8 confirms our claim that throughout lineages the parameter ranges change but not too drastically.

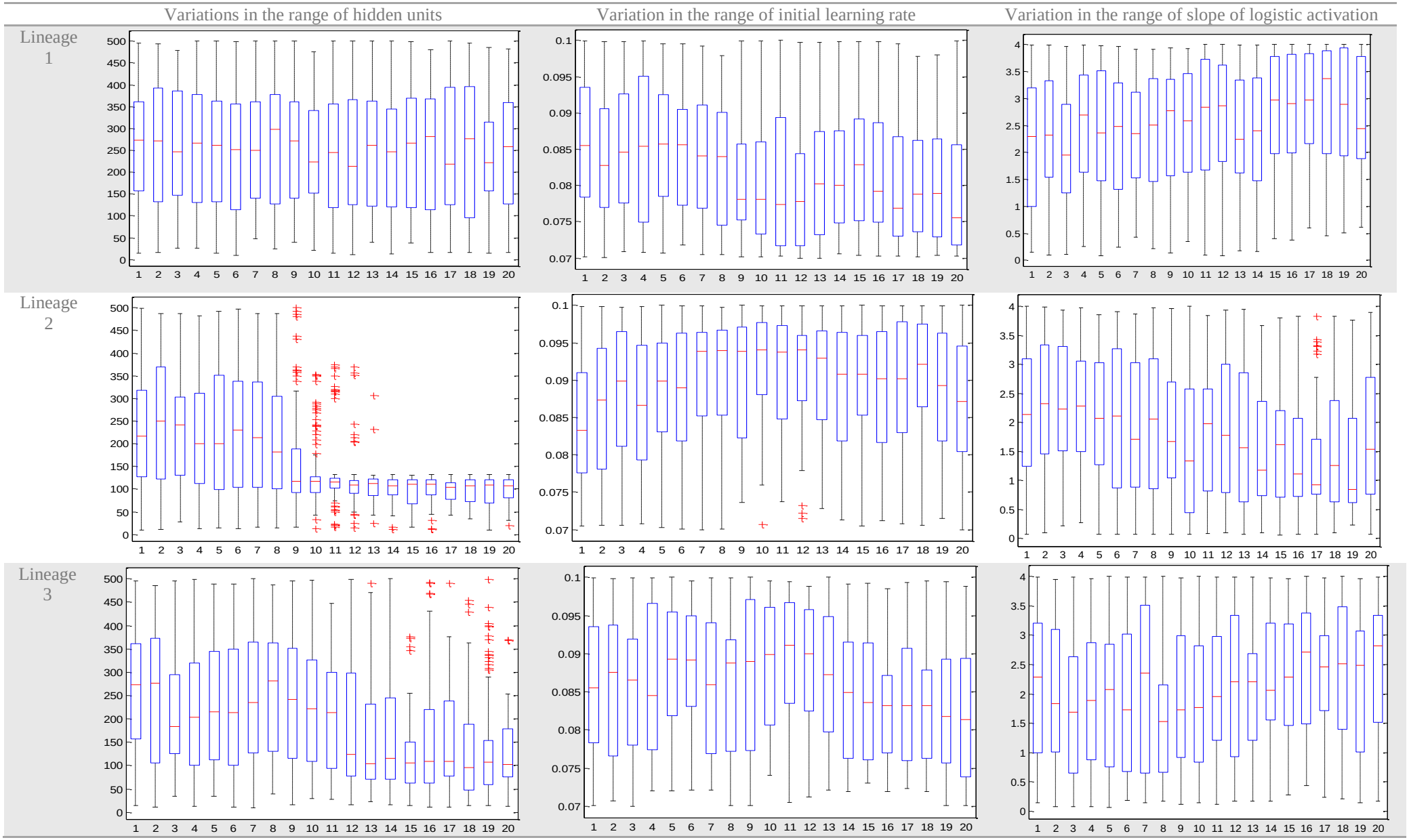


Figure 3.8: Range of Variation of Intrinsic parameters across Generations

In Figure 3.8, On every box, the central mark is the median, the edges of the box are the 25th and 75th percentiles, the whiskers extend to the most extreme data points not considered outliers and the outliers are marked separately by '+'. The height of the box represents the inter quartile range (IQR) of the data set, which is the difference between the 75th percentile and 25th percentile. The lines at the end of the whiskers mark the highest and lowest values of the data set that are within 1.5 times the inter quartile range of the box edge.

3.7 Truncation Selection based experiment results

Table 3.4 describes the experiment setting used in the second phase of framework evaluation using a more deterministic truncation selection operator, which is representative of directional selection occurring in nature. The next three lineages each with 20 generation duration were tested in this scenario.

Replications	R_4, R_5, R_6
No of Generations	20
Size of populations	Breeding = 100; Non-breeding= 100 Total R_4+ R_5+ R_6 across generations= 12,000 ANNs
Size of Datasets	Training= 508 Generalisation= 508
Training Mode	Batch
Max. training epochs	100
Early Stopping Criterion, maxstep (i.e. stop training if training accuracy does not improve till step == maxstep)	20
Initial weight update (Rprop learning rate)	Values from genome
Hidden units. Steepness of logistic	Values from genome
Selection Operator	Truncation (i.e. 50 best/top performers chosen at the end of training)
Crossover	6 crossovers/chromosome; different operators used
Environmental Factor (SES)	Probability value between 60% and 100%

Table 3.4: Experimental Design for truncation selection-based replications

3.7.1 Results and Analysis

The overall accuracy of the model on regular verbs was higher than that on irregular verbs. The mean performance on the full training set ranged between 88% - 92% for regular verbs, and between 15% - 35% for irregular verbs. The model was able to efficiently generalise the past tense rule in novel items with the mean accuracy ranging between 70% - 76%. As

discussed previously in Section 3.6.1, the performance of our model compares well with empirical data for children reported in the literature (Bishop, 2005; van der Lely and Ullman, 2001). It also compares well with two other past tense models (Thomas et al., 2009b) and (Karaminis and Thomas, 2010).

In this setting, we paralleled the analysis of replications 1-3 for new replications 4-6. That is, the results were analysed using independent linear regressions to assess performance / heritability / parameter changes for each population over the generations. Given the overall design, which combined repeated measures (e.g., regular verb performance, irregular verb performance, generalisation) and between group measures (replication population; breeding vs. non-breeding populations), trajectory analysis was used to assess overall patterns in the component linear regressions (Thomas et al., 2009a).

Figure 3.9 depicts the mean accuracy with which breeding and non-breeding twin populations formed past tenses for regular verbs across a sequence of generations, for three replications with differential initial genomes. Each of these graphs summarise the results from 12,000 networks. Figure 3.10 shows equivalent data for irregular verbs, while Figure 3.11 represents the generalisation results. In each case, a square on a zigzagged line indicates the mean accuracy level of the 100 networks per generation for each population, while a straight line represents the general trend observed in that experiment. The trend line was derived from a linear regression line based on the least squares method, predicting mean performance level from generation number. In some cases, R^2 values were relatively small, reflecting the non-monotonic changes in performance over generations. This is in line with changes in mean trait levels in animal populations following selective breeding, such as the open field behaviour of mice (DeFries et al., 1978; Plomin et al., 2008).

We initially considered performance of application of the past tense rule, comparing the measures of regular verb performance against generalisation, for the three replications and breeding versus non-breeding populations (12 trajectories). A fully factorial ANCOVA revealed significant increase in performance across the generations ($F(1,108)=67.61$, $p<.001$, $\eta_p^2=.385$). A similar differential pattern was observed between replications, with all replications showing rising performance ($F(2,108)=4.69$, $p=.011$, $\eta_p^2=.080$). This pattern was common across measures and breeding/non-breeding populations. Regular

verb performance (i.e. training performance) was reliably higher than generalisation ($F(1,108)=5.12$, $p=.026$, $\eta_p^2=.045$).

Irregular verb performance, also revealed significant change in performance across generations ($F(1,108)=24.45$, $p<.001$, $\eta_p^2=.185$), although the difference in performance gradient in different replications was of marginal significance ($F(2,108)=2.92$, $p=.058$, $\eta_p^2=.051$). Comparison to regular verb performance indicated that the performance gradients for regular and irregular verb types were not reliably different ($F(1,108)=.763$, $p=.384$, $\eta_p^2=.007$). Different replications did showed differential performance gradients depending on verb types ($F(2,108) = 7.10$, $p=.001$, $\eta_p^2=.116$).

The most notable difference between Figures 3.1 – 3.3 and Figures 3.9 – 3.11, is that right from the beginning of lineage(s), the performance levels achieved by the populations were higher in the latter case (truncation selection-based results), especially for past tense rule application i.e. regular verbs and generalisation. On the contrary, irregular verb accuracy levels were substantially lower in the latter case (truncation selection-based results), with replication 6 exhibiting a downward trend, despite the operation of a deterministic selection mechanism.

This difference in accuracy levels obtained in two settings transpires due to three important factors – (a) number of training epochs; (b) performance assessment used for fitness calculation and also for early stopping was based on cumulative (regular + irregular verb) accuracy; and finally (c) highly imbalanced dataset. One of the main differences between the roulette wheel based experiments and truncation selection based experiments is that in the former case, the networks were trained for 1000 epochs whereas in the latter case the maximum training epochs were limited to 100 with an additional early-stopping criterion. As per the early stopping criterion if the accuracy had not improved for certain number of epochs say n, [$(n = 20)$ in this case] then training stopped and the algorithm returned the best epoch performance. In both these settings, we used the cumulative verb performance (i.e. regular + irregular) for determining network accuracy, which in the latter setting was also used for checking early-stopping criterion. Additionally, since the past tense dataset is imbalanced comprising of 410 regular verbs in comparison to only 98 irregular verbs, the cumulative performance is mainly driven/modulated by regular verb efficiency.

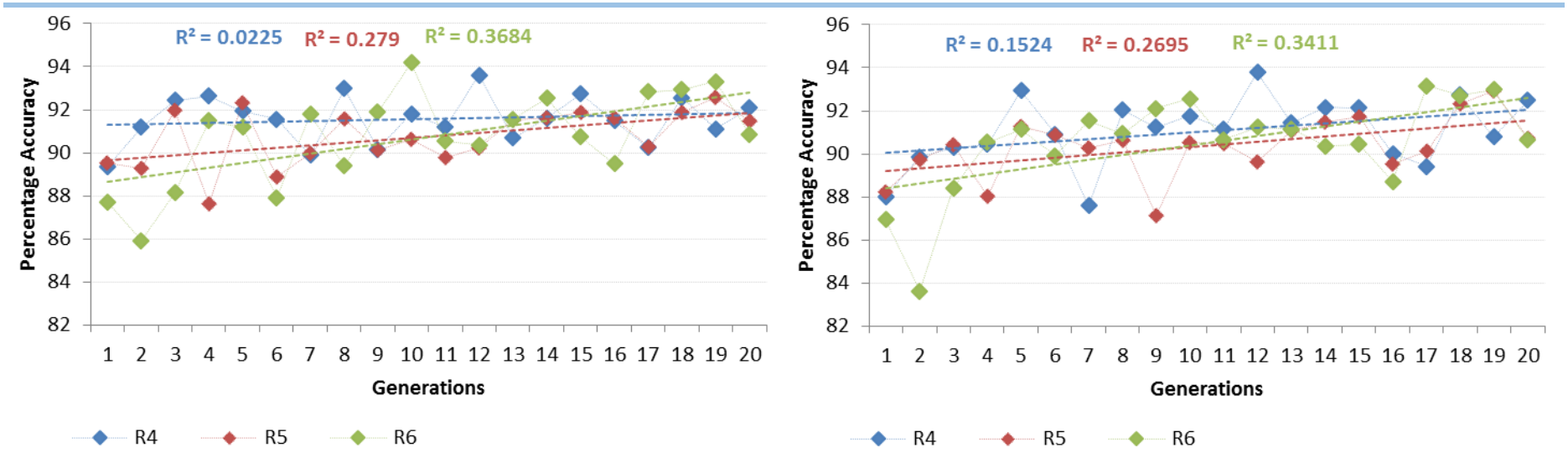


Figure 3.9: Mean performance per generation for breeding (left) and non-breeding (right) twin populations on regular verbs

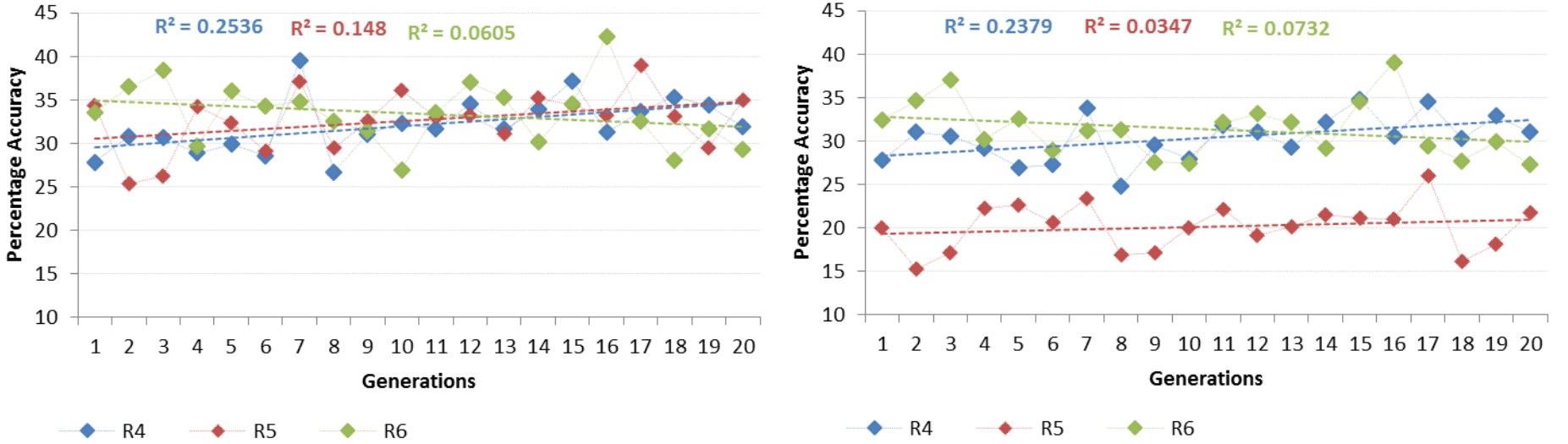


Figure 3.10: Mean performance per generation for breeding (left) and non-breeding (right) twin populations on irregular verbs

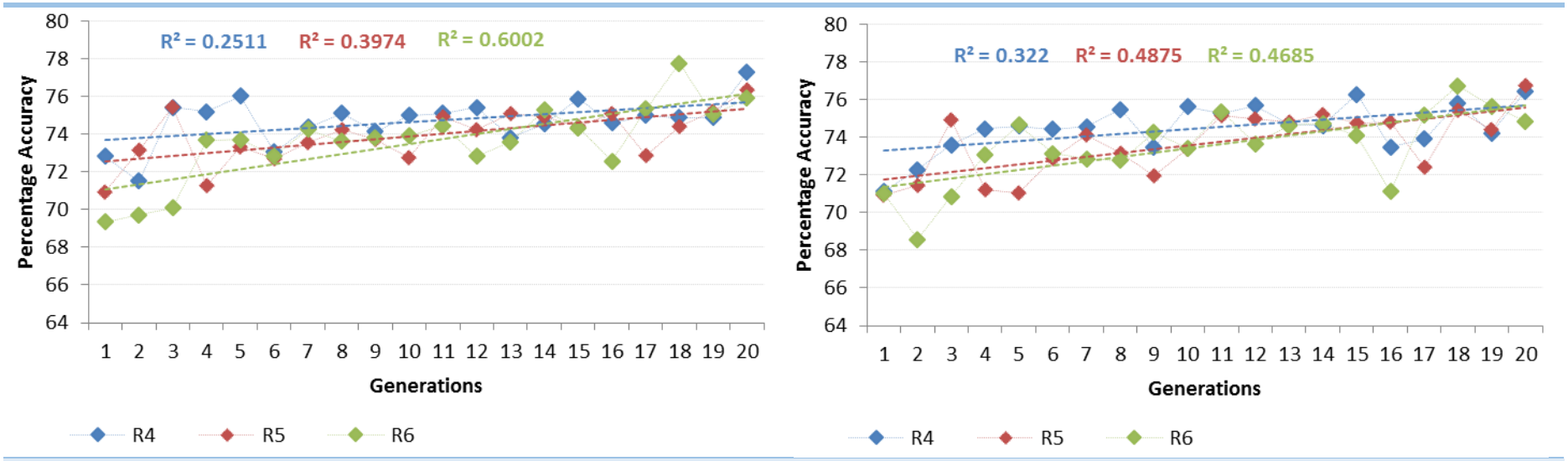


Figure 3.11: Mean generalisation accuracy per generation for breeding (left) and non-breeding (right) twin populations

Thus, based on the combined effect of the above-mentioned three factors, it can be deduced that in the roulette wheel based experiments population members were exposed to the training set for much longer duration and therefore were able to utilise their capacity (hidden units) to perform a sort of 'rote learning' for irregular verbs, and thus achieve decent performance despite the imbalanced dataset. Selection thus started targeting irregular verb performance at the expense of regular verb accuracy, as we have previously discussed (refer to Section 3.6.1). This consequently resulted in comparatively higher irregular verb accuracy and not-so-high past tense rule application accuracy (regular and generalisation).

By contrast, in the truncation selection condition, the population members were exposed to fewer epochs. Although networks here achieved very high cumulative accuracy levels, these were mainly modulated by regular verbs as seen from Figures 3.9 and 3.11. Since the networks weren't exposed to training set for as long and the fact that irregular verbs comprise less systematic mappings and thus are hard to learn, these two factors combined resulted in poor performance on irregular verbs in truncation selection based experiments. Finally, despite the fact that in this condition, only the fittest networks were chosen for breeding, the fitness was based on their cumulative accuracy, which in turn was mainly modulated by regular verbs. As a result, throughout the lineages irregular verbs maintained a rather poor performance whilst showing very high accuracy on regular and generalisation.

Figures 3.9–3.11, suggest that in the current scenario, selection by mean performance was driven by stronger regular verb performance and was consequently targeting parameter range(s) that favour learning of regular mappings. This is why we have improving/increasing accuracy trends for past tense rule application (regular and generalisation). Whereas for irregular verbs, although the performance trends show an improvement in two out of three replications, the accuracy levels are pretty low. Low irregular accuracy levels have already been discussed but the overall improving trends could be due to 'domain-relevant' computational parameters. This implies that although the selection is actively targeting regular verbs and consequently moving/optimising the neuro-computational parameter range to suit past tense rule application, it just so happens that this selected range of parameters is also relevant (or tends to work for) for irregular verbs as well. In replication six, however this does not hold true and which is why irregular verb performance showed a decreasing performance trend.

These performance trends further strengthen the claim that when selection is applied on a quasi-regular task, different aspects of the task may be optimised depending on the genetic propensities of initial populations as well as stochastic selection factors. The trend then continues throughout the lineage because of genetic inheritance, making it metaphorically similar to Waddington's epigenetic landscape (Waddington, 1957), as discussed in Section 3.6.1.

Thus, if, as shown in lineage 6 (which has steepest rise in regular verb accuracy trend and declining irregular verb accuracy trend) in Figures 3.9/11 and 3.10, the first few generations improve their learning of regular verbs at the expense of irregular verb performance, the lineage is committed to this pathway. Genes for good learning of irregular verbs have been lost from the gene pool. Evolution cannot go into reverse gear and find a pathway that combines good learning on both verb types. Replications 4 and 5 showed improvement in both verb types across generations.

Following the same analysis pattern as in roulette wheel based experiments, next we examined correlations in performance between MZ and DZ network twin pairs, using Falconer's equations to derive estimates of heritability (Plomin et al., 2008). Again the plotted data should be seen as proportion to the heritability and environmentability observed in populations, rather than direct estimates. Figure 3.12 shows the estimates of heritability (variance due to genetic factors) for regular (Fig.12a) and irregular verbs (Fig.12b). These six trajectories were compared in a fully factorial ANCOVA. Heritability reliably reduced over generations ($F(1,54)=21.65$, $p<.001$, $\eta_p^2=.286$), and this pattern was modulated by measure ($F(1,54)=9.86$, $p=.003$, $\eta_p^2=.154$), but not by replication population. However, there was significant interaction between Measure*Replication*Generation ($F(2,54)=5.01$, $p=.010$, $\eta_p^2=.157$), thereby implying that the relationship between the change in heritability of regular and irregular verbs across generations was different in different replications.

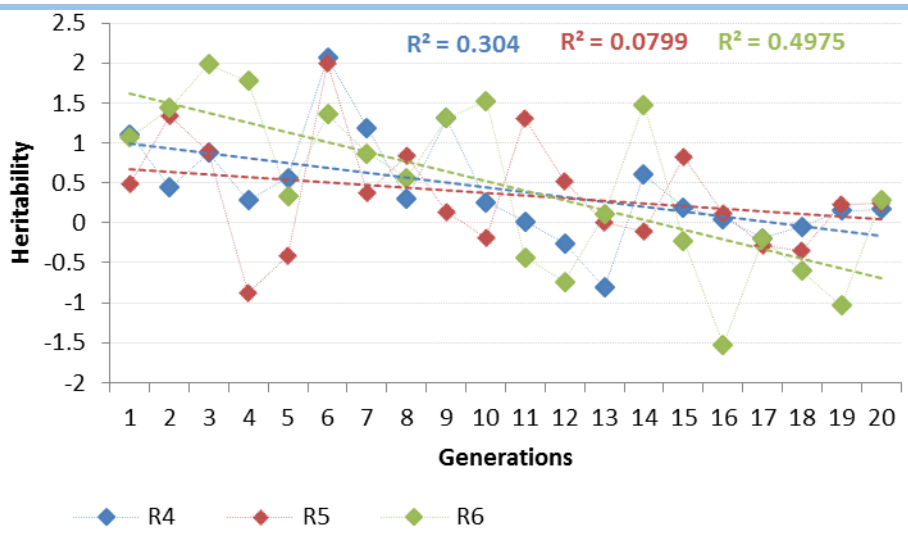


Figure 3.12(a): Heritability or proportion of variance due to genetic (or structural) factors for Regular Verbs.

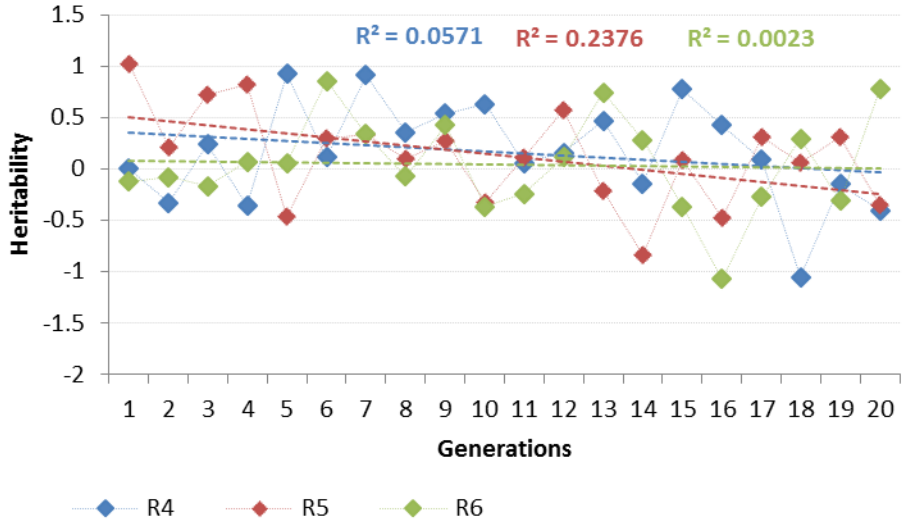


Figure 3.12(b): Heritability or proportion of variance due to genetic (or structural) factors for Irregular Verbs.

It has been already established (refer to Section 3.6.1) that optimisation and heritability should have an inverse relationship. In line with this expectation, in all lineages 4, 5 and 6 accuracy on regular verbs improved and heritability dropped. Similarly for irregular verbs, performance improved in replications 4 and 5 and correspondingly heritability reduced, albeit gradually. In replication 6, irregular verb performance declined and heritability maintained at almost the same level. Also, heritability for regular verbs was initially higher than that for irregular verbs, and it then decreased across generations, signifying effect of selection for parameter sets specialised for regularity. By contrast, heritability of irregular

verbs was initially lower, and it then decreased with generations gradually, implying lack of selection for parameter sets specialised explicitly for irregular mappings. The gradual decline indicates that, despite not being specialised for irregular verbs, the range still aided in acquisition of irregular past tense verbs, i.e. was domain relevant.

When heritability of a particular aspect of the task reduces, it implies that variance in performance is less due to genetic factors and more due to shared and non-shared environmental factors. Figures 3.13(a) and 3.13(b) display the variance due to shared environmental factors, in this case the filtered training datasets. The effect of shared environment reliably increased over generations ($F(1,54)=9.62$, $p=.003$, $\eta_p^2=.151$) with gradients for two verb types being reliably different across lineage ($F(1,54)=4.59$, $p=.037$, $\eta_p^2=.078$); and finally the replications showed differential gradient depending on the measure (i.e. verb type) i.e. gradients for regular verbs were significantly steeper than those of irregular verbs ($F(2,54)=5.24$, $p=.008$, $\eta_p^2=.163$).

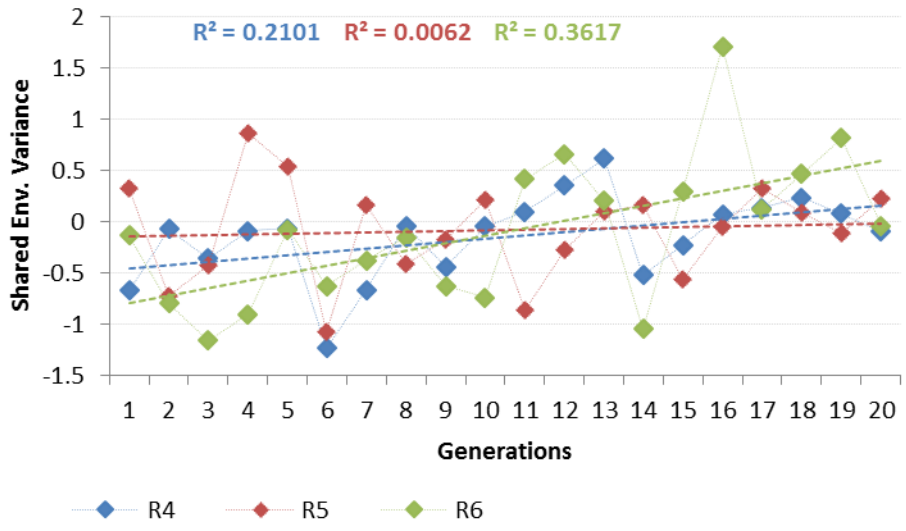


Figure 3.13(a): Proportion of variance due to shared environmental factors - Regular Verbs.

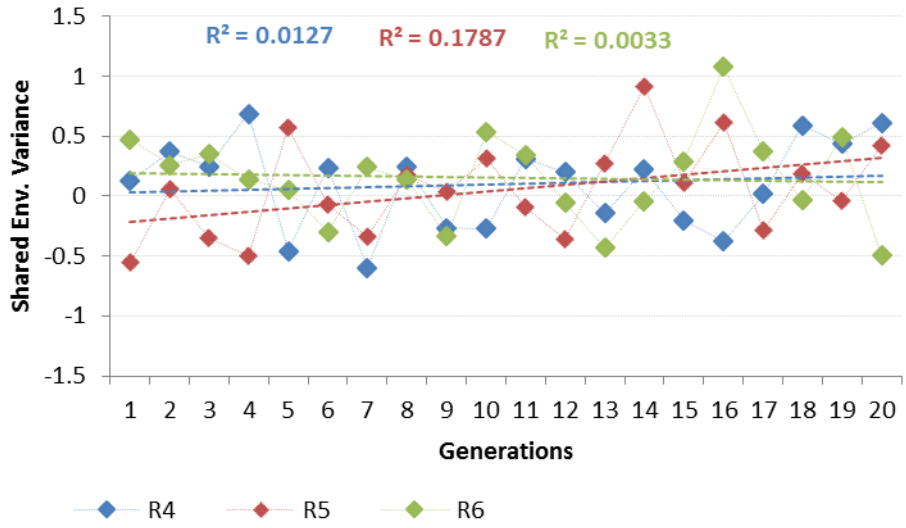


Figure 3.13(b): Proportion of variance due to shared environmental factors - Irregular Verbs.

Figures 3.14(a) and 3.14(b) represent the variance in performance due to non-shared environmental factors, or initial weights in our implementation. Analyses revealed reliable effect of generation ($F(1,54)=13.25$, $p=.001$, $\eta_p^2=.197$) i.e. variance due to unique environmental effects increased reliably with generations. The analysis also revealed significant interaction between Measure*Generation ($F(1,54)=42.10$, $p<.001$, $\eta_p^2=.163$) suggesting differential gradients of variance due to unique environmental effects for two verb types i.e. gradients for regular verbs were significantly steeper. The figures show that the differences in initial weights led to large variability in behavioural outcomes.

The aforementioned analysis are in contrast with the variance due to non-shared influences analysis done for roulette wheel based results (refer Figure 3.6(a) and Figure 3.6(b)), wherein no reliable effects were revealed. This difference in the role of non-shared influences in truncation-based and roulette-wheel-based results is probably attributed to the higher accuracy levels attained in truncation-based lineages. It is known that learning speed and fast convergence depends, to some extent, on connection weights. Therefore, in truncation-based lineages ANNs relied more on their connection weights to achieve accuracy levels and a small difference in weights led to significant difference in mean accuracy levels attained by ANNs.

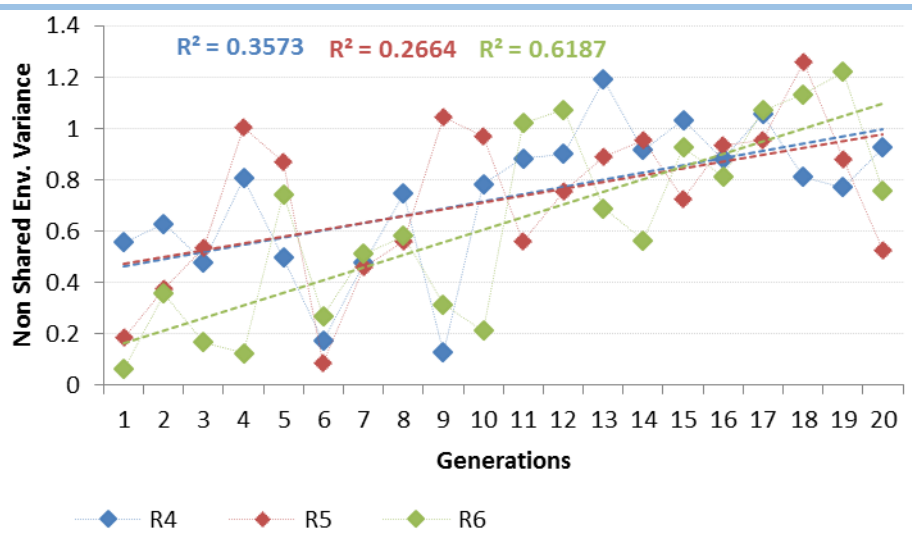


Figure 3.14(a): Proportion of variance due to Non-shared environmental factors - Regular Verbs.

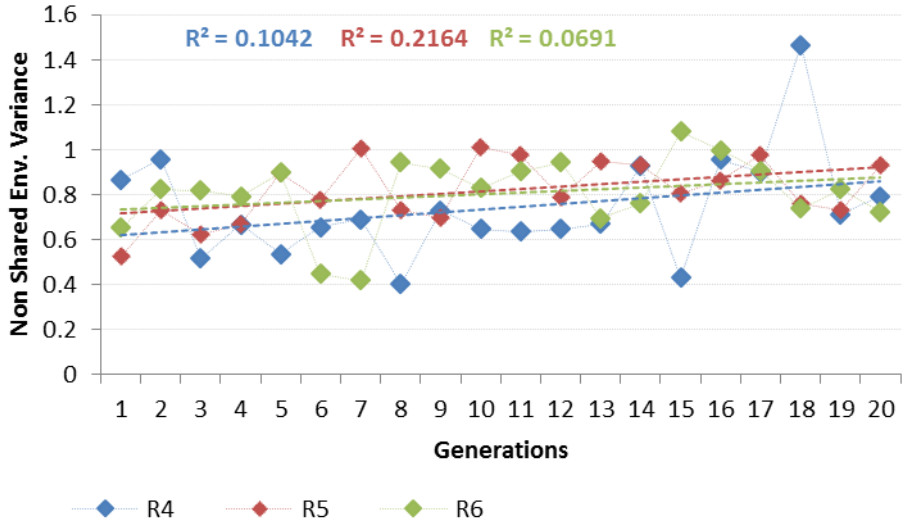


Figure 3.14(b): Proportion of variance due to Non-shared environmental factors - Irregular Verbs.

The next step of analysis within this setting, then, was to examine the change in mean parameter values for a given lineage across generations. This should reveal the domain-relevant parameters that were selected, in those cases where performance on one verb type was enhanced at the expense of the other, and therefore in turn reveal the drivers behind changes in heritability. Figure 3.15 depicts changes in mean parameter values for number of hidden units, initial learning rate, and slope of the logistic activation function. For hidden units - Figure 3.15(a), there was a reliable increase in number of hidden units across generation ($F(1,54)=255.42$, $p<.001$, $\eta_p^2=.825$), although the gradients weren't quite reliably different for the three lineages ($F(2,54)=2.99$, $p=.058$, $\eta_p^2=.100$).

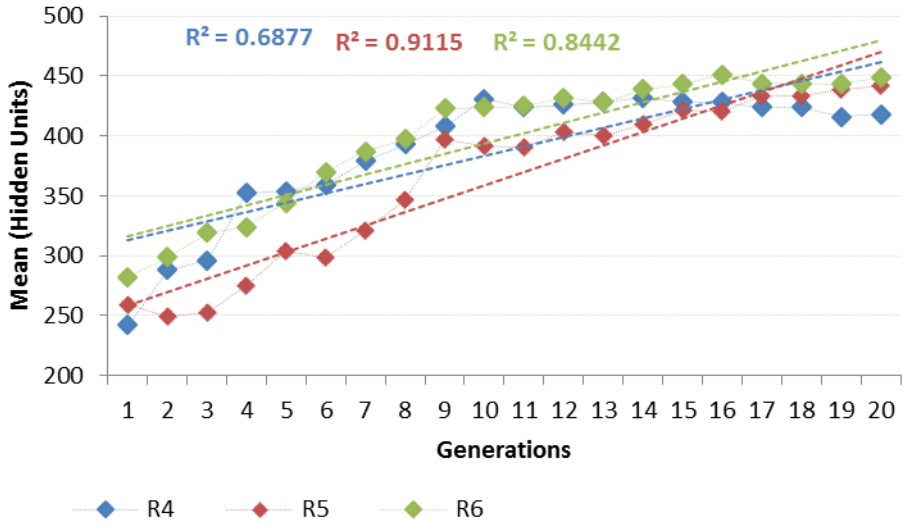


Figure 3.15(a): Change in the mean value of the number of hidden units per generation

For learning rate, the same pattern was observed, with an overall pattern of reduction across generations ($F(1,54)=42.83$, $p<.001$, $\eta_p^2=.442$) modulated by replication, with the reduction appearing in only two of the three replications ($F(2,54)=10.86$, $p<.001$, $\eta_p^2=.287$). Lastly, for slope of logistic activation, an overall pattern of reduction across generations was observed $F(1,54)=252.93$, $p<.001$, $\eta_p^2=.824$) which however, was not modulated by replication $F(2,54)=1.26$, $p=.291$, $\eta_p^2=.045$).

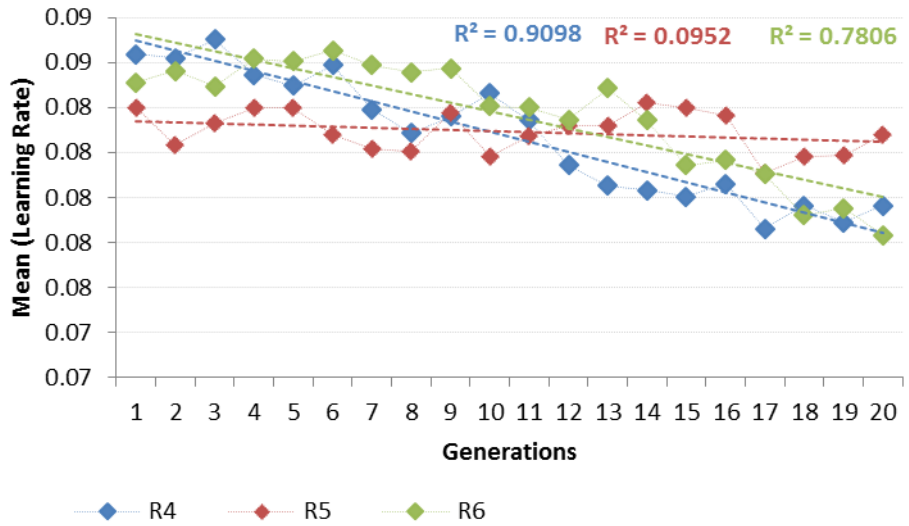


Figure 3.15(b): Change in the mean value of the initial learning rate per generation

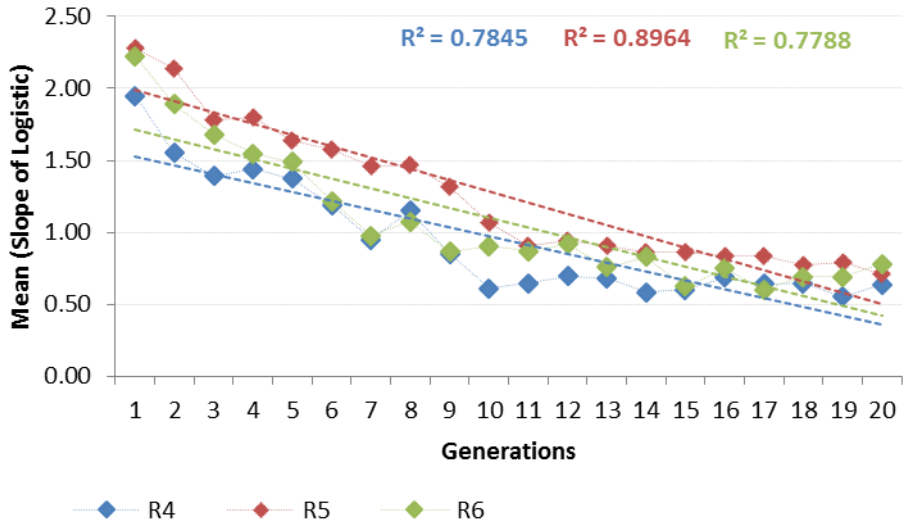


Figure 3.15(c): Change in the mean value of the slope of logistic activation per generation

Overall, replications 4, 5 and 6 showed a common pattern of increase in the number of hidden units, reduction of learning rate and slope of logistic activation more uniformly than roulette wheel based replications. The only exception was in replication 5, wherein the learning rate maintained constant throughout. These chosen parameters, and the developmental pathway their range of variation followed over generations, provided networks with capacity to learn (more hidden units can accommodate more input-output mappings) and ability to learn (optimum values of initial learning rate and steepness of logistic activation allow discovery of connection weights for those mappings). The networks were not only able to attain very high accuracy on regular verbs but were also able to maintain and even improve performance on irregular verbs, which belong to category of non-systematic mappings, and are more demanding on computational capacity. However, the accuracy levels achieved for irregular verbs in truncation-based lineages weren't as high as those achieved in roulette wheel based replications. Figure 3.16 depicts the changes in the range of variation of the neuro-computational parameters in truncation selection based lineages. Although the range of variation of neuro-computational parameters was kept the same at the beginning of each generation, by means of sexual reproduction and fitness based selection, networks with good capacity and good/quick learning abilities were chosen for breeding and thus the range of variation of neurocomputational parameters appears skewed.

Comparing Figure 3.16 with Figure 3.8, focusing on lineage 1, wherein selection was actively targeting irregular verbs (with accuracy as high as 40%, higher than other lineages), we can see that there are almost opposite trends of parameter range optimisation. In lineage1, hidden units were increasing, however the learning rate dropped and the slope of logistic increased as well. This shows that regular and irregular verbs were sensitive to different parameter ranges. However, as the performance results for lineage 4, 5 and 6 have depicted that even if the parameter (or range) being targeted is not quite domain specific (in case of irregulars), the populations could still learn and improve their performance on random mappings because the range tends to be domain relevant, despite substantially fewer learning epochs available. By domain relevant, in this work we imply/interpret that selection is not targeting/optimising for that particular domain/task, however the resulting range of variation can still help in learning the said task (refer Karmiloff-Smith, 1998 for actual definition of domain-relevant).

Since truncation selection was used as a representative of directional selection that occurs in nature, the changes in the range of variation as depicted in Figure 3.16 affirm that our method does yield results similar to how actual directional selection works by favouring one end of the range.

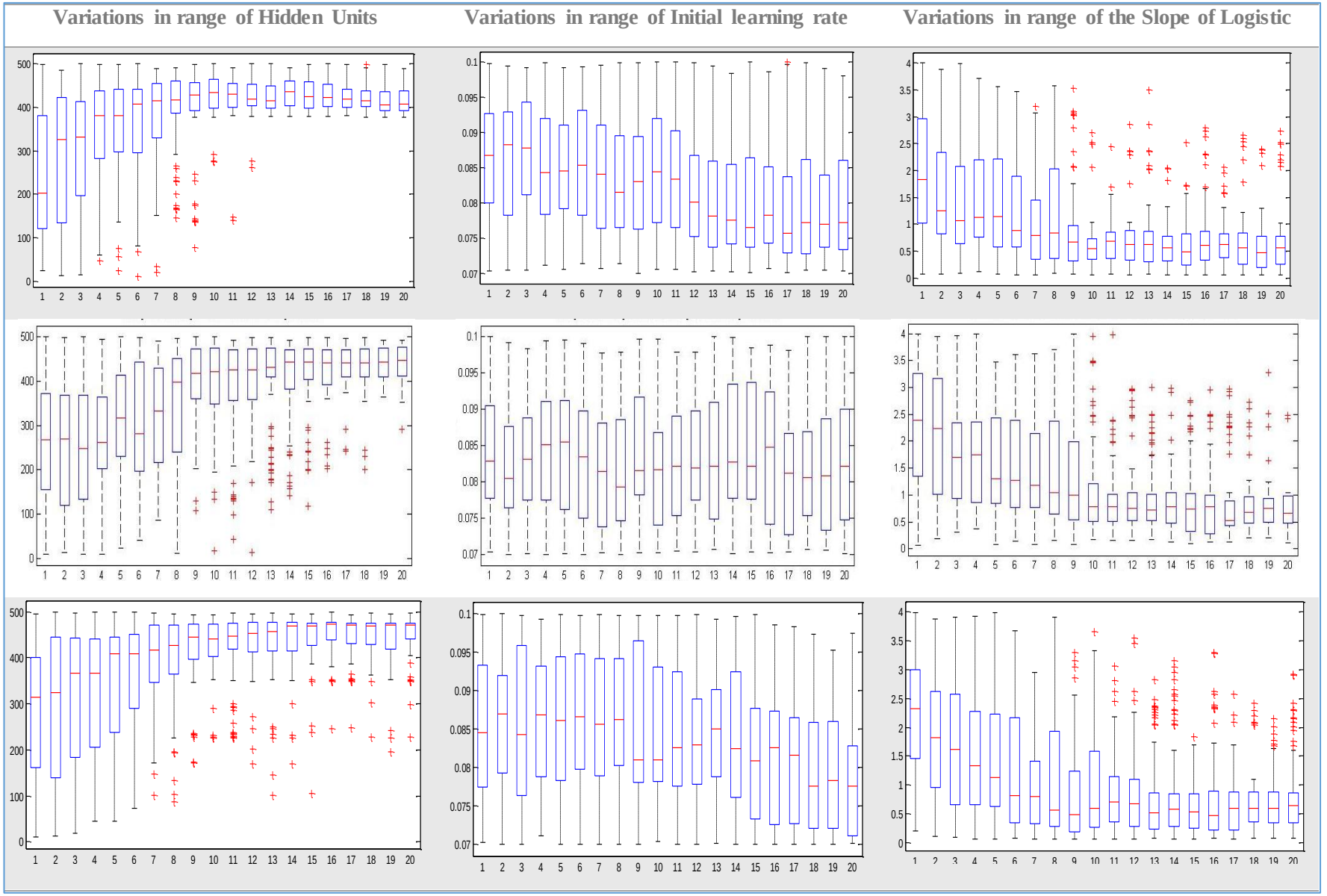


Figure 3.16: Range of Variation of Intrinsic parameters across Generations

3.8 Analysing the effects of selection

Through the course of this chapter we have maintained that evolution (via selection) and learning interact and complement each other. Our roulette wheel and truncation-selection-based experimental results support this claim. However, in order to see if the differences in verb performances and differences in genetic and environmental contributions were reliably modulated by the type of selection operator, we used a fully factorial ANCOVA design once more, directly comparing replications 1, 2 and 3 with replications 4, 5 and 6 and included a categorical factor of selection operator. We asked the following question, is the rate of increase/decrease of this particular entity modulated by selection type? The answers are listed below.

Beginning with performance, for regular verbs the rate of increase in accuracy over generations was modulated by selection type ($F(1,108)=5.29$, $p=.023$, $\eta_p^2=.047$), implying that regular verbs attained higher accuracy levels and had a faster rate of change over generations in truncation selection lineages. Since generalisation patterns over regular verbs, we observed similar relationship whereby selection significantly modulates differential gradients in two settings ($F(1,108)=25.13$, $p<.001$, $\eta_p^2=.189$). On contrary, for irregular verbs, the rate of change in accuracy over generations was not reliably modulated by selection type ($F(1,108)=.992$, $p=.321$, $\eta_p^2=.009$). This might suggest that differences between performance trends of irregular verbs in two settings were more due to differences in training period. The fact that in latter setting, the networks had capacity which supports learning random mappings, but still didn't reach the same accuracy levels as in first setting is again indicative of differences in accuracy levels being not so much modulated by selection but due to training period differences. Thus, to summarise, selection (type) had differential effect over verb types, wherein regular verbs (including generalisation performance) had reliably higher accuracy and faster rate of increase over generations in truncation-based lineages. This trend, however, was not replicated by irregulars.

Next, we assess whether selection mechanism modulated change in heritability over generations. The rate of decrease in heritability of regular verbs was found to be significantly modulated by selection ($F(1,108)=7.64$, $p=.007$, $\eta_p^2=.066$), implication being the rate of decrease in regular verb heritability over generations is faster in truncation-based lineages; however differences in irregular heritability gradients weren't due to selection, as shown by

nonsignificant selection* generation relationship ($F(1,108)=.006$, $p=.941$, $\eta_p^2=.000$). However, the interaction of selection*measure*generation revealed that the selection method reliably altered regular verb heritability change over generations but not irregular ($F(1,108)=5.51$, $p=.021$, $\eta_p^2=.049$), i.e. the modulatory effect of selection was differential over verb types. In contrast, for environmentability i.e. variance in performance due to shared environmental factors, analyses revealed no reliable effects, with selection having no effect on variance due to shared environment for either verb type. However, the rate of increase in non-shared environmental influences was reliably modulated by selection and was steeper in truncation-based lineages ($F(1,108)=23.97$, $p<.001$, $\eta_p^2=.18$). This effect was differential over verb types i.e. rate of increase in non-shared environmental influences for regular verbs was reliably faster compared to irregular verbs ($F(1,108)=14.12$, $p<.001$, $\eta_p^2=.116$).

Finally, for neurocomputational parameters, we found that rate of change in hidden units was significantly modulated by selection ($F(1,108)=443.82$, $p<.001$, $\eta_p^2=.80$). However, learning rate revealed no reliable effect of selection ($F(1,108)=.931$, $p=337$, $\eta_p^2=.009$). This has partly to do with the fact that we have used RPROP algorithm for training, which derives its own learning rate from the initial value provided. Lastly, the slope of logistic activation also had differential gradients in two settings, implying significant modulation by selection method ($F(1,108)=178.60$, $p<.001$, $\eta_p^2=.623$).

Therefore, it can be concluded that the modulating effects of selection were increasingly reliable in deterministic (i.e. fitness-based) selection scenario. This occurs since only the fittest members of population get chosen for breeding and these have intrinsic parameters suited for acquisition of learning task, which get passed onto offspring. The changes occurring over generations thus become faster and more reliable as a result. In contrast, in roulette wheel based lineages, not-so-fit members of populations also tend to get selected for breeding owing to stochastic nature of selection operator. Thus, the genes inherited by offspring were not necessarily relevant to learning domain and ergo the changes over generations in performance and other measures were not necessarily relevant.

To sum up, since most results obtained from above analysis were significantly modulated by selection, this implies that evolution (acting via fitness based selection) does indeed guide learning and thus results in diverse overt behaviours. Further since the fitness (used as deterministic measure for selection) is derived from quality of learning of population members,

it also indirectly affects how evolution will shape up, i.e. which developmental pathway will get chosen for that lineage. Thus a circular relationship exists between evolution, learning and overt behaviour.

3.9 Summary and contribution of the chapter

The following is the summary of the findings derived from the neuro-evolutionary past tense acquisition model presented in this chapter: (a) Applying selection on the individual's performance level in a quasi-regular task such as past tense acquisition results in the emergence of divergent behaviours depending on initial conditions – both genetic and environmental. (b) Once selection starts targeting a particular aspect of task domain, it starts behaving similar to Waddington's epigenetic landscape. That is, from an initial pluripotent state, the developmental (or learning) pathway of populations (in that lineage/replication) becomes more specialised in the particular targeted aspect. Reversing this trend is difficult, if not impossible. (c) Selection based on a stochastic method such as roulette-wheel, when combined with sexual reproduction method for population generation, has a limiting effect on final behavioural (or performance) levels achieved. Performance is affected in two ways: first, since roulette-wheel selection has a stochastic nature, there is a possibility of not-so-fit members being selected in the breeding pool. Secondly, the sexual reproduction method used to generate offspring prevents reliable transfer of best properties from parents to offspring. (d) However, selection based on a deterministic method when coupled with sexual reproduction method has an encouraging effect on final behaviour, for both aspects of quasi-regular task. As we saw in Section 3.7.1, although the selection was seemingly targeting regular verbs, the irregular verb performance albeit less accurate, still had an upward trend in accuracy, despite substantially fewer learning epochs. (e) Heritability acts as an identifier of the aspect of the quasi-regular task being targeted by selection. Highly heritable behaviour indicates that the trait is not being selected for, whereas behaviour with low heritability implies selection and optimisation. Thus an inverse relationship exists between heritability and optimisation. (f) A higher proportion of variance caused due to shared environmental factors (filtered training sets) is an indicator of an optimised population of learners. In other words, it shows that the particular population members have greater genetic predisposition of successfully acquiring the desired behaviour or task. (g) Non-shared environmental factors (initial weights) lead to significant proportions of behavioural variance. This effect becomes magnified when intrinsic properties are not particularly suitable. In such cases, having good initial weights can provide networks with the extra support needed to

acquire a task. Hence, training could be biased towards non-shared factors to improve performance.

Several avenues would merit further investigation. More complex genome representations, for example, may allow encoding more computational parameters and increasing genetic variability. Also, it is necessary to understand the implications for non-random assignment of environments to genotypes implied by gene-environment correlations believed to hold in human populations (Plomin et al., 2013).

Chapter 4 Behavioural Genetics inspired model for Transfer Learning

4.1 Overview

This chapter focuses on the application of BG inspired neuroevolutionary framework to evolve individuals (ANNs) capable of learning task(s) different from those for which they are selected, i.e. a scenario wherein the evolutionary task is different from the learning task(s). In such a situation, the members of population have to evolve (or become fitter) at the population level on the evolutionary (or source) task and also learn various other tasks at the individual level. From our previous experiments (refer to Chapter 3) we know that selection by mean performance driven by the evolutionary (or source) task fitness tends to optimise neuro-computational parameters in a way that enhances learning of source task mappings. However, different tasks can have mappings that are differentially sensitive to different parameters in ANNs. These two points combined could cause a potential conflict for populations trying to learn tasks different from those for which they were selected. However, research in psychology shows that the same set of genes is mostly responsible for genetic influence on varied cognitive areas (Kovas and Plomin, 2007). In other words, genes affecting one ability like reading are the same genes which affect other completely different cognitive ability such as mathematics. These genes are referred to as 'generalist genes' (Kovas and Plomin, 2007). In the current work the objective is to exploit this idea of generalist genes in order to devise a method that enables the populations to store and reprocess the knowledge gained while learning one task to learn completely different tasks. This concept is commonly known in the field of machine learning as heterogeneous transfer learning, or simply transfer learning (when used more generically), where, with few exceptions, only one source task and one target task are considered. Transfer learning methods provide frameworks to exploit previously acquired knowledge to solve new but similar learning problems faster and with better solutions. The rest of the chapter is organised as follows: In Section 4.2, a review of current trends in transfer learning field is presented. The open questions and research issues in the field of transfer are then presented in Section 4.3. Section 4.4 discusses heterogeneous transfer and challenges in performing heterogeneous transfer. Next we summarise the main observations drawn from previous research and outline the research questions we address though our transfer approach in Section

4.5. Our BG inspired transfer approach is presented in section 4.6 followed by summary and contribution of chapter in Section 4.7.

4.2 Introduction to Transfer Learning

Traditional machine learning strategies are very popular amongst researchers in different computational fields; however, most of them work under a number of assumptions such as, learning for each new task is usually isolated and begins from scratch. Another requirement is that the training and testing data have identical feature spaces with underlying distribution (Pan and Yang, 2010). Fulfilling these assumptions is often a difficult and expensive process; hence there is a growing need for methods of learning that can prevent *reinventing the wheel* by using some sort of knowledge transfer between several tasks.

Transfer learning is a research field in machine learning which aims to store and reprocess the knowledge gained while learning one task to learn different but related tasks. The task from which the knowledge is extracted is called the *source* task and the novel task to which it is applied is the *target* task. This concept draws inspiration from the psychological concept of *transfer of practise*, proposed by Thorndike and Woodworth (1901), which explores how "enhancement in one mental function" could influence another related one. Human learners have the ability to recognise and apply suitable knowledge acquired from a previous learning experience when facing a new but similar task. Some examples include, learning to play tennis helps a person to learn to play badminton, learning mathematics helps in learning physics and learning to play chess can help one become a better strategic planner in business or politics.

The vital motivation for transfer of learning in the field of machine learning was first discussed in NIPS-95 workshop on 'Learning to Learn', wherein the focus was on the need for lifelong machine learning methods that retain and reuse learned knowledge. Since then research on transfer learning has become more and more popular in different names such as: learning to learn, knowledge transfer, multitask learning, metalearning, context sensitive learning (Pan and Yang, 2010; Thrun and Pratt, 1998). Amongst these, a learning technique very similar to transfer learning is the multi task learning framework proposed by Caruana, (1997). In multi task learning methods, learning in the source and target tasks happens simultaneously and it exploits implicit pressures from additional training patterns, via shared or common internal

representations. The aim here is to optimise performance on all tasks. In transfer learning, on the other hand, source and target learning occurs separately in time and an explicit representation is transferred from the source to the target. It cares most about learning the target task.

Hence, the goal of transfer learning is to improve learning in task by utilising knowledge acquired whilst learning the source task. There are three main ways through which transfer can potentially improve learning of target task. Firstly, it might reduce the total time taken to learn the target task given the transferred knowledge compared to time taken to learn it from scratch. Secondly, an improved final performance can be achieved in the target task in presence of transfer versus without transfer. Finally, more accurate initial performance that can be achieved using the transferred knowledge only, without any further learning, compared to initial performance achieved by using initial random settings (Torrey and Shavlik, 2009). Due to these advantages, transfer learning approaches have been increasingly applied to various areas such as activity recognition (Hu and Yang, 2011), eye tracking (Shell et al., 2012), gaming (Sharma et al., 2007), image classification (Quattoni et al., 2008), NLP problems (Blitzer et al., 2006), named entity recognition problems (Arnold et al., 2007), and Wi-Fi localisation models (Yin et al.,2005; Pan et al., 2007; Zheng et al., 2008) to name a few.

There are different types of transfer learning methods. To better understand them, some basic definitions and notations are explained as follows. Transfer learning has two main components – a *domain* and a *task*. As explained in Pan and Yang, (2010), a *domain* D consists of two components – a feature space χ and a marginal probability distribution $P(X)$, where $X = \{x_1, x_2, \ldots, x_n\} \in \chi$. Two domains are said to be different if they have, either different feature spaces or different probability distribution or both. Given a particular domain $D = \{\chi, P(X)\}$, a *task* comprises of a label space $Y = \{y_1, y_2, \ldots, y_m\}$ and a predictive function $f(.)$, and is denoted as $T = \{Y, f(.)\}$. The predictive function is not observable but can be learned from training data pairs $\{x_i, y_i\}$, where $x_i \in X$ and $y_i \in Y$. The function $f(.)$ is used to predict the corresponding label $y_i = f(x_i)$, of a new input instance x_i. The *source domain* is defined as $D_s = \{(x_{s_1}, y_{s_1}), \ldots, (x_{s_n}, y_{s_n})\}$, where $x_s \in X_s$ is the source instance and $y_s \in Y_s$ is the corresponding class label. The *target domain* is represented as $D_t = \{(x_{t_1}, y_{t_1}), \ldots, (x_{t_n}, y_{t_n})\}$ where $x_t \in X_t$ is the target instance and $y_t \in Y_t$ is its corresponding class label. Based on this notation, Pan and Yang (2010) define transfer learning as:

Given a source domain D_s and a learning task T_s, a target domain D_t and learning task T_t, transfer learning aims to improve the learning of the target predictive function $f_t(.)$ in D_t using the knowledge of D_s and T_s where $D_s \neq D_t$ or $T_s \neq T_t$.

In the above definition, the condition $D_s \neq D_t$ implies that either feature space between domains are different, i.e. $X_s \neq X_t$ or the marginal probability distribution are different, i.e. $P(X_s) \neq P(X_T)$. For example, assuming that the task is document classification then the former corresponds to when two sets of documents are in different languages and the latter might correspond to when source and target domain documents focus on different topics. Likewise, the condition $T_s \neq T_t$ indicates that either the label spaces between domains are different, i.e. $y_s \neq y_t$, or the conditional probability distribution between domains are different, i.e. $f_s(.) \neq f_t(.)$. Following the abovementioned document classification task, the former corresponds to situation wherein source domain has binary document classes whereas target document domain has 20 potential document classes, while the latter situation corresponds to a scenario wherein source and target documents are unbalanced in terms of user defined classes. Additionally, there exist some explicit or implicit relationships between feature spaces of two domains such that it can be inferred that the source and target domains are related (Pan and Yang, 2010; Lu et al., 2015).

Based on the aforementioned definition, transfer learning techniques can be broadly categorised in three main classes (Pan and Yang, 2010; Lu et al., 2015):

(1.) *Inductive transfer learning* – in this case the learning task in the target domain is different from the learning task in the source domain, i.e. $T_s \neq T_t$. The source and target domains however may or may not be the same.

(2.) *Transductive transfer learning* – in this scenario the learning tasks are same in both domains, however the source and target domains are different, i.e. $D_s \neq D_t$ but $T_s = T_t$.

(3.) *Unsupervised transfer learning* – here the target task is different yet related to the source task. The aim is to acquire unsupervised learning tasks in a target domain such as clustering,

dimensionality reduction and more. Inductive transfer learning is the most widely used whereas unsupervised transfer learning is comparatively more recent (Pan and Yang, 2010).

4.3 Research issues in transfer learning

Research in the transfer learning domain identifies four main research issues in the field. These are: (1.) What to transfer? (2.) How to transfer? (3.) When to transfer? and finally (4.) How to assess task relatedness or how to model task similarity? The following subsections discuss these.

4.3.1 What to transfer?

This aspect concerns which part of knowledge is transferrable across domains or tasks. The literature in the field broadly identifies five main aspects that can be transferred. These are:

1. *Literal transfer* – this is the simplest method of transfer which uses the learned or final weights from source neural networks as the initial weights for initialising the target network, which then undergoes training on the target task training set. Although simple, this method has a major drawback, it sometimes interferes with target task learning and even degrades the accuracy achievable on target task –a phenomenon called catastrophic interference (Pratt and Jennings, 1996; Pratt, 1992).

2. *Instance transfer* – this approach is based on the assumption that some parts of the labelled source domain data can be reused in learning the target task. In order to avoid catastrophic interference or any performance degradation in the target task, the approach iteratively reweights the source domain data to maximise similarity between source domain and target domain distribution (Dai et al., 2007a; Dai et al., 2007b; Sugiyama et al., 2008; Jiang and Zhai, 2007a; Liao et al., 2005; Tsuboi et al., 2009).

3. *Feature representation transfer* – these methods aim to find suitable feature representations such that source and target distributions look similar, i.e. to minimise domain divergence (Pan and Yang, 2010; Lu et al., 2015). These approaches can further be divided into two categories: (i) Distribution similarity approaches, which tend to maximise the similarity between source and target domain distribution by penalising features whose

statistics vary between domains (Lu et al., 2015; Arnold et al., 2007; Jiang and Zhai, 2007b; Satpal and Sarawagi, 2007); (ii) Latent feature approaches, which construct new feature representations by using unlabelled source and target domain data (Lu et al., 2015; Blitzer et al., 2008; Ben-David et al., 2010; Huang et al., 2006; Huang and Yates, 2010; Huang and Yates, 2009; Pan et al., 2010).

4. *Parameter transfer* – these approaches discover parameters or priors that can be shared between source and target domains, thereby benefitting transfer (Pan and Yang, 2010; Lawrence and Platt, 2004; Gao et al., 2008).

5. *Relational-knowledge transfer* – these methods assume that data from source and target domains are independent and not identically distributed. In this technique the relationship amongst data in the source domain is transferred to the target domain (Pan and Yang, 2010). Researchers like Mihalkova et al., (2007) and Davis and Domingos, (2009) have proposed methods that can transfer relational knowledge using Markov logic networks across relational domains.

Though techniques 2 – 4 are commonly used, however they work with an *independent and identically distributed (i.i.d)* assumption on the data, whereas approach 5 deals with relational data. This limits the applicability of these approaches to scenarios where the source and target domains are related. If this isn't the case, transfer will not be successful.

4.3.2 How to transfer?

After determining which part of knowledge to transfer, the next important aspect is to develop learning algorithms capable of transferring this knowledge. Literature in the field of computational intelligence broadly categorises all the different approaches for transfer in three broad categories.

i. ***Transfer using artificial neural networks*** – artificial neural networks (ANNs) are computational abstractions of biological information processing systems. Research in machine learning has found several areas where ANNs demonstrate superior performance compared to traditional techniques and statistical methods (Khashei et al., 2012; Ding et al., 2013; Hemanth et al., 2014; Lu et al., 2015; Pratt, 1992; Shavlik et al., 1991; and Thrun et al., 1991). This advantage of neural networks over other techniques has led to their

widespread use in the field of transfer. There are many ways through which ANN-based transfer has been implemented, some of the more commonly used methods include:

Multitask neural networks: these mostly involve feedforward networks trained in supervised mode. This technique was proposed by (Caruana, 1997) and it is an inductive transfer based mechanism with the main aim to improve generalisation performance. In this method all related tasks are trained in parallel using a shared representation. The information contained in related tasks helps improve the performance on target task by acting as an inductive bias (Lu et al., 2015; Caruana, 1997). A modified version of this approach called η MTL was introduced by Silver and Mercer (2001). This method employs a separate learning rate for each task depending on measure of task relatedness between tasks.

Deep neural networks: these networks behave as an intelligent feature extraction module having great flexibility in extracting high-level features in transfer learning. These networks have multiple hidden layers and use unsupervised learning to pre-train each layer, with the output of one layer serving as the input to another. Finally, supervised learning is used to fine-tune all layers (Lu et al., 2015). When ANNs are trained on *related* tasks, sharing deep layers improves features produced by them and hence helps improve generalisation (Collobert and Weston, 2008).

Radial basis function neural networks: These have been used mostly for a category of transfer learning called the covariate shift. This scenario arises when training data are biased towards one region of input space. These networks are mostly used (i) to initialise weights of labelled data in the source domain, (ii) as a pre-processing technique to extract features from high-dimensional space to low-dimensional space, and have also been used (iii) in conjunction with other intelligent methods to improve transfer learning performance (Lu et al., 2015; Liu et al., 2009; Ueki et al., 2010; and Celiberto et al., 2011).

Though all these techniques have been applied successfully, however an issue that still needs addressing is that there is no reliable theory of relatedness that can predict whether or not shared information will be useful.

ii. ***Bayesian Transfer*** - this comprises modelling probability distributions and using conditional independence among variables to simplify the model. These models often have a prior distribution, so given the data the Bayesian model can make prediction by

combining it with the prior distribution to produce a posterior distribution. This is a way to incorporate prior knowledge, which in the case of transfer learning forms the source task knowledge (Torrey and Shavlik, 2009). Some of the commonly used techniques include:

Naïve Bayes: It works under the assumption that there is independence between each pair of features given the class variables. Although this assumption is not valid for most real world applications, nevertheless these classifiers work well for some rather complicated applications such as automatic medical diagnosis, spam filtering, and text categorisation (Lu et al., 2015; Kononenko, 1993; Androutsopoulos et al., 2000; and Sebestiani, 2002). One limitation of this method is that if the test or the target domain data have a different distribution from the source domain data, it becomes difficult to estimate an accurate feature distribution for the new data from old data. This difficulty in estimating the distribution of unlabelled target domain data limits the applicability of naïve bayes in transfer learning scenarios (Lu et al., 2015).

Bayesian network: The ability of Bayesian networks to: handle incomplete datasets, to discover causal relationships hidden within data, to avoid data over-fitting and the ability to integrate domain knowledge and data into one model makes it very suitable for transfer learning. When training data are scarce, using transfer learning with Bayesian networks helps improve their robustness by exploiting data from related tasks. Bayesian network learning was extended from single tasks to multiple tasks by Niculescu-Mezil and Caruana (2007) and Luis et al., (2010) proposed learning models from auxiliary tasks to improve performance on related tasks. To learn multiple tasks, the relationships between tasks are taken into consideration. However, the assumption that all tasks are equally related is the main limitation of this learning algorithm (Lu et al., 2015).

iii. ***Transfer using fuzzy systems*** **–** The fuzzy transfer method consists of two main phases. In phase one, the system uses labelled data which includes, fuzzy concepts and their relationships, from source task to initiate the learning process. These data are then used by the learning process to create a fuzzy inference system (FIS). The second phase involves the adaptation of fuzzy components by utilising knowledge of application context, i.e. the captured knowledge is transferred to the target task (Shell and Coupland, 2015). Transfer learning using fuzzy systems has been applied to many real world problems, for example, Shell and Coupland (2015) and Shell and Coupland (2012), have proposed a framework for

prediction environment in intelligent environments and (Behbood et al., 2011; Behbood et al., 2014) have used fuzzy based transductive transfer learning for long-term bank failure prediction.

There are of course more ways to perform transfer than those listed above, but this discussion has been restricted to widely used and reported methods in literature.

4.3.3 When to transfer?

This involves correctly assessing situations where transfer is feasible and successful. This in turn encompasses the question of *how to avoid negative transfer*? Negative transfer refers to the impairment of current learning and performance due to the application of non-adaptive or unsuitable information. Therefore, negative transfer is a type of interference effect of prior experience causing a slow-down in learning, completion or solving of a new task when compared to the performance of a hypothetical control group with no respective prior experience. One way of avoiding negative transfer is to recognise and reject harmful source task knowledge. Some approaches for doing this have been proposed by (Torrey et al., 2006; Torrey et al., 2005), wherein the authors proposed an approach to reject bad information. An example is the KBKR advice-taking algorithm for transfer in reinforcement learning. This algorithm trades-off between matching the agent's experience and matching the advice, therefore the agent can learn to discount advice that does not matches its experience (Torrey and Shavlik, 2009). Another approach was proposed by (Rosenstein et al., 2005) for detecting negative transfer in naïve bayes classification tasks. Their approach learns a hyperprior for both source and target tasks and the variance of this hyperprior is proportionate to difference between tasks (Torrey and Shavlik, 2009).

4.3.4 How to assess task relatedness or how to model task similarity?

The essence of the concept of transfer relies on the fact that the tasks should be interrelated somehow. If they are not, it would lead to negative transfer. Task relatedness has been studied in cognition under various core headings (refer to Torrey and Shavlik, 2009 and references therein for detail). Some of them are – structural or deep similarity, i.e. similarity based on core underlying features or surface similarity based on general simple description

of objects. Other taxonomies of similarities include a three-level approach which consists of: element similarity, which is based on individual feature elements. Entities are similar at this level if their features overlap. The next level is relational similarity, which is based on relationships that exist between pairs of elements. The final level is called the system similarity. Entities are considered to be similar at this level if they include relations that are in some way related to each other (refer to Torrey and Shavlik, 2009 and references therein for detail).

In computational intelligence, assessing relatedness is a difficult subject but studies show that at least three factors are involved in measuring relatedness. They are – (i) the representational language of the learning system, (ii) the learning (search) algorithm used and (iii) the task domain. In the context of neural networks, the representational language for fixed network architectures is the set of connection weights. Smooth changes in weight representation would equate to smooth variations in task function. Also the distance between weight space representations of two tasks is considered to be a first approximation of relatedness (Silver, 1996). Other techniques include comparing policies, value functions and rewards, but these are only measurable while the target task is being learned, so their use in practical transfer scenario is limited. Other approaches include, graph-based methods where nodes represent source tasks and the distance represents the transferability metric whereas in Kernel methods the learning system learns a meta-kernel that serves as a similarity function between tasks (Torrey and Shavlik, 2009; Carroll and Seppi, 2005; Eaton and Lane, 2008; and Rückert and Kramer, 2008). So although there are various methods available for measuring similarity and they do well in their application to a specific domain or scenario but there are no standards available that can be used effectively for measuring relatedness between any types of tasks, of any domain.

4.4 Heterogeneous Transfer: introduction and issues

Heterogeneous or unrelated tasks are those which vary with respect to their characteristics such as degree of similarity between the input and output patterns, the presence of structure or regularity in mappings and the overall accuracy (Kohli et al., 2013). Most work in the field of transfer learning has focused on improving generalisation by assuming that the feature spaces between source and target domains are the same, an assumption too strong for many practical applications. In many real time applications transferring knowledge across domains and/or

tasks with non-overlapping feature spaces could be helpful, such as for identity-emotion recognition (Pan and Yang, 2010).

Though most research in the field of transfer focuses on homogeneous transfer, there has been considerable research effort in heterogeneous transfer scenarios too, mostly focusing on heterogeneous multitask learning (Romera-Paredes et al., 2012; Argyriou et al., 2008; Zhou et al., 2014b; Zhu et al., 2011; and Wei and Pal, 2011). Most of the methods proposed for heterogeneous transfer focus on learning a common feature representation using the relationship between domains such that the source and target domain data could be characterised by homogeneous features (Zhou et al., 2014b). For example, Shi et al. (2010); Wang and Mahadevan (2011) and Duan et al. (2012) propose methods capable of learning two different feature maps in order to transform source and target domain data to a underlying feature space; authors Zhou et al., (2014a) propose a method capable of learning an asymmetric transformation mapping data from one domain to another; Xue et al. (2015) have proposed a task selection algorithm for heterogeneous tasks in unsupervised transfer learning domain which uses an extended feature method and Dai et al. (2008) proposed a method that can learn mappings by integrating instance correspondences between domains.

Thus it is evident that many efforts have been made towards achieving heterogeneous transfer. Most of these research efforts are successful, however there are some issues outstanding. These include – firstly, most current methods developed for heterogeneous transfer focus only on improving performance on the principal (or target) tasks (Romera-Paredes et al., 2012). Secondly, risk of negative transfer has not yet been eliminated. It is in fact higher since different feature spaces make it difficult to know the similarity between source and target tasks (Wei and Pal, 2011). And finally, the aforementioned approaches are successful, albeit within their limited scope or case-specific applications. However, there is an increasing need for transfer learning techniques used for broader and more challenging applications. This in turn requires having more generalised methods that can be applied on any given set of tasks (Kohli et al., 2013).

4.5 What is next?

So far in this chapter an overview of the current trends of transfer learning has been presented. Research shows that transfer learning, especially when used in conjunction with computational

intelligence methods plays an important role in almost all kinds of applications. However, there are still several research challenges in the field of transfer learning that need to be addressed. First of all, most methods of transfer learning implicitly assume that the source and target tasks are somehow related to each other - when, for example, the source task concerns training on female-only speech whilst the target task is to recognise speech from males only. In addition, most existing transfer learning algorithms assume that the feature spaces between the source and target domains are the same. However, in practice, it is useful to transfer knowledge across domains or tasks that have different feature spaces - the so-called heterogeneous transfer learning (Pan and Yang, 2010). There is no reliable theory of task relatedness that could be used as a benchmark and successfully applied in every scenario. In addition, the transferability among source and target domains needs to be researched deeply so that comprehensive and accurate transferability measures can be implemented that can guarantee that negative transfer does not happen.

Various attempts have been made to overcome the aforementioned challenges. Many are successful, albeit within their limited scope or case-specific applications. However, there is an increasing need for transfer learning techniques used for broader and more challenging applications. This in turn requires having more generalised methods that can be applied on any given set of tasks. Given the observations collected from previous research efforts, in this thesis an attempt has been made to address the aforementioned research challenges.

- First, a neuroevolutionary transfer learning approach which draws inspiration from behavioural genetics has been proposed. This approach uses artificial neural networks (ANNs) as computational models capable of learning various ***heterogeneous*** tasks in an evolutionary framework. These tasks vary with respect to their characteristics such as features, degree of similarity between input-output patterns, the presence of structure or regularity in mappings and overall complexity.

- The proposed method spans transfer learning systems and multi-task learning systems, incorporating "good/useful" features of both, and then combines them with principles of Behavioural Genetics. Table 4.1 explains this further.

– Research in the field of behavioural genetics shows that performance is highly dependent on both the genes and the environment. This work draws an analogy between genes and intrinsic parameters of ANNs, and the training dataset and the environment. This method therefore, imitates more closely learning as it happens in human beings – taking into account both structure and the environment where the learning system is placed. By using same genetic range and environmental proportion for all tasks, this work transfers the ***ability to learn*** across heterogeneous tasks.

With the proposed BG-inspired transfer learning model, we address the following key challenges: to perform heterogeneous transfer, avoid negative transfer, and propose a mechanism for determining task relatedness which extrapolates well to different domains and embodies the effects of both structure/intrinsic parameters and training datasets to which the learning system is applied.

	Learning Goal	**Type of Transfer**	**Degree of task relatedness**	**Means of assessing task relatedness**	**Special features**
Multi task Learning	Improving performance in all tasks	Functional	Highly interrelated	Case-specific relatedness measures	Involves use of shared internal representations such as weights, common data sets for all tasks
Transfer Learning	Improving target task performance	Representational	Related but may be from different domains	Application/case/domain specific measures only but cannot be applied per se on a generalised basis	Highly application/case sensitive
BG inspired Transfer Approach	Improving performance in all tasks	Hybrid: works sequentially like representational and uses common internal representations of intrinsic parameters, like the functional	Can be unrelated or heterogeneous	Heritability and change in heritability, can be used in any scenario	It is based on principles of Behavioural genetics; incorporates shared intrinsic parameters and effects of environment on performance (epigenetics)

Table 4.1: Comparison between the proposed approach and other related approaches

4.6 Extending the BG inspired model to transfer learning

In an attempt to address some of the issues discussed in the previous section, in this thesis a novel neuroevolutionary transfer approach to learn multiple heterogeneous tasks is presented

which draws inspiration from behavioural genetics. We explore the use of a population of artificial neural networks (ANNs) that evolve according to behavioural genetic principles in order to create a computational model capable of transferring knowledge across heterogeneous tasks. This work draws an analogy between genes and the intrinsic parameters of ANNs, and between a combination of training dataset and unique weights for ANNs, and the environment – shared and non-shared, respectively. Table 4.2 provides the high-level description of this transfer learning model. The various phases in transfer learning approach are explained in the sections below.

Initialise: error tolerance threshold, *err_threshold* → user defined value

Maximum number of generations, *maxgen* → user defined value

1. Choose 'x' heterogeneous tasks and identify evolutionary (or source) and learning tasks
2. Simulate variations in genetic influences
    - Calibrate range of variation of each of the chosen intrinsic parameters
    - Encode parameters into genome using fixed predetermined precision i.e. bits
3. Generate initial population
    - Randomly generate 'n' pairs of DZ/MZ genotypes i.e. binary population $G(i)$. Set $i = 0$ as generation counter.
    - Convert 'n' pairs of DZ/MZ genotypes in $G(i)$ into phenotypes i.e. real values, $G_{ph}(i)$ using the genome from step 2
4. Simulate variations in shared environmental influences
    - Randomly generate 'n' SES values within a chosen fixed range, $SES\ (i)$.
5. Create general ability to learn
    - Randomly pair members of $G_{ph}(i)$ with $SES\ (i)$ such that twins have the same $SES\ (i)$ value.
    - Copy these to all tasks, source as well as target
6. REPEAT for each generation
    a. For <u>each task</u>, DO
        i. Generate a filtered training set for each ANN twin pair using $SES\ (i)$
        ii. Simulate non-shared environmental variations: initialise unique ANN weights for EACH ANN in $G_{ph}(i)$

---

iii. Train each ANN using filtered training set generated in step 6.a.i
iv. Evaluate training (tested against full training set) and generalisation performance at end of training.
v. Compute heritability

b. For Source Task only, DO
i. Calculate fitness of each ANN and Select parents from binary population $G(i)$ to breed members of next generation
ii. Apply genetic operators (meiosis and fertilisation) to create next generation of binary genotypes ANN twins, $G(i+1)$

c. Update general ability to learn
i. Convert binary $G(i+1)$ population members into phenotypes $G_{ph}(i+1)$ using same genome from step 2
ii. Repeat step 4 i.e. generate new random 'n' SES values within same range, $SES(i+1)$
iii. Randomly pair members of $G_{ph}(i+1)$ with $SES(i+1)$ and copy to all tasks, go back to step 6.a.i and update generation counter (i.e. replace i with i+1)

7. UNTIL, learning error on source task (and preferably on all tasks) gets reduced to err_threshold **OR** gen == maxgen
8. Swap the source task and repeat steps 2 - 7

Table 4.2: Various phases involved in neuroevolutionary approach for heterogeneous transfer

4.6.1 How to choose tasks – related or heterogeneous

The first step (Table 4.2, Step1) involves identifying source (or evolutionary) and target (learning) tasks. Since we focus on learning unrelated or heterogeneous tasks, we chose n tasks which vary with respect to their characteristics such as the degree of similarity between the input and the output patterns, the presence of structure/regularity in the mappings, and the overall complexity. Therefore, each task posed different requirements to the networks.

As an example suppose, source task (T_S), is the acquisition of English past tense verbs. This task belongs to the past tense domain (D_S). Similarly, let's assume that target task (T_{Tj}) is classifying real world objects into classes such as fruits, veg, mammals, motor vehicles and

more. This task belongs to cognitive categorisation domain (D_{Tj}). Yet another example target task (T_{Tn}) can be auto association or learning to produce same output code that was presented as input. This falls under the domain of cognitive imitation (D_{Tn}).

As per the definitions given in Section 4.2, it can be inferred that $D_S \neq D_{Tj} \neq D_{Tn}$ since these domains have different feature space and probability distributions. Similarly, $T_S \neq T_{Tj} \neq T_{Tn}$ as these have different label spaces and different mapping functions. The aim is to help improve the learning of mapping functions $f_{Tj}(\cdot)$ and $f_{Tn}(\cdot)$ belonging to D_{Tj} and D_{Tn} respectively, using the knowledge gained whilst learning D_S and T_S.

4.6.2 Simulating neurocomputational variation (What to transfer?)

In this work, the effects of genetic influences were simulated via variations in the neuro-computational parameters of the ANNs. ANNs contain an array of parameters that increase or decrease their ability to learn a given training set or the rate at which the learning occurs. These parameters relate to how a network (an individual of the population) is built, its processing dynamics, how it is maintained, how it adapts and how it generates behavioural outputs. These parameters are usually optimised to achieve best learning in a given task. Parameters chosen for encoding could include number of hidden units, learning rate, momentum, slope of the logistic activation function, since these have general computational functions and have no specific relation to the problem domain that ANNs need to acquire, thus making these in synch with our 'generalist genes' hypothesis.

These parameters were encoded in a genome, which was constructed as binary strings of given length. We used binary encoding scheme (although any other encoding scheme such as Gray coding will also work fine) whereby each gene had two alleles with m bits per parameter split into two chromosomes. A noteworthy point is that m has to be an even number since half the information to encode each parameter comes from each parent.

Calibration was carried out (based on source task data) to establish the full range of variation for each parameter over which the artificial neural network exhibited some degree of learning. An initial 'normal' set of parameters was defined. These were projected based on previous research. Each of the continuously valued parameters was then varied in turn, holding all the other parameters at their initial values. For each parameter, the range was derived that produced failure of learning up to highly successful learning. In some cases,

parameters had a monotonic relationship to performance (e.g., hidden units, where more was better); in other cases, there was an optimal intermediate value (e.g., activation function). The aim was to determine an average or adequate value for each parameter, which was defined heuristically as 'just enough to ensure above average performance'. Values were then derived that caused either increasingly poorer or increasingly better performance around this value. We chose to emphasise behavioural symmetry around the average parameter value rather than parametric symmetry, on the grounds that the symmetrical bell curve is a common pattern observed in human abilities. Although only main effects of each parameter were considered as sources of variability during calibration, we expected interactions between these neurocomputational parameters in subsequent learning. An example is that large numbers of hidden units can partially compensate for a shallow sigmoid function in those processing units. Any number of parameters can be chosen to be encoded within this framework as explained in Chapter 2, Section 2.5.2.

4.6.3 How was shared environmental variation implemented? (What to transfer?)

The effects of environmental influences were simulated via a filter applied to the training set. The filter creates a unique subsample of the training set for each simulated individual, based on a parameter determining the quality of the environment. An individual's environmental quality is modeled by a number selected at random from the range [0, 1]. This gives a probability that any given pattern in the full training set would be included in that individual's training set. This filter is applied at each generation to create unique training subsets for all members of the population in that generation. The range we chose was [0.6, 1.0] to define the range of variation of environmental quality, and ensured that all individuals were exposed to more than half of the training dataset (Thomas et al., 2009; Kohli et al., 2013). The process of applying a training set filter has been discussed in more detail in Chapter 2, Section 2.5.3.

This process of filtering training sets is quite similar to data resampling in machine learning. Data resampling is done in machine learning mostly to take into account class imbalance (including data distribution within each class) and to improve classifier performance in general. To counter the effects of class imbalance, research efforts have been made in two directions – the first is data level solutions wherein different forms of data resampling such as random oversampling with replacement, directed oversampling, directed undersampling

or a combination of such techniques are used. The second line of research effort focuses on algorithm level solutions such as adjusting the cost of various learning classes, adjusting the decision threshold, or focusing on recognition based (i.e. learning from only one class) instead of discrimination based (multi class) learning (Chawla et al., 2004).

In addition to these counter class imbalance techniques, there are many commonly used methods for improving classifier performance in general. Some of them include bagging (Breiman, 1996), boosting (Freund and Schapire, 1995) to name a few. Bagging and boosting manipulate training data to generate different classifiers. Bagging generates replica training sets by sampling with replacement from the training instances. Boosting uses all instances at each repetition, but maintains a weight for each instance in the training set that reflects its importance. Adjusting this weight causes the learner to focus on different instances and ergo leads to different classifiers. In either case, multiple classifiers are finally combined mostly by voting to form a composite classifier. In bagging, each component classifier has the same vote whereas boosting assigns different voting strengths to component classifiers on the basis of their accuracy (Quinlan, 1996).

The intent behind training set filtering or resampling done in our BG inspired approach is not to counter the effects of class imbalance but to simulate the levels of cognitive stimulation available to each ANN in the population that could potentially improve classifier performance. Our interpretation of cognitive stimulation relates to the quantity of information (proportion of training data) available and not the quality. Since the filter is applied randomly (though within a fixed range), there could be samples of class imbalances in some cases. However, this is in line with our non-perfect family quotient or SES presumption. Eventually, the aim is that irrespective of what kind of training sample an ANN gets, it should be able to – (a) successfully learn/acquire the given task at an individual level, because this will enhance its chances for being chosen for breeding the next generation, and (b) improve performance at the population level so that this learning ability could be transferred to the subsequent generations. Additionally, the point of training set filtering/variation is that it enables measurement of the impact of computational parameter variation to performance variation, that is, heritability. If all individuals have the same environment, heritability will be fixed at one, because all variation must come from the neurocomputational parameters.

4.6.3.1 Initial weights of ANNs as representatives of non-shared environment (What *not to* transfer)

In this approach, we considered initial values of ANN weights as representatives of unique environments. These values were not encoded in the genome and each network had unique values for initial weights for each task, thereby representing the unique or non-shared environmental influence within the perspective of given task. Chapter 2, Section 2.5.3.2, provides details on implementing non-shared environmental variations.

4.6.4 Role of using twins population (Determining task relatedness and avoiding negative transfer)

Our approach uses a population of twins (ANNs with some degree of similarity in their neuro-computational parameters) to disentangle genetic and environmental influences on performance. This approach is inspired by cognitive development, where twins are more closely matched for age, family and other social influences. This is because twins are either genetically identical (genetic relatedness of 1.0 for *mono-zygotic,* MZ, or identical twins) or as similar as siblings (genetic relatedness of 0.5 for *di-zygotic,* DZ, or fraternal twins) and, to an approximation, share the same environment (applicable for both MZ and DZ twins based on the *Equal Environment assumption*) (Plomin and Spinath, 2004). The difference in the similarity in performance between MZ or DZ twin pairs, along with assumptions about their similarity of environment, allows inferences to be drawn about the influence of genetic relatedness on behaviour (Plomin et al., 2008).

The genomes were created (as explained in Section 4.6.2) sharing 50% of their values on average, thereby simulating DZ twins, and identical genomes were used to simulate MZ twins. The twin population of *x* individual ANNs was created by simulating the biological processes of meiosis and fertilisation as explained in Chapter 2, Section 2.5.4. The population should consist of $\frac{x}{2}$ MZ network pairs and $\frac{x}{2}$ DZ network pairs.

From a computational point of view, in particular we exploit the notion of *heritability* within Behavioural Genetics to assess task relatedness. Heritability is a statistic that describes the effect size of genetic influence and refers to the proportion of observed or phenotypic variance that can be explained by genetic variance. In simpler terms, it is the amount of population variability explained by genetic similarity (Plomin et al., 2008). In

computational terms, heritability can be interpreted as the amount of performance variation accounted for by structural similarity. Twin studies provide an exact computation of heritability. Additionally, twin studies provide a valuable tool for exploring environmental influences, especially family or shared environment, against a background of heritability. Since twins are genetically similar, if heritability affects behaviour then MZ twins will be more similar than the DZ twins. In other words, heritability (with twin studies) provides an estimate of the magnitude of genetic influence on behaviour (Plomin and Spinath, 2004).

Heritability is an integral part of the current work for the following reasons. As a population is bred and optimised across generations on a particular task, the range of variation of its *suitable or relevant* computational (or intrinsic) parameters reduces, i.e. optimisation leads to reduction in heritability. If the range of environmental variation is kept the same, the variation in performance will be more due to environmental variation, since the optimised population will now be more genetically homogeneous. Now, consider if the same population were trained on another non-related or heterogeneous task, and this also experienced a reduction in heritability. This would be an indication of the presence of some kind of relatedness among the tasks. The *direction of the change in heritability indicates task relatedness*. If the change in heritability for different tasks is in the same direction (e.g. all values decrease or all increase proportionally), this implies that the same set of intrinsic parameters are appropriate for learning the tasks. This in turn can help in identifying a set of *domain-relevant* parameters, which, like generalist genes, are useful for learning various heterogeneous tasks. Thus, change in heritability has the potential to act as a mechanism for identifying task relatedness, which extrapolates to different task domains, and consequently avoids negative transfer (Kohli et al., 2013).

4.6.5 Implementation of transfer approach (How to transfer?)

Table 4.2 depicts the implementation scheme of this BG inspired transfer approach. The main steps involved in implementation are explained below:

i. *Choose Tasks:* the first step is to identify x number of tasks (heterogeneous or related). Choose any one task as the source task (T_S) and the remaining $(x-1)$ tasks become target tasks (T_T). The aim is to successfully learn (i.e. improve performance) on all x tasks.

ii. *Encode parameters and calibrate range:* next, encode the neuro-computational parameters of artificial neural networks in a genome. This stipulates the range of variation for all neuro-computational parameters and enables each member of the population to have a different set of values, but within the same fixed chosen range for the encoded parameters ensuring genetic diversity. The range of variation is calibrated with respect to help learning the source task. Chapter 2 Section 2.5.2 provides more details.

iii. *Generate ANN twins' population:* next, a binary (i.e. genotypes) population, $G(i)$ of n pairs of MZ twins and n pairs of DZ twins is created by simulating the biological processes of meiosis and fertilisation. The binary population is then converted into real-valued i.e. phenotypes of ANN twins, $G_{ph}(i)$ using genome from step 2. As we progress with the generations, only offspring are included in the new generation populations. The binary population is converted into real-values i.e. phenotypes using the values from Genome from step 2. The twins are split into two groups – breeding and nonbreeding (as explained in Chapter 2, Section 2.5.4).

iv. *Simulate shared-environmental variations*: an individual's environmental quality is modeled by a number selected at random from a given fixed range. This gives a probability that any given pattern in the full training set would be included in that individual's training set. Therefore, randomly generate 'n' SES-values within a chosen fixed range, $SES(i)$.

v. *Generate 'general ability to learn':* in this step, the ANN phenotypes, $G_{ph}(i)$ are randomly paired with $SES(i)$ values, ensuring that twin pairs have the same SES values. Therefore, each population member is now characterised by its own intrinsic values and training set filter values. These values are then copied across all tasks – source as well as target.

vi. *Apply filter to training data:* subsequently for each task, implement environmental variability as a filter applied to the training tasks using $SES(i)$ values. The filter creates a unique subsample of the training set for each simulated individual, based on a parameter determining the quality of the environment. This filter is applied at each generation to create unique training subsets for all members of the population

in that generation. Due to the *equal environment assumption*, twin pairs have the same training subset. Refer to Chapter 2, Section 2.5.3, for further details.

vii. *Simulate non-shared environmental variations*: for each task, initialise unique weights for each ANN in current population.

viii. *Train:* the population of twins, breeding and nonbreeding are then trained on the source task and independently on each of the target task using their filtered training sets created for each task in step 6.a.i of Table 4.2, using any learning algorithm.

ix. *Performance assessment & heritability computation:* then on completion of training, performance on each task is assessed independently on their respective full training set and previously unseen generalisation set. Additionally heritability is also computed for each task independently. Refer to discussion in Chapter 2, Section 2.5.5, for more details.

x. *Select & breed next generation members:* this step is valid for the source task only. Based on the performance of the population of networks on source task, members are *selected* from breeding ANN twins only for breeding the next generation. To this end, a selection metric is applied at the end of training. The selected members enter the breeding pool and then breed with randomly chosen members from that pool. After selection, only the offspring form the next generation, $G(i+1)$ of populations- parents (or members of previous/breeding populations) are discarded. Chapter 2, Section 2.5.6, discusses the said issue in greater detail.

xi. *Update 'general ability to learn':* here, the binary members of $G(i+1)$ are converted into phenotypes, $G_{ph}(i+1)$. Next, step 4 of Table 4.2 is repeated to generate new random n pairs of SES values, $SES(i+1)$. The values in $G_{ph}(i+1)$ and $SES(i+1)$ are randomly paired in order to update the ability to learn in accordance with the Darwinian inheritance concept.

xii. *Repeat:* steps vi-xi (of current section 4.6.5) are then repeated until the learning error on the source (and preferably all tasks) reaches a pre-determined error threshold value OR until maximum number of generations is reached.

Then the source task can be swapped with any one of the target tasks and the process can be restarted from step 1.

4.6.6 Factors affecting transfer of 'ability to learn'

The interaction between quality of environment (i.e. filtered training set) and good (or not-so-good) genes (i.e. encoded ANN parameters) gives networks the *ability to learn* a given task. Thus using the same quality of training set and same neuro-computational parameters leads to transfer of *ability to learn* across different tasks. This idea of transferring the ability to learn draws inspiration from the concept of generalist genes (Kovas and Plomin, 2007; Plomin and Kovas, 2005). The assumption is that the neurocomputational parameter range that works for one task can potentially suit other non-related tasks as well, as long as chosen parameters have general computational functions. Therefore care must be taken to choose parameters that are general i.e. parameters that provide ANNs or any learning system with an ability to learn and can accelerate the speed of learning. In theory this approach should work, however there are two important factors within this approach that modulate the reliability of transfer. These are – a) type of selection operator used and b) nature of chosen source task. In the subsequent paragraphs we discuss how these factors could potentially affect reliability of transfer.

i. *Type of selection operator*: The two more commonly observable natural selections are stabilising selection and directional selection. In the former case, extreme varieties from both ends of the frequency distribution/range of variation of a trait are eliminated (Darwin, 1897). Examples of stabilising selection include, average birth weight of human babies (3.5 kg), number of eggs robins lay (which is always four) and many more. In terms of machine learning, stabilising selection is equivalent to stochastic selection metrics like the roulette wheel selection. Such methods display no bias towards a specific range of fitness or parameter values and ergo there is no apparent shift/preference towards a given set of parameters (or range of variation) across generations. With these selection methods, there is a chance that some weaker (or less fit) population members may survive the selection process, although this could be advantageous, since a 'weak' ANN may include

some properties which could prove useful following the recombination process. However, the downside is that this type of selection might put a constraint on the learnability (or the accuracy levels achieved) of population members especially if not so fit members get selected often.

On the other hand, in case of directional selection, an extreme phenotype is preferred over other phenotypes, causing the allele frequency to shift over time in the direction of that phenotype (Darwin, 1897). Under directional selection, the beneficial allele increases as a result of differences in survival and reproduction amid different phenotypes. Evolution usually opts for directional selection when there is change in environment. A classic example of this type of selection is the evolution of the peppered moth in 18^{th} and 19^{th} - century England. Before the Industrial Revolution, peppered moths were predominately light in colour, which allowed them to blend in with the light coloured trees in their environment. However as soot began spewing from factories, the trees darkened and the light-coloured moths became easier for predatory birds to locate. However, over time the frequency of the melanic form of the moth increased because darker coloration provided good camouflage against the sooty tree and consequently they had a higher survival rate in habitats affected by air pollution (Majerus, 2009). Another example is the fossil records that show that the size of the black bears in Europe decreased during interglacial periods of the ice ages, but increased during each glacial period.

In machine learning, directional selection corresponds to a deterministic selection mechanism wherein only the fittest individuals get a chance to reproduce. An example is truncation selection in which the candidate solutions (or population members) are ordered by fitness, and some proportion, p, (e.g. $p = 1/2, 1/3$ ...), of the fittest individuals are selected for breeding next generation.

It is evident from the discussion above that using selection mechanisms belonging to aforementioned two categories would lead to populations with completely different overt behaviours (or learning abilities and accuracy levels). This in turn might have direct impact on the effectiveness of transfer.

ii. *Nature of source task*: The benefits/effectiveness of transfer, to a large extent, also depends on the type/nature of source task especially if the chosen tasks are heterogeneous. In that case each learning task will pose different requirements to the network depending on its properties such as the mapping function, the overall complexity and so on. Therefore there is a possibility that neurocomputational parameters (or their range of variation) that a given source task favours are domain specific i.e. work for only that particular task/problem domain. This might lead to negative transfer. However, the parameters (or their range of variation) could also turn out to be domain relevant, which implies that they are suited for different types of task/problem domains even though they aren't being specifically targeted/chosen for said tasks. Therefore, type of source task could also potentially lead to very different results with regards to the success of transfer.

In our experiments (presented later in the thesis) we have explored both these branches to evaluate how each of these factors modulates reliability of transfer.

4.7 Summary and contribution of the chapter

In this chapter, we first presented a review of current trends in the field of transfer learning. We then discussed the four main research issues in transfer learning – what to transfer, how to transfer, when to transfer and how to assess task relatedness. Further we discussed the existing challenges in performing heterogeneous transfer. Based on the observations collected from previous research efforts, we then presented our BG inspired transfer learning approach. Our method utilises ANNs as computational models capable of learning various tasks (related and heterogeneous) in an evolutionary framework. The following is the summary of the main features of our transfer approach.

– Our transfer method incorporates useful features from multi task learning and conventional transfer learning methods. Therefore, it enables optimising performance (or learning) on multiple tasks independently and even at different points in time.

– In this method we draw an analogy between on the one hand genes and intrinsic parameters of ANNs, and on the other environment and training datasets. Therefore, our method imitates learning as it happens in humans more closely.

– Inspired by the generalist genes hypothesis, our transfer approach chooses intrinsic parameters that have general computational functions and are not specific to the problem/task domain that the learning system has to acquire. This enables it to find domain relevant parameters and range(s) which could potentially suit unrelated tasks.

– Thus using the same quality of training set and same neuro-computational parameters for all learning tasks, leads to transfer of *ability to learn* across different tasks.

– It uses population of ANN twins and exploits the notion of heritability to assess task relatedness. Twin studies provide an exact computation of heritability and as discussed in Section 4.6.4, direction of change in heritability has the potential to act as a mechanism for identifying task relatedness, which extrapolates to different task domains, and consequently avoids negative transfer.

– Finally, by incorporating the aforementioned features in our transfer model, we aim to address the following key challenges: to perform heterogeneous transfer, avoid negative transfer, and propose a mechanism for determining task relatedness which extrapolates well to different domains and embodies the effects of both structure/intrinsic parameters and training datasets within which the learning system is placed.

We also identified two key factors that could potentially modulate the performance of our model – selection operator and nature of source task. In the next chapter, we use the BG inspired transfer approach to learn different heterogeneous tasks. We shall be experimenting with different selection operators and swapping source tasks in order to test reliability of our model.

Chapter 5 Experimental evaluation of BG inspired Transfer Learning framework: selection operators and impact on transfer

5.1 Overview

This chapter presents the experimental evaluation of the BG inspired transfer framework. We identified two important factors that could potentially modulate the performance of the transfer model – type of selection operator and nature of source task. Thus experiments were designed to take into account effects of both of these factors separately. This chapter focuses on the type of selection operator and its effect on transfer. This chapter is organised as follows: Section 5.2 explains the various tasks chosen for testing transfer and Section 5.3 describes the datasets for each of the tasks. Section 5.4 presents the experiment design and Sections 5.5 and 5.6 discuss the results based on the effects of type of selection operator. The discussion is presented in Section 5.7 and finally summary and contribution of chapter in Section 5.8.

5.2 The Heterogeneous Tasks

To investigate the robustness and effectiveness of the proposed approach, five tasks were chosen which vary with respect to their characteristics. These included the degree of similarity between the input and output patterns, the presence of structure/regularity in the mappings, and the overall complexity. Therefore, each task posed different requirements to the networks and each of these tasks was representative of a different cognitive ability of human beings. The tasks chosen were:

– **Modelling performance of 6-year-old children on English past tense (verbs) acquisition**: this task is representative of one aspect of language acquisition in human beings. The ANNs were required to learn the correct mappings between the English verb and its past tense. Given the phonological code of a verb stem presented in the input, the networks had to learn to output the phonological code of its past tense form.

– **Auto-association**: this task is representative of cognitive imitation in human beings. Cognitive imitation combines imitation and observational learning (Subiaul et al., 2004).

Imitation is broadly understood to be a powerful way to learn. Research has shown imitation by new borns and toddlers for adult facial expressions, tongue protrusions to name a few (Meltzoff and Moore, 1999). Similarly in this task, the networks were required to learn to output the code presented in the input layer.

– **Arbitrary-mappings**: in terms of cognition, this task can be described by one of the features of human language wherein the relationship between the sound of the word and its meaning is completely arbitrary. Given the sound of an unknown word, it is next to impossible to infer its meaning. This form of mapping is extremely hard since arbitrariness of word sound–meaning mappings introduces a cost for learning. This occurs because the mapping between the word form and its referent has to be formed afresh for each word, and having prior knowledge of all the other words in the vocabulary does not assist in learning a new word (de Saussure 1916; Hockett, 1960). Thus in this task the networks have to learn mappings between random inputs and random output patterns.

– **Categorisation**: This can be viewed as the process of grouping things based on prototypes. Categorisation in humans is achieved by recompiling a frugal set of rules on presentation of each stimulus. These rules are created from examples retrieved from long term memory based on similarity to prototypes (Aisbett and Gibbon, 1999). The new entity is assigned a category which is closest to the prototype by using the logical information provided by frugal rule set and prior knowledge obtained from long term memory. Similar to cognitive categorisation, in this task the networks have to learn to assign input patterns to different categories based on their similarity to a prototype pattern for each category.

– **Categorisation with Exceptions**: Principally this task is very similar to normal categorisation task. However there are some entities (patterns) which are exceptions to the prototype theory. These entities acquire membership in a particular category only through extension. As an example consider tomatoes, avocadoes, courgettes, all of which are categorised as vegetables although formally these all are fruits. These are classified as vegetables because their ‘fruit-like’ properties like sweetness, acid level and more are in conflict with typical members of fruit category like mangoes, bananas and apples (Langacker, 1987). Therefore in this task, the networks have to learn to assign input patterns to different categories based on their similarity to a prototype pattern for each category.

However, a small set of input patterns are exceptions to this rule. Based on some chosen condition, these exceptional patterns should be assigned to a category which is different from the one corresponding to the more similar prototype pattern.

5.3 Dataset Description

The datasets are summarised in Table 5.1. For each of the five tasks there were two datasets: one was used for training and the other one was used for calculating the generalisation accuracy. For this instantiation of the framework, all training and generalisation/test datasets had 57 bit inputs and 62 bit outputs representing different types of features from the five tasks. Although all five datasets have same number of bits in their inputs and outputs, the correlation between these datasets was very close to zero or negative, thereby affirming that the chosen tasks were in fact heterogeneous i.e. had different feature spaces and different mapping functions. The dataset for each task is explained below.

- **English Past Tense**: The dataset was based on the "phone" vocabulary from the Plunkett and Marchman (1991) past-tense model. The past tense domain is modelled by an artificial language created to capture many of the important aspects of the English language, while retaining greater experimental control over the similarity structure of the domain (Plunkett and Marchman, 1991).

  The dataset comprised of artificial verbs which in effect were artificial monosyllabic phoneme strings that followed one of three templates – CCV, VCC, and CVC (where C → consonant and V → vowel). There were 508 verbs in the dataset. Each verb had three phonemes – initial, middle, and final. The phonemes were represented over 19 binary features using an encoding based on linguistic articulatory features (Thomas and Karmiloff-Smith, 2003). A network thus had 3×19 = 57 input units and 3×19 + 5 = 62 units at the output. The extra five units in the output layer were used for representing the affix for regular verbs in binary format.

  In the training dataset, there are 410 regular and 98 irregular verbs. These were further divided into four types: regular verbs that form their past tense by adding */ed/* - e.g. talk – talked; regular verbs which form past tense by adding */d/* - e.g. tame – tamed, regular verbs which suffix */t/* - e.g. send – sent and finally the irregular verbs, e.g. hide – hid or go – went. In the dataset, out of 410 regulars, there were 271 */ed/* verbs,

90 */d/* verbs, 49 */t/* verbs. As this was an imbalanced dataset generating a classifier is challenging, as the classifier tends to map/label every pattern with the majority class. A second dataset was used to assess the generalisation performance of the model. The main intent was to measure the degree to which an ANN can reproduce in the output layer properly inflected novel items presented in the input. The generalisation set comprises 508 novel verbs, each of which shares at least two phonemes with one of the regular verbs in the training set, for example *wug – wugged* (Thomas et al., 2009). The generalisation dataset consisted of verb stems with differing degrees of similarity to the verb stem of training set. Three different degrees of similarity were used to create generalisation dataset. In first case, the first phoneme of the training set verb stem was changed, in second case, first two phonemes of verb stems were changed. Both of these changes were however consistent with the rules of what phoneme strings are legal within a language or phonotactics, i.e. a C was replaced by another C and a V by another V. In third case, however the first two phonemes were changed such that the conformity to phonotactics was violated.

- **Auto-association**: The input patterns of the training set for this task were random vectors of 57 binary digits. The target patterns were same as the input patterns. As the architecture of the network had 62 units in the output layer, the targets were presented in the 57 leftmost units of the output layer. The remaining five units had zero values for all mappings. There were 500 patterns in the training dataset.
A generalisation dataset was also used, in order to assess the degree to which the network could reproduce in the output layer novel patterns presented in the input. The patterns of the generalisation set were constructed by altering probabilistically the patterns of the training set. Each bit of input pattern was flipped (from 1 to 0 or vice versa) with a probability of 0.2. If the resulting pattern was not novel the procedure was repeated. There were 500 patterns in the generalisation dataset.

- **Arbitrary-association**: The training set for arbitrary mappings task had the same input patterns as the training set for auto-association task. As explained previously, these mappings were random vectors of 57 binary digits. However, for this task the targets were not identical to the input patterns. Instead, they were random vectors of 62 binary digits, each one corresponding to one of the output units. Finally,

because the mappings were random, there was no underlying systematic function. Therefore, there was no generalisation dataset to test the extension of the function. There were 500 patterns in the dataset.

- **Categorisation**: The training set for the categorisation task considered the assignment of input patterns to ten categories, based on their similarity to a prototype pattern for each category. The training set was created by first defining the ten prototype patterns for each category. These were ten random vectors of 57 binary digits. Next, clusters of input patterns, which were similar to the prototypes were generated. Each input pattern was created by altering each bit of corresponding prototype pattern with a probability of 0.05. Any duplicates produced were discarded. To encode categories in the output layer of the network architecture, the 60 leftmost output units were used. Ten groups of six units were considered, each group corresponding to one category. When an input pattern belonged to a certain category the units of the corresponding group of output units had the value of 1. Training set contained 500 patterns.

  A generalisation dataset was also used to evaluate the ability of the networks to categorise novel items based on their similarity to the prototype patterns. The input patterns of the generalisation set were constructed with the same procedure followed for training set generation. Bits of the prototype vectors were altered probabilistically, with any resulting duplicates being removed from generalisation dataset. Generalisation dataset had 500 patterns.

- **Categorisation with Exceptions**: The training set for this task used the same input patterns as the categorisation training set. However, this version corresponded to a slightly more complex categorisation problem, which additionally considered a sub-cluster of exceptions. This sub-cluster consisted of all input patterns of category 9 whose Euclidean distance from the prototype element was less than 2. The sub-cluster of these input patterns should be assigned to category 7, instead of category 9. The generalisation set was implemented for this task with the same methods used for categorisation task but which also included the sub-cluster of exceptions. There were 500 patterns in training and generalisation dataset respectively and 10 patterns belonged to the sub-cluster.

Table 5.1 summarises the datasets used for the experimental evaluation.

Tasks	Input Bits	Output Bits	Description of data set
Modelling performance of 6-year-old children on English Past Tense	57	62	a) The training set consists of 508 English past tense verbs type frequency of verbs: 410 – regular, 20 – identical, 68 – vowel change, 02 –arbitrary b) 8 arbitrary (non-English) verbs for ensuring finer graduations of performance. c) Separate test set consists of 508 novel verbs
Auto-association	57	62	a) Training set consists of 500 patterns and Target patterns same as input patterns b) ANNs produce 62-bit output vectors (62 output nodes) but the last 5 bits get zero values for all mappings. c) Separate test set consists of 500 novel patterns.
Arbitrary-Association	57	62	a) Training set consists of 500 patterns; targets are not same as the inputs b) No generalisation set since random inputs have random outputs.
Consistent Categorisation	57	62	a) Training set consists of 500 patterns belonging to 10 categories b) Each pattern is assigned a category based on its similarity to the prototype pattern of each category. c) Each pattern is created by altering each bit of corresponding prototype pattern with a probability of 0.05. d) Test set consisting of 500 novel patterns using the same procedure used for training set.
Categorisation with exceptions	57	62	a) Training set consists of 500 patterns where same input patterns as in previous categorisation data set are used. b) Slight modification in the mappings. Includes a sub cluster of exceptions. c) This sub cluster consists of all input patterns of category 9 whose Euclidean distance from prototype element of the category is less than 2. d) These patterns are assigned category 7, instead of 9. 10 patterns in sub-cluster Test set consisting of 500 novel patterns using the same procedure used for training set.

Table 5.1: Summary of Datasets used

5.4 Experiment Design

The previous chapter established that the behaviour of the transfer model was potentially modulated by type of selection operator and nature of source task. In order to test this hypothesis, we explored performance of the model in different lineages, i.e. combinations of genetic and environmental influences. Overall ten replications of the model were tested, each with a twenty-generation duration. The experiments were conducted on Condor, which is a platform that supports running high throughput computing on large collections of distributive owned computing resources (Thain et al., 2005). It follows a master-slave type configuration, which has proved suitable for training neural network architectures (Plagianakos et al., 2006).

Each scenario was characterised by its own initial population (produced with random binary genomes) and unique values for the other heuristics involved, such as initial weights. The evolutionary methodology was then applied to each of these model instantiations, such that they all shared the same range of variation for genetic and shared environmental influences. At the same time, however, they were unique, for each of them began with a different initial population of 100 networks created from random binary genomes. Thus, having ten replications (*r1*, *r*2… *r10*) of the model aided in evaluating the robustness of the method.

The first six replications were dedicated to investigating the effects of selection operator on the behaviour of transfer model. This was roulette wheel (RW) selection for replications 1, 2 and 3 and truncation selection for replications 4, 5 and 6; the source task, acquisition of English past tense was kept same for all 6 replications. This chapter is dedicated to the experiments and results involving the first six replications and thereby focussing on effects of selection operators and their impact on transfer.

In order to take into consideration the stochastic effects of RW selection, the networks in replications 1 – 3 had a longer training period (of 1000 epochs). The intent was to give enough training exposure to any possible not-so-good networks chosen due to the stochastic nature of RW selection. On the other hand, in the remaining three replications, the training period was much shorter and was kept flexible, since only the fittest networks were being chosen for breeding.

For each generation, 50 pairs of DZ and 50 pairs of MZ twins were created with their computational parameters encoded into a genome. These were split in breeding and non-breeding sets, where the former was the population containing the 1st twin out of each of the twin pairs (100 networks) and the latter was the population containing the remaining 2nd twin of a twin pair (100 networks). These were instantiated as three-layered feed-forward networks and were trained using the batch version of the Rprop algorithm. The stopping condition was an error goal (mean squared error) of 10^{-5} within 1000 epochs (or max. epochs specified). The networks were trained on the filtered training sets, but performance was always assessed on the full training set and then tested on the previously unseen generalisation set. The filter applied was based on SES values of each twin pair. These values represent the probability of including a particular data point (or training pattern) of the full training set into an individual's filtered training set. This varied between 60% and 100% so that each individual would come across at least half of the training set. Twin pairs had the same filtered training set. In order to

breed twins, different crossover operators were employed like single point, multi-point and more.

ANNs were designed with neuro-computational parameters encoded into the genome to constrain their learning abilities. For this work, three free parameters were selected, each of which corresponded to how the network was built. These were (i) number of hidden units; (ii) its activation dynamics, i.e. slope of logistic function; and (iii) how it adapted, i.e. learning rate, or the initial learning rate of Rprop. The range of variation of each of these parameters was calibrated to avoid the presence of genes in the population that produced networks with no learning ability. To this end, work began with random values for all parameters and trained 100 neural networks for 1000 epochs while varying the values, in steps of 5 for hidden units and 0.01 otherwise, for each of these parameters individually. The networks were trained on English past tense task due to its quasi-regular nature. The calibration process was carried out for all parameters, until values were identified beyond which the learning failed, as well as the values which resulted in increasingly successful learning. For the encoding, binary representation was used, whereby each gene had two variants or alleles, with 10 bits per parameter, split into two chromosomes. Chapter 2, Section 2.5.2, especially Table 2.2 gives the details for the same.

The experimental settings are summarised in Tables 5.3 and 5.4 in respective sections. In the following experiments, populations comprising over 120,000 neural networks in total were trained on five different tasks.

5.4.1 How was behaviour (performance) measured?

The populations of twin ANNs were trained on the filtered training dataset using the Rprop algorithm (Riedmiller and Braun, 1993). The performance was assessed on the full training set, as well as on another novel dataset that was created to test the generalisation ability of the networks (see Subsection 5.3). First, the continuous outputs produced by networks were converted to binary by applying threshold. Then the performance was assessed using recognition accuracy based on Hamming distance, for English past tense (refer to Chapter 3, Table 3.1) and auto-association and arbitrary-association tasks as explained in Table 5.2.

Input:	Actual output of network, $\boldsymbol{Y_n}$
	Desired output, $\boldsymbol{Y_d}$
Output:	Performance accuracy, A
Variables:	$I \rightarrow$ total number of patterns in $\boldsymbol{Y_n}$
	$J \rightarrow$ total number of patterns in $\boldsymbol{Y_d}$
	$P_i \rightarrow$ a pattern in $\boldsymbol{Y_n}$, where $, i < I$
	$P_j \rightarrow$ a pattern in $\boldsymbol{Y}_d$, where $, j < J$
	$h_{dist} \rightarrow$ hamming distance between phonemes of P_i and P_j
	$corr \rightarrow$ No. of correctly produced outputs

1. **initialise** $\boldsymbol{corr = 0}$
2. **for** $(\boldsymbol{i = 1: I; i < I; i++})$ **Repeat**
3. **for** $(\boldsymbol{j = 1: J; j < J; j++})$ **do**
4. Calculate $\boldsymbol{h_{dist}}$ between corresponding phonemes of $\boldsymbol{P_i}$ and $\boldsymbol{P_j}$
5. **If** $\boldsymbol{h_{dist} == 0}$, **do**
6. $\boldsymbol{corr = corr + 1}$;
7. Break;
8. **end**
9. **end**
10. **end**
11. $\boldsymbol{A = \left(\frac{corr}{I}\right) * 100}$;
12. **Return** $\boldsymbol{A}$

Table 5.2: Algorithm for calculating performance accuracy

For computing performance accuracy achieved in categorisation and categorisation with exceptions tasks, we used Matlab' built-in function called 'confusion', which is the classification confusion matrix. Its syntax is –

$$[c, cm, ind, per] = confusion(targets, outputs)$$

and it takes the following values,

$targets \rightarrow S - by - Q$ matrix, where each column vector contains a single 1 value, with all other elements 0. The index of the 1 indicates which of S categories that vector represents.

$output \rightarrow S - by - Q$ matrix, where each column contains values in the range [0,1]. The index of the largest element in the column indicates which of S categories that vector represents.

As the output this function returns the following,

$c \rightarrow$ Confusion value = fraction of samples misclassified

$cm \rightarrow S - by - S$ confusion matrix, where $cm(i,j)$ is the number of samples whose target is the i^{th} class that was classified as j

$ind \rightarrow S - by - S$ cell array, where $ind\{i,j\}$ contains the indices of samples with the i^{th} target class, but j^{th} output class

$per \rightarrow S - by - 4$ matrix, where each row summarizes four percentages associated with the i^{th} class:

$$per(i,1) false\ negative\ rate = \frac{false\ negatives}{(all\ output\ negatives)}$$

$$per(i,2) false\ positive\ rate = \frac{\text{false positives}}{(\text{all output positives})}$$

$$per(i,3) true\ positive\ rate = \frac{\text{true positives}}{(\text{all output positives})}$$

$$per(i,4) true\ negative\ rate = \frac{\text{true negatives}}{(\text{all output negatives})}$$

In the following sections, we present the results and analysis of results in various experimental settings.

5.5 Results and Analysis – roulette wheel selection

The results reported in this section follow three lineages each with a twenty generation duration that were increasingly optimised on the English past tense task using a stochastic roulette-wheel selection operator. The change in performance was traced across generations on this task, and the change in heritability; but also, crucially, the same measures are reported when each succeeding past-tense-optimised generation is instead trained on the other four target

tasks. Table 5.3 summarises the experiment design for lineages under this setting, i.e. replications 1, 2 and 3 (denoted by 'R').

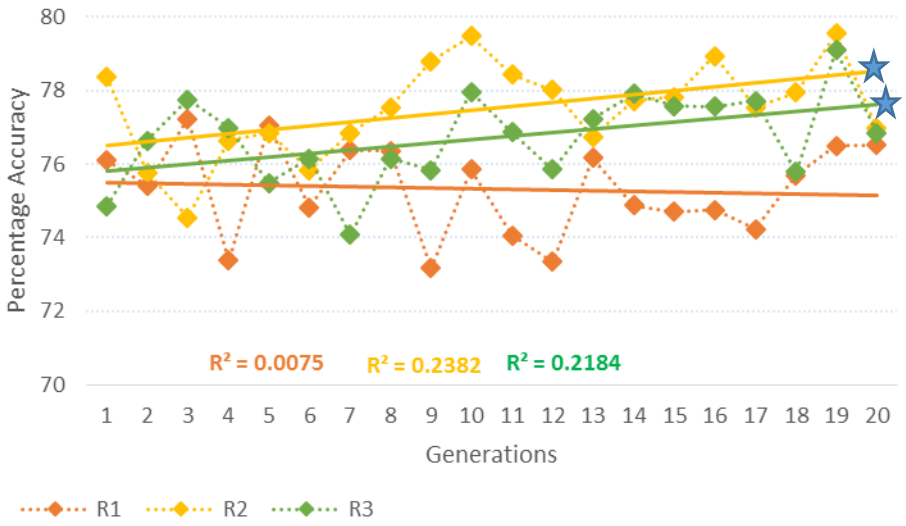
No. of replications	3 (R_1 – R_3)
No of Generations per replication	20
Size of population	Breeding = 100; Non-breeding= 100 Total R_1+R_2+R_3 across generations= 12,000 ANNs per task
Size of Datasets	Training= 500 { 508(for past tense)} Generalisation= 500
Training Mode	Batch
Max. training epochs	1000
Initial weight update (Rprop learning rate)	Values from genome
Hidden units, Steepness of logistic	Values from genome
Selection Operator	Roulette Wheel - applied at the end of training (1000 epochs)
Crossover	6 crossovers/chromosome; single-point, multi-point & shuffle operators used
Environmental Factor (SES)	Probability value between 60% and 100%
Range of encoded neurocomputational parameters	No. of hidden units (10 – 500); initial learning rate (0.7 – 1.0); slope of logistic activation (0.0625 – 4.0)

Table 5.3: Experimental Design for RWS based replications

The mean accuracy levels achieved on the source task centred around 75%. This accuracy could be construed as average from machine learning perspective, however this accuracy level is within the acceptable range of expected past tense results for the age groups between 5-7 year olds (refer to Chapter 3, Section 3.6.1) in all three replications. For the target tasks, categorisation and categorisation with exceptions, had very high accuracy levels, above 90% in all three lineages. In initial generations, the populations started off with good mean accuracy levels of about 85% on auto association and 40% for arbitrary mappings tasks, however these experienced gradual decline across generations, a trend that replicated itself in all three lineages. Generalisation accuracy followed similar trends at lower accurate levels.

This setting is characterised by a stochastic selection operator acting on a quasi-regular source task, a combination which resulted in some thought-provoking results. Figures 5.1 (a) –(e) and 5.2 (a) – (d) show the overall performance accuracy on the full training set and test/generalisation set for all five tasks. Each of these graphs summarise the results from 12,000 networks. A zigzagged line indicates the mean accuracy level of the 100 networks for each

population at each generation, while a straight line represents the general trend observed in that replication scenario across generations. The trend line was derived from a linear regression line based on the least squares method, predicting mean performance level from generation number. Regression analysis was used to determine individually reliable trend lines at .05 level, shown in graphs below with a blue star (★) either next to the gradient or near the corresponding legend. In some cases, R^2 values were relatively small, reflecting the non-monotonic changes in performance over generations.



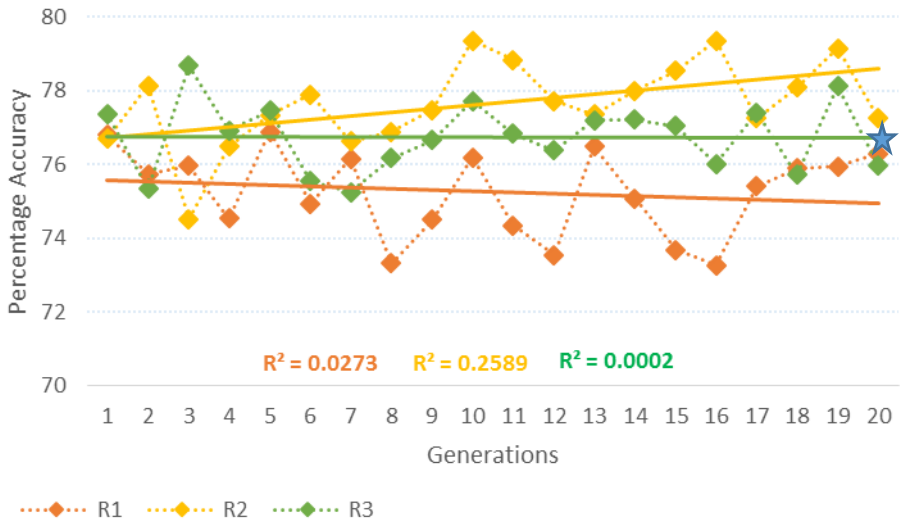
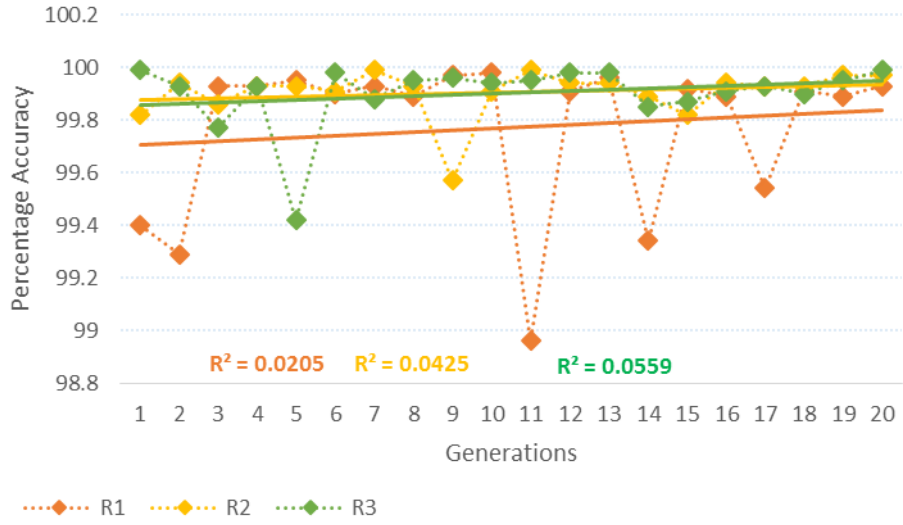


Fig. 5.1 (a): Mean performance per generation for breeding (left) and non-breeding (right) twin populations on English past tense acquisition task



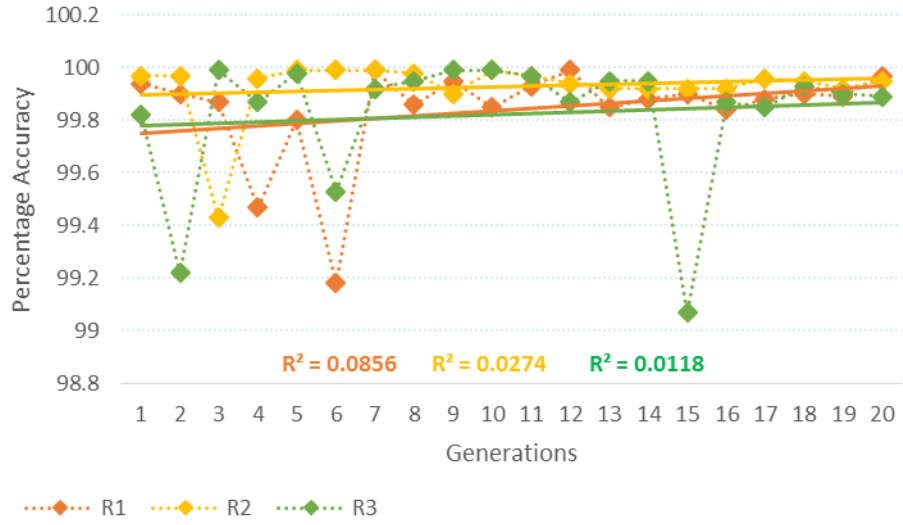
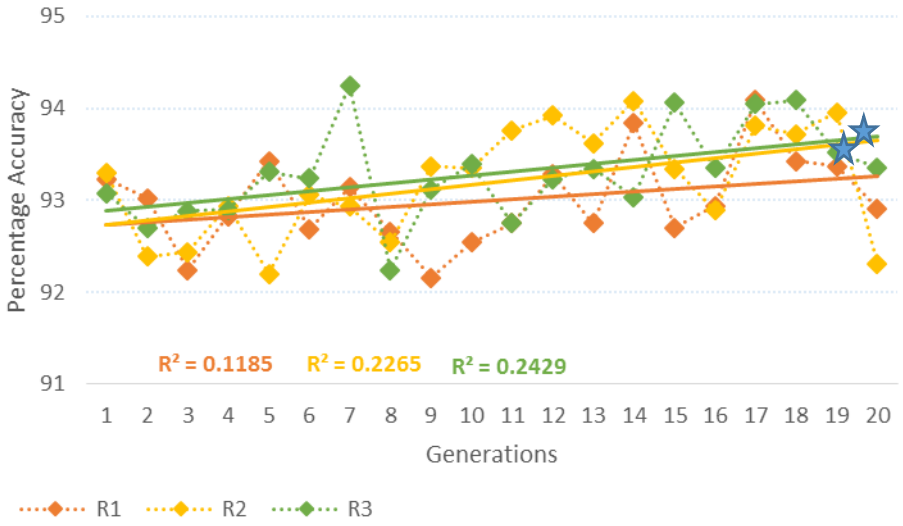


Fig. 5.1 (b): Mean performance per generation for breeding (left) and non-breeding (right) twin populations on Categorisation task



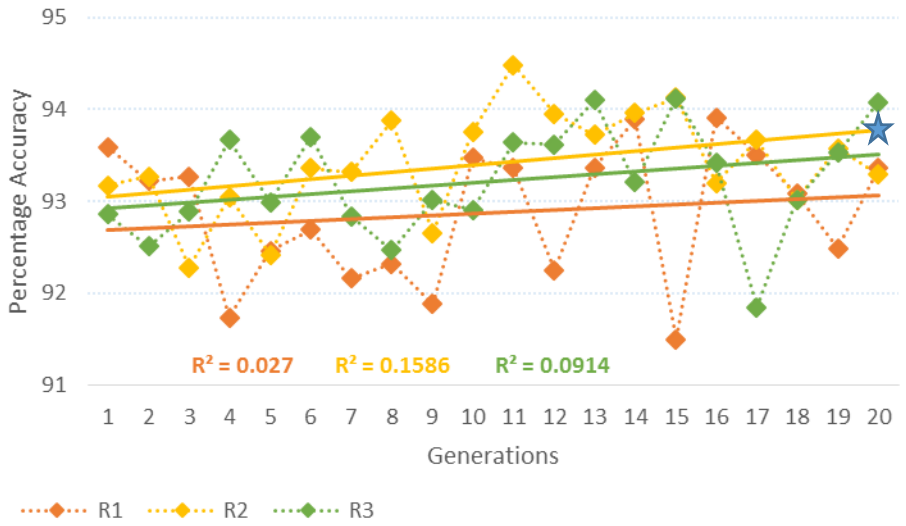
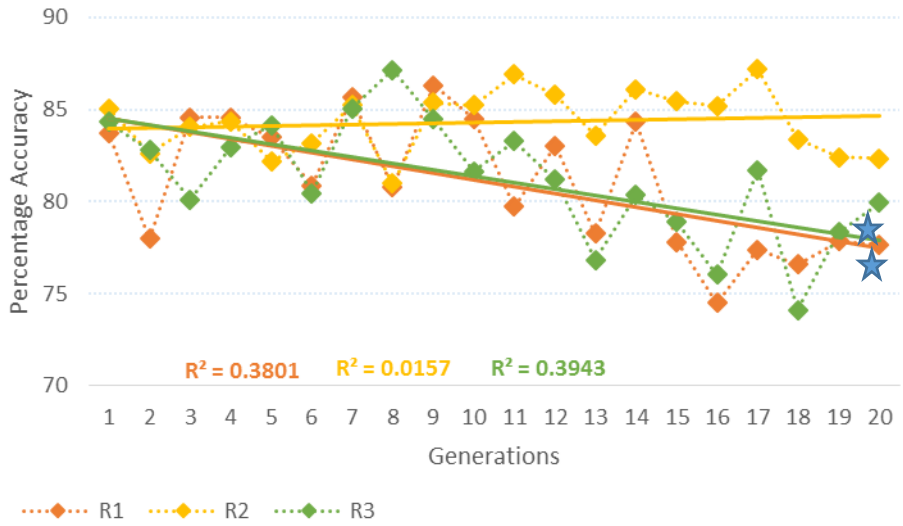


Fig. 5.1 (c): Mean performance per generation for breeding (left) and non-breeding (right) twin populations on Categorisation with exceptions task



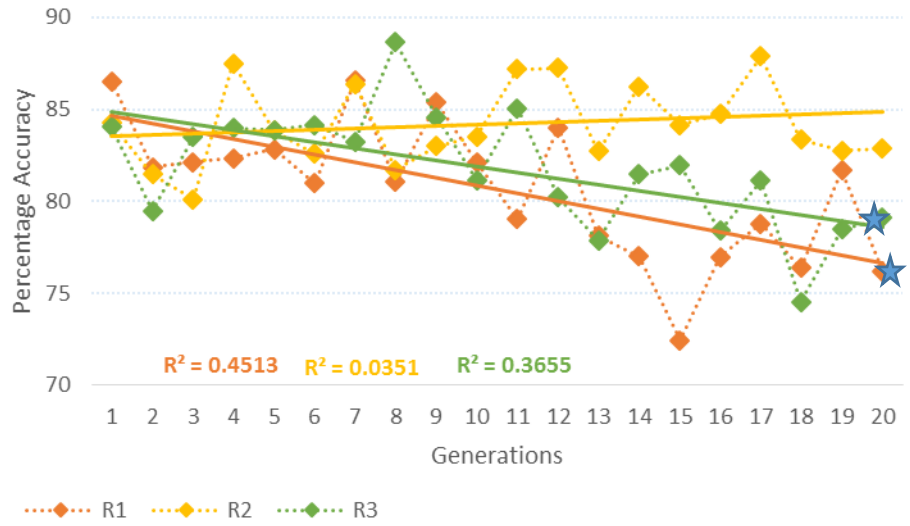


Fig. 5.1 (d): Mean performance per generation for breeding (left) and non-breeding (right) twin populations on Auto association task

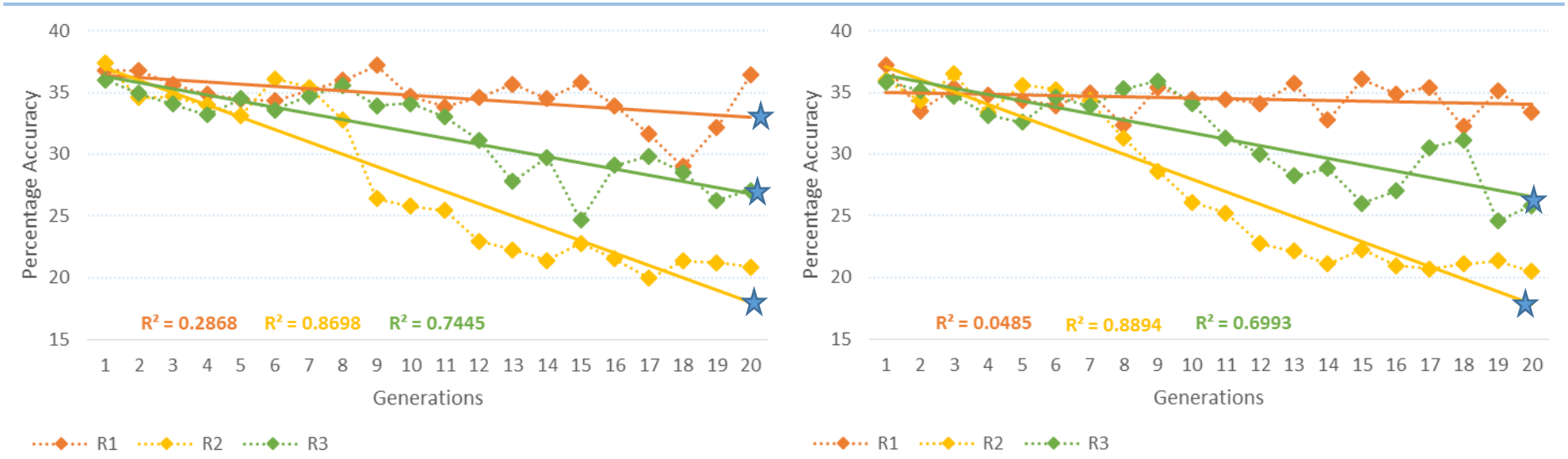


Fig. 5.1 (e): Mean performance per generation for breeding (left) and non-breeding (right) twin populations on Arbitrary association task

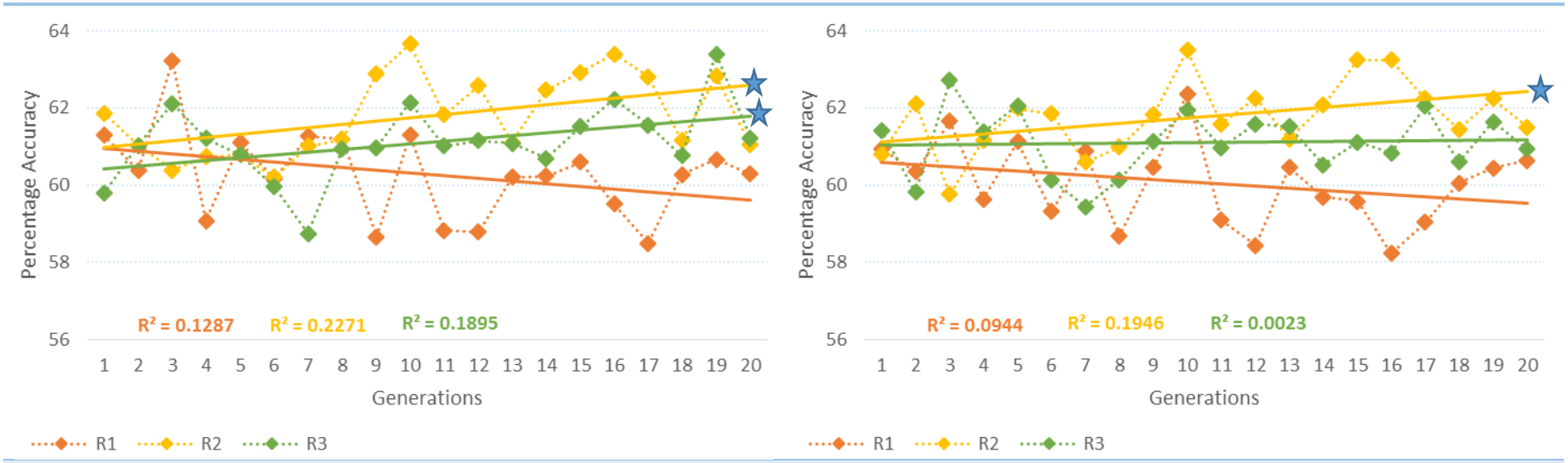


Fig. 5.2 (a): Mean generalisation accuracy per generation for breeding (left) and non-breeding (right) twin populations on English past tense acquisition task

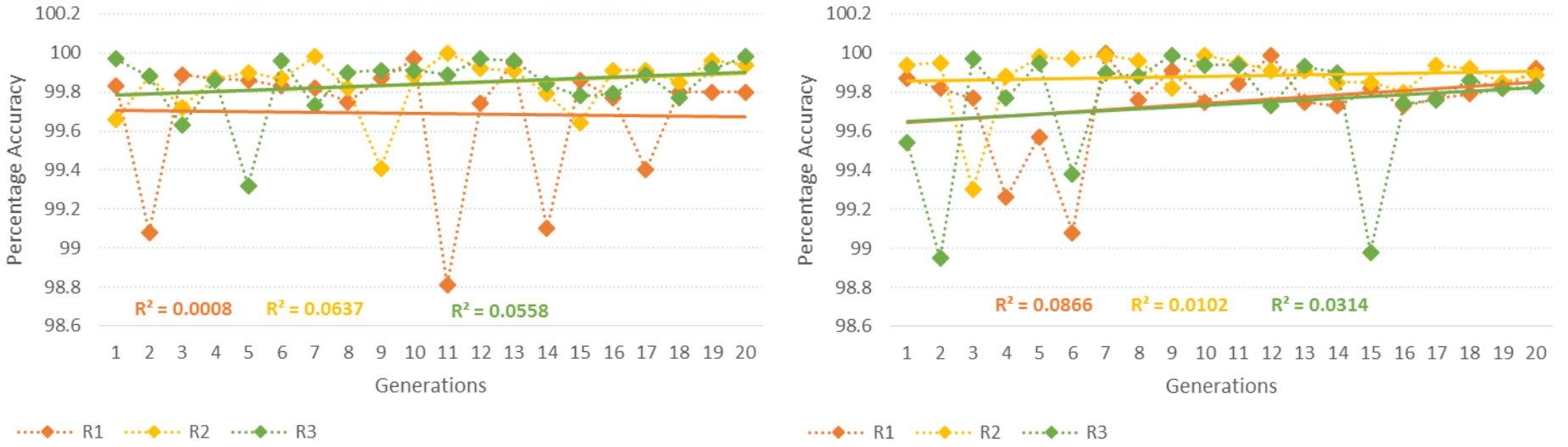


Fig. 5.2 (b): Mean generalisation accuracy per generation for breeding (left) and non-breeding (right) twin populations on Categorisation task

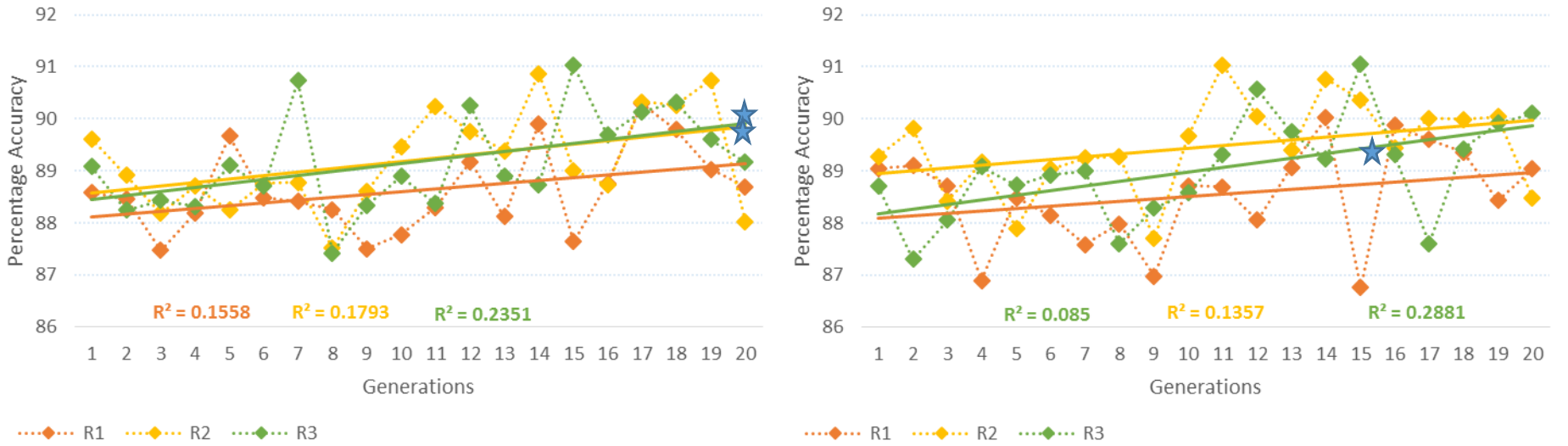


Fig. 5.2 (c): Mean generalisation accuracy per generation for breeding (left) and non-breeding (right) twin populations on Categorisation with exceptions task

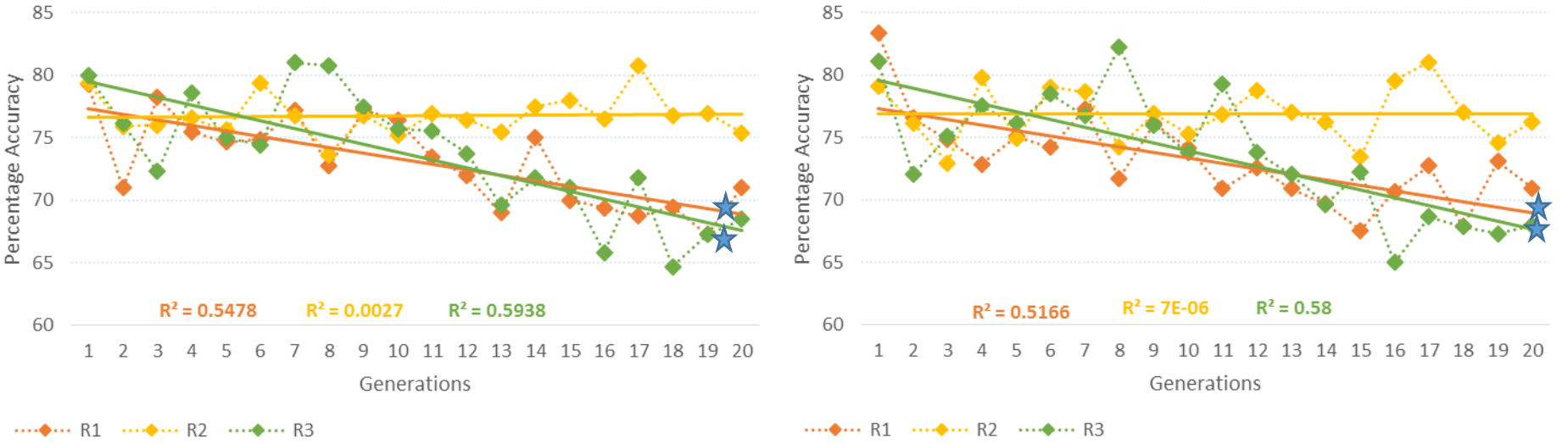


Fig. 5.2 (d): Mean generalisation accuracy per generation for breeding (left) and non-breeding (right) twin populations on Auto association task

Two important observations made from these accuracy graphs include, slow change in mean performance over generations and presence of some downward trends especially in the source task despite the operation of selection. This is somewhat counter-intuitive since, selection should serve to improve performance over generations. Genes conveying an advantage in learning are more likely to be transmitted to the next generation. The mode of sexual reproduction does not guarantee that the advantageous genes of an individual selected to breed will appear in the offspring, and the selection mechanism is itself probabilistically related to final performance level. Therefore, the probabilistic nature of this transmission accounts for the slow change in population mean performance over generations. It does not account for why performance could become *worse* over generations.

The downward trend especially in the source task, i.e. acquisition of English past tense in lineage one (R1), although not statistically significant, is explained by the quasi-regular domain of the task and stochastic selection operating on it. As explained in Chapter 3, English past tense has dual nature owing to its quasi-regular domain. Stochastic selection can target one of the aspects of the task, i.e. optimising performance on either regular verbs or irregular verbs or sometimes both. This can occur if there are parameters which favour learning on each (or both) aspects of the task. The combination of selection by mean performance, primarily driven by either aspect of the task, and the shift in neurocomputational parameter range towards values that favour learning of the targeted aspect of the task, together set the stage for replication being optimised only at the targeted aspect.

Since in replication 1, selection was targeting irregular verbs (refer to Chapter 3, Section 3.6.1) and past tense dataset is highly imbalanced with substantially fewer irregulars, the shift in intrinsic parameter range is not proving to be domain relevant. This also explains why the performance on auto association and arbitrary association experiences decrease through generations in these lineages. The tasks categorisation and categorisation with exceptions, on the other hand, have an improving trendline in all three lineages suggesting that the range of neurocomputational parameters is appropriate for learning these tasks. However, the accuracy levels achieved on these tasks are very high indicating possibility of ceiling effects. This means that the performance on these tasks is not really modulated by any specific neurocomputational parameter ranges and the networks can learn these tasks pretty easily. The estimates of heritability, shared and non-shared environmental variance give further insight. Figures 5.3 depicts the estimates of heritability for lineages 1, 2 and 3.

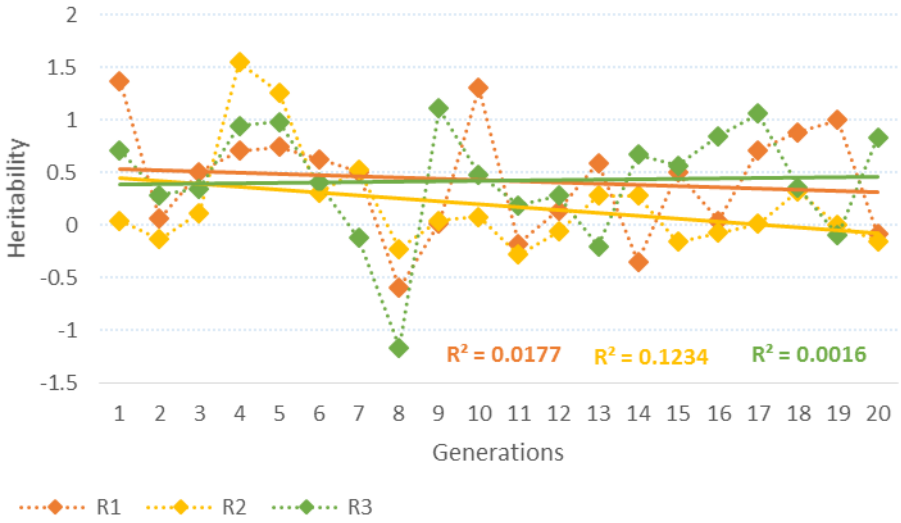


Figure 5.3(a): Heritability or proportion of variance due to genetic (or structural) factors for English PT

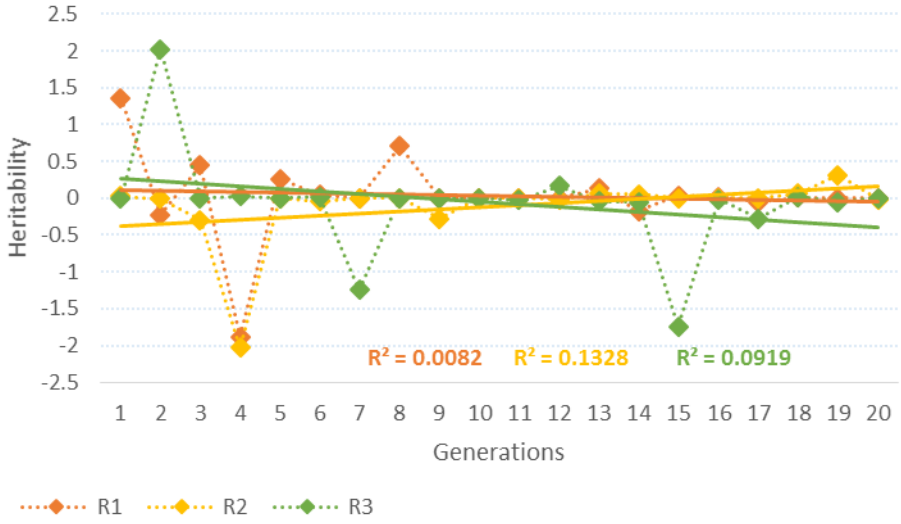


Figure 5.3(b): Heritability or proportion of variance due to genetic (or structural) factors for Categorisation

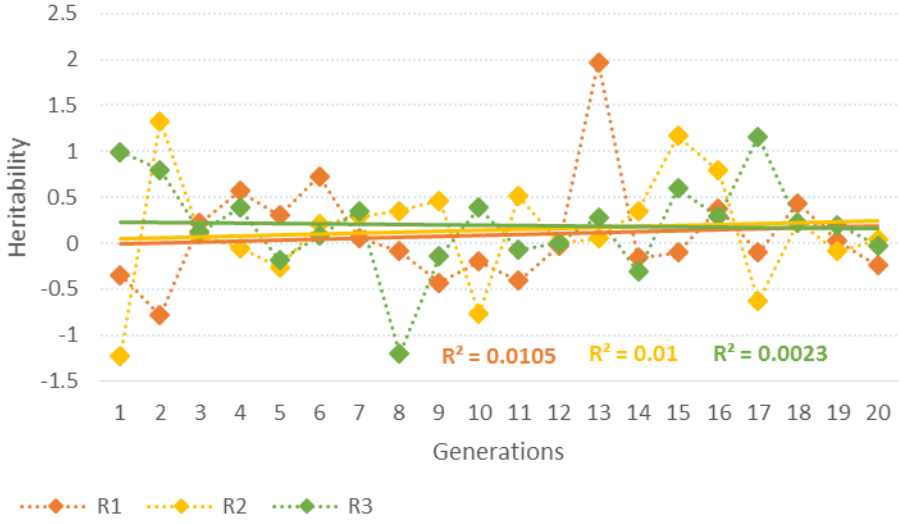


Figure 5.3(c): Heritability or proportion of variance due to genetic (or structural) factors for Categorisation Exp.

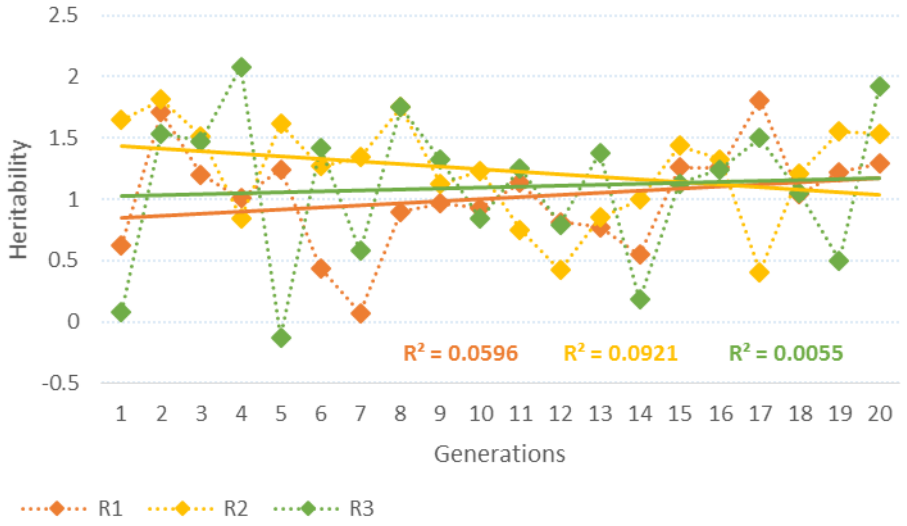


Figure 5.3(d): Heritability or proportion of variance due to genetic (or structural) factors for Auto association

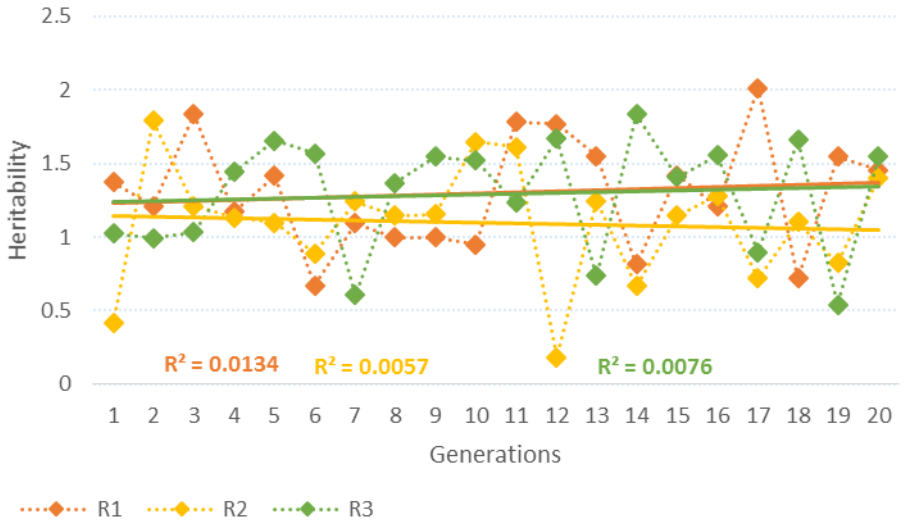


Figure 5.3(e): Heritability or proportion of variance due to genetic (or structural) factors for Arbitrary association

To compute heritability (refer Chapter 2, Subsection 2.5.5.2), Falconer's equations were used for these computations. These equations assume an additive genetic model and MZ correlation no more than four times the DZ correlation (Plomin et al., 2008). Though, according to the quantitative genetics fully meeting both of these conditions typically requires very large sample size to maintain standard error (Falconer and Mackay, 1995), we still use this metric since it helps in assessing the proportion of variance contributed by genetic and environmental factors and is also robust to parameter scale up. That is, if we were to increase the size of the genome by adding more parameters into it, this metric would still work.

When a population gets optimised on a task, the range of its domain relevant neurocomputational parameters should decrease across generations. For example, if populations are being optimised on task A, which requires more capacity to hold random mappings, then across generations, networks with lots of hidden units will have greater chance to get selected in the breeding pool. As a result, across generations the variability in the range of number of hidden units will become smaller, whereas the range of variation in other parameters, say initial learning rate remains unaffected. Assuming that learning task B depends on learning rate instead of number of hidden units, then in that lineage, heritability of task A will decrease and that of task B will increase or remain the same. This indicates that *optimisation and heritability have an inverse relationship*.

None of the heritability gradients in Figure 5.3 were significant. However, in-line with the above mentioned expectation, in lineage 1 the training and generalisation performance on English past tense task decreased, however the heritability for this task maintains an almost constant trend (centred on 0.5). The correlation between accuracy and heritability gradient was 0.24 which suggests a neutral relationship between the two in this lineage. Similarly, the mean accuracy trends for auto and arbitrary association tasks also experienced gradual decrease and their corresponding heritability showed an increasing trend. The accuracy-heritability correlation for these two tasks were either negative or close to zero, thereby further corroborating that optimisation and heritability are in-fact inversely related. The heritability for these two tasks is always maintained at very high values (over 1.0) throughout the lineage. This indicates that variation in performance on these tasks is largely due to neurocomputational differences. The accuracy-heritability correlation for categorisation and categorisation with exceptions, were negative and the heritability gradients maintained almost constant at nearly nil values, thereby affirming the inverse heritability-optimisation relationship.

Replication 2 is marked by improving accuracy trends for all tasks except arbitrary association. The heritability for past tense and auto association steadily declines, albeit non-significantly and it still centres around high value of 1.0 for the latter, signifying that selection isn't necessarily targeting domain relevant parameters and ANNs rely on their intrinsic properties for learning. Despite the non-significant heritability gradients, the negative accuracy-heritability correlation values for the said tasks affirm the inverse relationship between the two entities. Categorisation and categorisation with exceptions, experience a slight increase in heritability, despite an improving accuracy trendline. However, we should note that heritability

values and the accuracy-heritability correlation values for these tasks were always negative or closer to zero, reaffirming the ceiling effects, i.e. genetic factors do not have an effect on accuracy levels. A rather bizarre case is that of arbitrary mappings, wherein the accuracy levels decreased but the corresponding heritability levels also exhibited slightly decreasing trend. This seems counter-intuitive but Fig. 5.3(e) shows that the values of heritability are always higher than 1.0, suggesting that targeted neurocomputational ranges are not suited for optimising performance on arbitrary mappings. The accuracy-heritability correlation came out close to zero implying no relationship between the two for this specific task.

In replication 3, performance on English past tense, categorisation and categorisation with exceptions showed marked improvement. The heritability for the latter two decreased as expected, however the accuracy-heritability correlation was close to zero for categorisation with exceptions and 0.5 for categorisation task. This implies that there wasn't any dependence between accuracy achieved and genetic propensity of ANNs. Heritability for English past tense exhibited a non-reliable increasing trend across generations and the accuracy-heritability correlation was close to zero. In Chapter 3, Section 3.6.1, we saw that in this replication selection was targeting both regular verbs and irregular verbs, therefore the overall accuracy levels improved albeit slightly but at the same time since the two types of verbs are sensitive to differential ANN parameter ranges, the overall heritability shows some increase. Maintaining consistency with the inverse relationship between optimisation and heritability, auto and arbitrary association tasks displayed a marked decrease in their performance trend and an inverse/negative accuracy-heritability correlation value in this lineage.

Another noteworthy observation made from the aforementioned heritability Figures is that the range of variation of heritability gradients in all three replications is quite similar for - English past tense, categorisation and categorisation with exceptions, ranging mostly between (-0.5, +0.5) whereas auto and arbitrary association have their respective heritability values on higher end of spectrum ranging between (0.5, 2.0). It can thus be inferred that although these tasks are heterogeneous with respect to the definitions given in Chapter 4, Section 4.6.1, yet they have some underlying similarities that place these tasks in two groups – first containing the former three and the second consisting the latter two. Heritability therefore acts as an indicator of task relatedness.

When the heritability for a given task reduces, it implies that variation in performance is less due to genetic factors and more due to environmental factors, especially since we maintained

the range of shared environmental influences constant at all times. Figure 5.4 depicts the amount of variance in performance owing to variations in shared environmental factors. From Figure 5.4, one thing is evident that for both – breeding and non-breeding populations, variations in filtered training sets modulate performance variations very moderately. The only major exception to this observation is lineage 2, English past tense and auto-association, Figure 5.4 (a) and (d), wherein the trends were significant. The gradients start from low values and gradually reach very high values for English past tense, thereby implying that variations in performance were largely due to variations in the quality of training sets of ANN members, since the neurocomputational ranges were extremely good for all networks. For all other tasks, shared environmental factors were not relatively accountable for variations in performance accuracy levels attained. Although the filtered training sets only affect performance variations quite moderately, these are however important in attaining good accuracy levels.

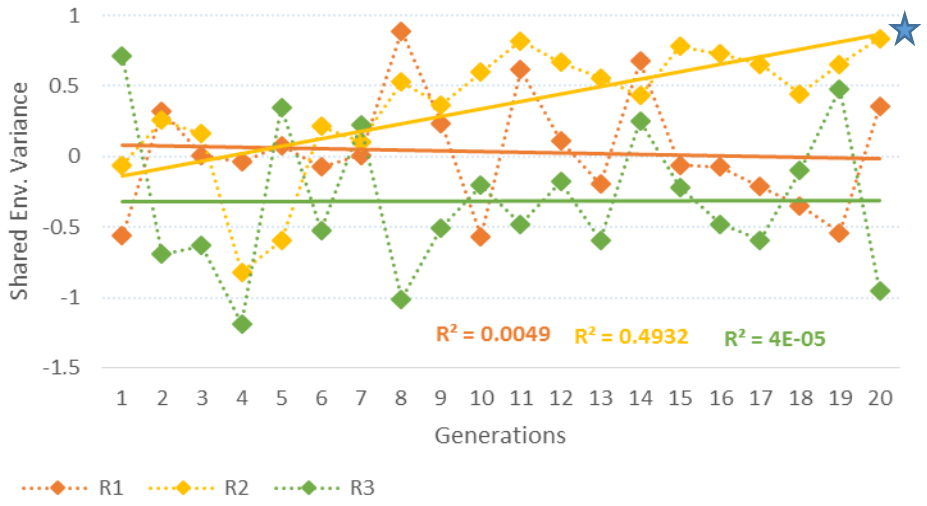


Figure 5.4(a): Proportion of variance due to shared environmental factors – English PT

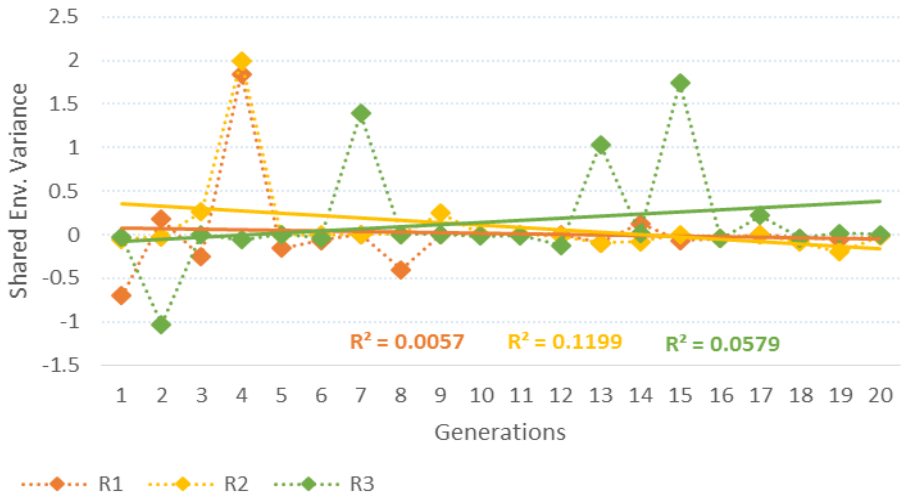


Figure 5.4(b): Proportion of variance due to shared environmental factors – Categorisation

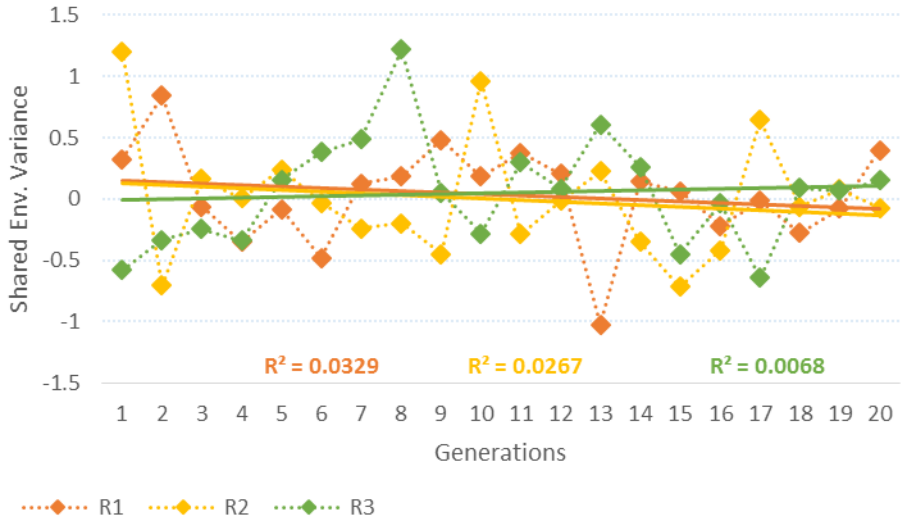


Figure 5.4(c): Proportion of variance due to shared environmental factors – Categorisation Exp.

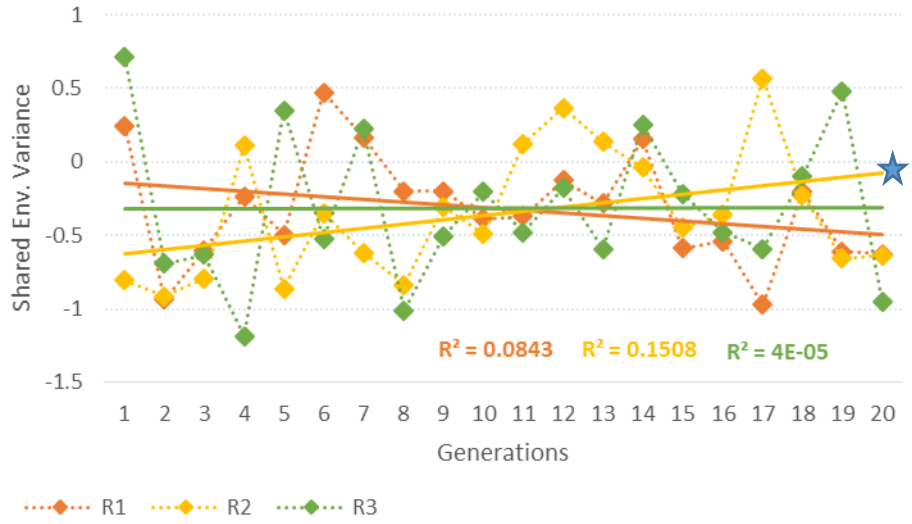


Figure 5.4(d): Proportion of variance due to shared environmental factors – Auto association

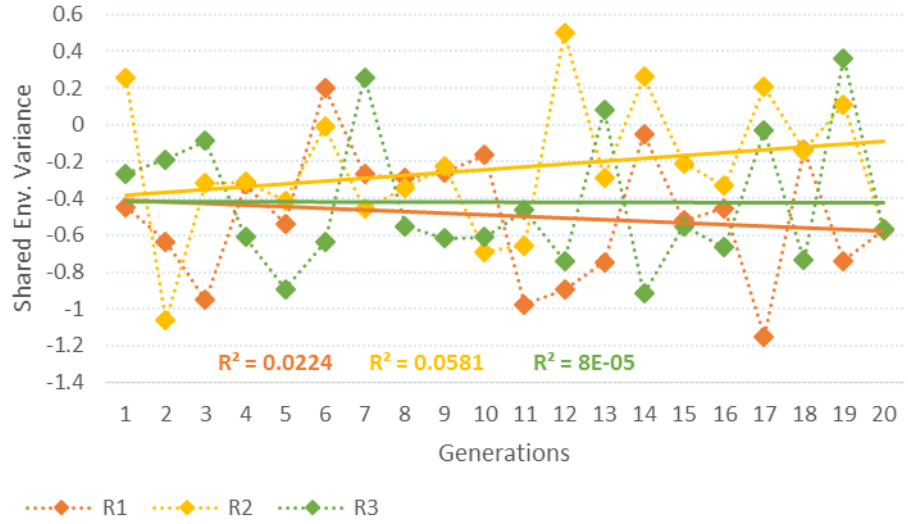


Figure 5.4(e): Proportion of variance due to shared environmental factors – Arbitrary association

The proportion of performance variation not accounted for by either genetic or shared environmental factors is due to non-shared factors (which includes error of measurement). In this work, initial weights of ANNs were used as representatives of non-shared environmental factors and Figure 5.5 (a) – (e) depict the proportion of variation caused due to variations in initial weight values of ANNs. The first observation drawn from these figures is that non-shared factors, i.e. differences in initial weights substantially modulated variations in behavioural outcomes, especially in case of English past tense, wherein all three gradients were found to be statistically significant.

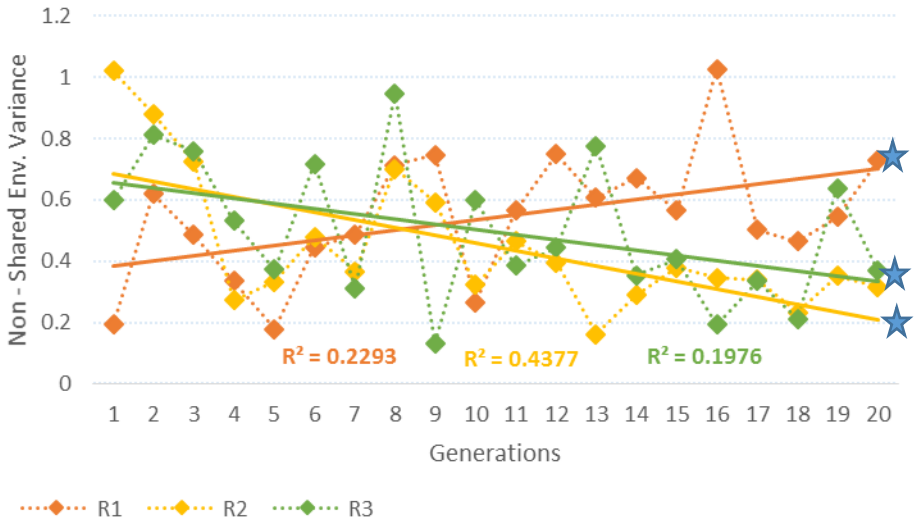
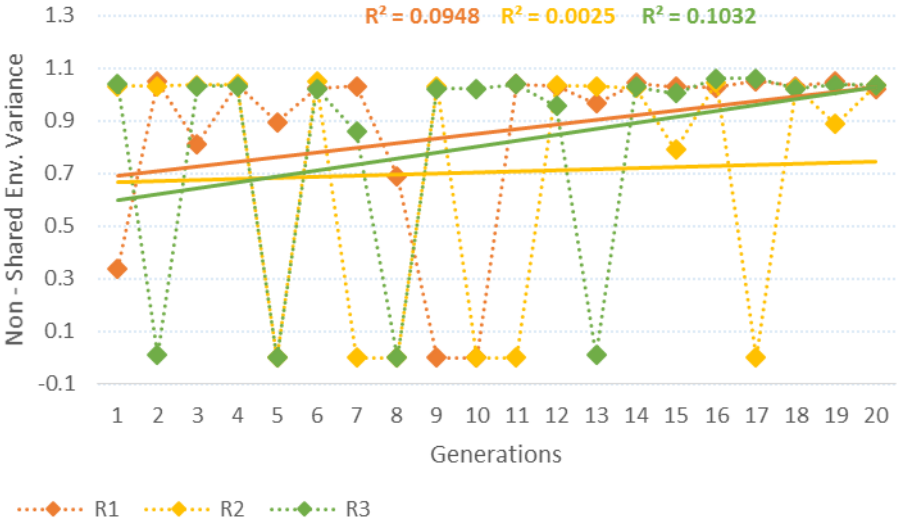
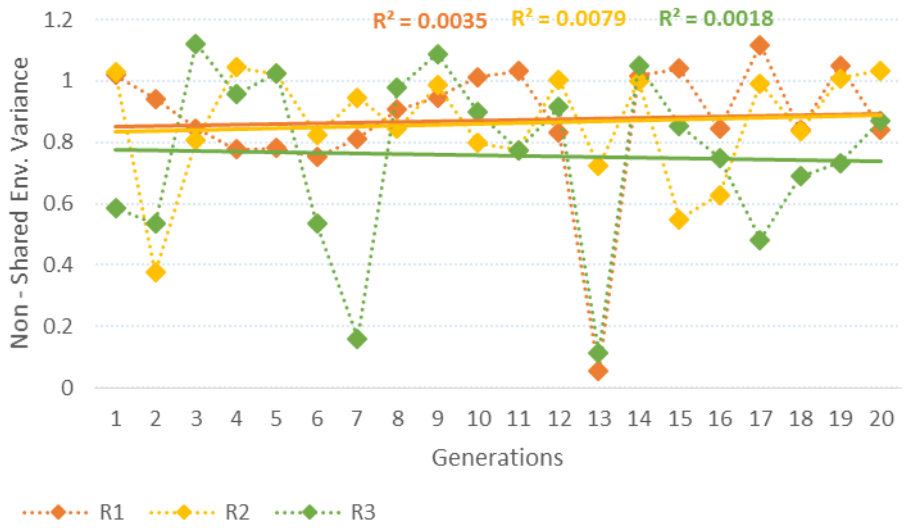


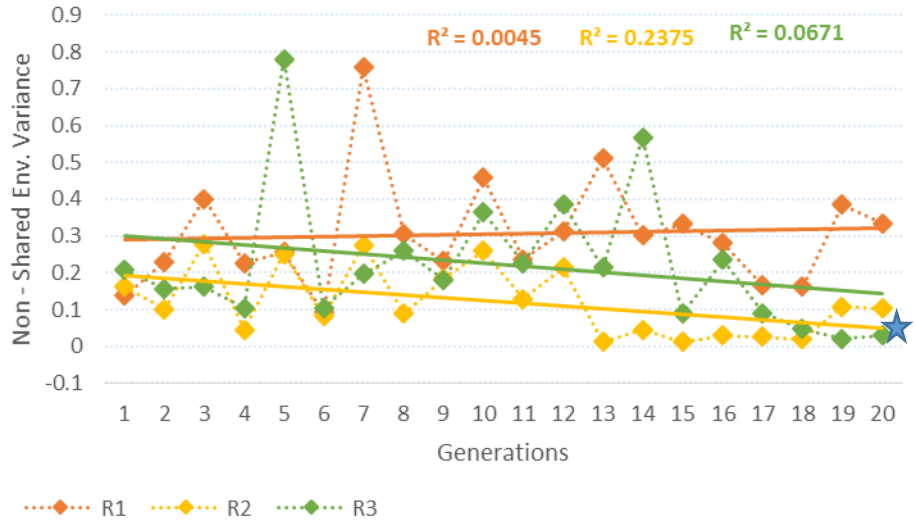
Figure 5.5(a): Proportion of variance due to non-shared environmental factors – English PT



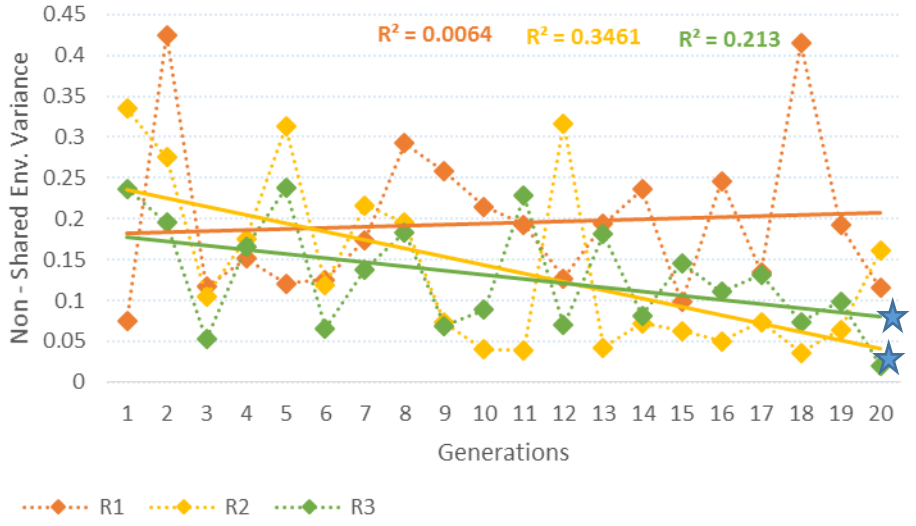
5.5(b): Proportion of variance due to non-shared environmental factors – Categorisation



5.5(c): Proportion of variance due to non-shared environmental factors – Categorisation Exp.



5.5(d): Proportion of variance due to non-shared environmental factors – Auto association



5.5(e): Proportion of variance due to non-shared environmental factors – Arbitrary association

Replication 1 is marked by an increasing trend for non-shared environmental variance for all tasks, albeit at different levels. For instance, in case of English past tense the trend starts off at moderate values (0.4) but gradually progresses to higher values (0.8). For categorisation and categorisation with exceptions, these values are consistently very high (over 0.8) but not reliable, however for the remaining two tasks, auto and arbitrary association, although the trend is increasing yet the contribution is as such moderate to low (around 0.2).

In lineages 2 and 3, the source task, English past tense sees a significant gradual decrease (from relatively high values to lower end values) in variance due to non-shared environmental factors. Categorisation, categorisation with exceptions maintain these non-reliable trends at the same level (at higher end of scale) and auto-association maintains the same nonsignificant constant trend (again at higher end of scale), throughout lineage 2 and in lineage 3. Arbitrary association tasks, experienced a significantly decreasing trend in replications 2 and 3, with values at the lower end of the spectrum.

These results suggest that non-shared environmental factors i.e. initial values of ANN weights play a significant role in modulating variations in behavioural outcome. Therefore it can be inferred that in situations where the intrinsic factors are not quite suited to the task at hand, having good initial weight values could give networks a learning bias i.e. training could be biased towards non-shared factors to enhance behavioural performance.

Finally, it was investigated which parameters and range of variation were being targeted by selection. To do that, the changes in the - mean values of these parameters and the entire range of variations throughout the lineages were measured. Figure 5.6 shows the changes in the mean values across generations and Figure 5.6d depicts the range of variation in each lineage. These parameters provide ANN populations with capacity (more number of hidden units equals more learning capacity) and ability (optimum values of learning rate and neither too steep nor too shallow values of slope of logistic) to acquire new tasks. Since each generation was instantiated using the same range, changes in the mean values of these parameters (owing to effects of selection and sexual reproduction) shows which parameters and what ranges are being targeted by selection whilst optimising on the source task.

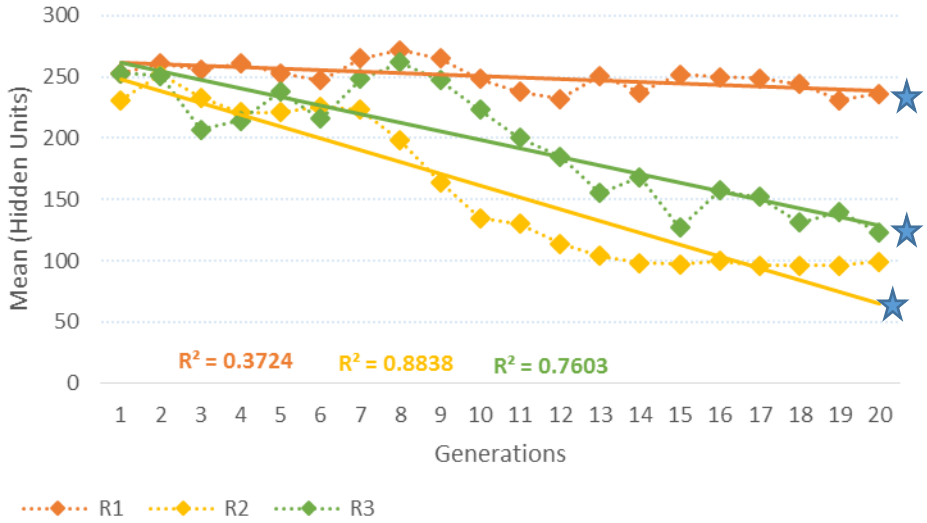


Figure 5.6(a): Change in the mean value of the number of hidden units per generation

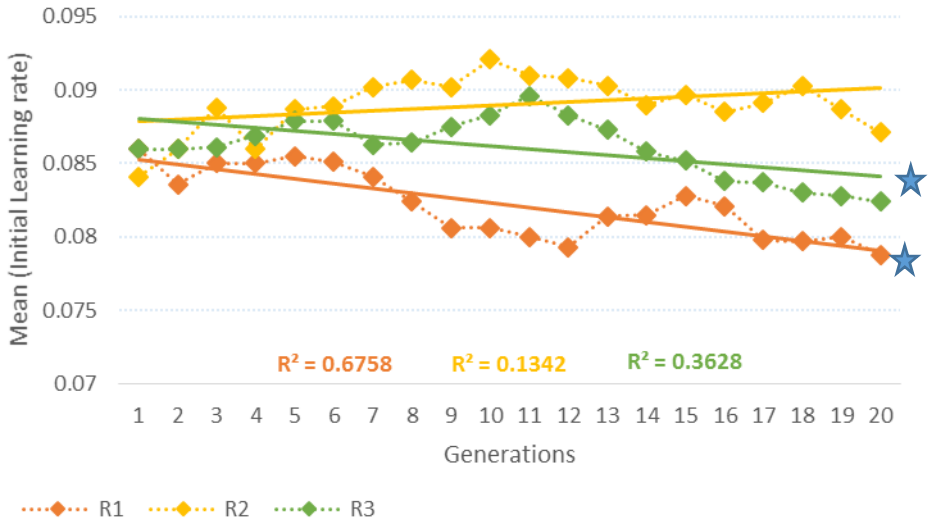


Figure 5.6(b): Change in the mean value of the initial learning rate per generation

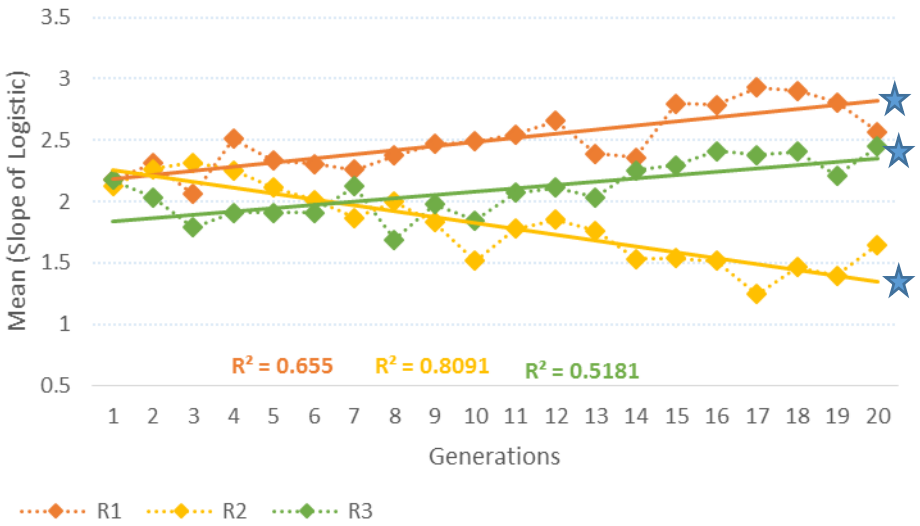


Figure 5.6(c): Change in the mean value of the slope of logistic activation per generation

Lineage 1 was marked by a decrease in performance accuracy of three tasks, and this was reflected by maintenance of high levels of hidden units along with decreasing learning rates and steeper logistic slopes. Also in that replication, the range of variation of parameters does not show any significant change/reduction for any parameter, implying that selection is not targeting/favouring any particular parameter or range. In lineage 2, performance steadily improved on almost all tasks with an exception of arbitrary association. This is reflected by an increasing learning rate and decreasing hidden units and slope of logistic. The range of variation of hidden unit narrows down significantly in this replication indicating that networks with lesser number of hidden units are being chosen by selection. Although networks are losing in terms of capacity but the mean of slope of logistic activation is also decreasing thereby giving networks ability to learn. However this loss of capacity led to reduced accuracy in arbitrary association tasks, implication being that reduced capacity hinders learning of arbitrary mappings. Finally in lineage 3, both the hidden units and learning rate decreased whereas the slope of logistic showed an increase, although the range of variations do not show much change across generations. This intermediate range of variation in parameters confirms the intermediate performance achieved in this lineage.

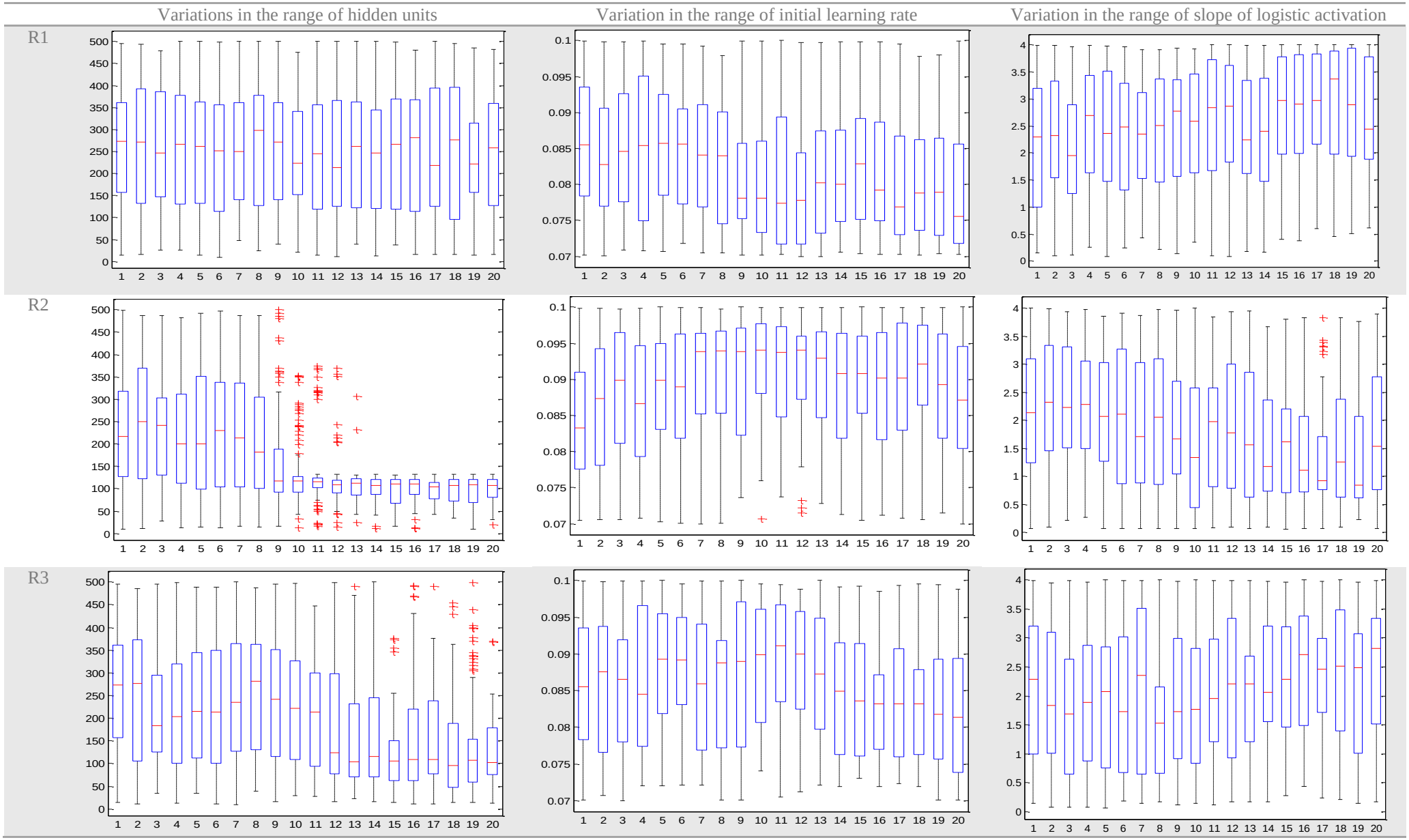


Figure 5.6d: Changes in range of variation of neurocomputational parameters across generations

5.5.1 Evaluating benefits of transfer

So far our analysis has led to some interesting findings. However, one question that requires more probing is – how successful has the transfer framework been within the scenario in which it was placed i.e. a quasi-regular source task and a stochastic selection operator used in conjunction with sexual reproduction. The success of the framework depends on the following key features:

- Ability to learn different heterogeneous tasks whilst retaining performance on the source task
- Avoiding negative transfer by assessing task relatedness
- Ability to find the domain-relevant range of variation for neurocomputational parameters

To analyse the performance of the proposed approach along these lines, two questions were formulated and the observations made from reported results were used to answer them. These are presented below:

Q1. Did the framework enabled the ANN twins' populations to learn multiple heterogeneous tasks?

The answer broadly speaking is yes. ANN populations were in fact able to successfully learn three tasks – English past tense (source task), categorisation and categorisation with exceptions. Nonetheless, it should be noted, that variance in performance of categorisation and categorisation w/exceptions was more due to variations in initial weights, although not significant statistically (refer to Figures 5.5 (b) and (c)) compared to heritability (refer to Figures 5.3 (b) and (c)), thereby suggesting that ANNs relied more on their initial weight values whilst learning these two tasks. However, accuracy on auto and arbitrary association tasks dropped across generations in all three replications. In the initial generations, the accuracy levels achieved on these tasks were good, but the populations were not able to maintain them. This is potentially due to the loss of capacity as indicated by the steady decrease in the number of hidden units, Figure 5.6 (a). An exception is lineage 2, wherein performance on auto association is maintained steady at 85% accuracy levels. In this level, although the networks lose in terms of capacity to learn, however they make up in terms of ability as reflected by

values for initial learning rate (ranging between 0.085 – 0.09) and slope of logistic activation (decreasing steadily) (refer to Figures 5.6 (b) and 5.6 (c)). Performance on arbitrary association, on the other hand suffers in all three replications because this task comprises random mappings and for learning random mappings, networks rely on lots of hidden units. Loss in the number of hidden units led to loss in performance accuracy levels on this task as well.

Overall, it can be inferred that though transferring the 'ability to learn' from source task to target tasks proved fairly helpful for categorisation and categorisation with exceptions, it did not prove very beneficial for auto and arbitrary association tasks. The main perpetrator however, is not the aspect of knowledge being transferred itself, instead it is the stochastic selection being applied on a quasi-regular task which is resulted in slow optimisation in some cases and performance degradation in others.

Q2. Was the proposed method able to avoid negative transfer by assessing task relatedness and having a domain-relevant range of variation for neurocomputational parameters?

The answer here is yes – the heritability metric used in this work provided us with a generic method of determining task relatedness between any given set of tasks. The range of variation of heritability values through the lineage acts as an indicator of task relatedness, i.e. tasks with heritability values varying between similar range tend to have some underlying similarity even though they might be heterogeneous per the definition in Chapter 4, Section 4.6.1. Figure 5.3 shows that trends for heritability values for English past tense, categorisation and categorisation with exceptions tended to vary between (-0.5, +0.5) range and have similar direction of observed heritability trends, whereas, in case of auto and arbitrary association heritability trends varied mostly between the range (+0.5,+1.5). This demonstrates that the former three tasks belong to one group and the latter two to another group based on their heritability values. This further strengthens the claim that range of neurocomputational parameters being optimised are more suited or domain relevant for former tasks, as affirmed by their respective performance accuracies as well. Therefore, the range of variation of heritability and the direction of observed trends emerging therein, could act as an indicator of underlying task relatedness.

Even in scenarios wherein heritability gradients turn out to be statistically insignificant, a key advantage of using heritability as a metric of task relatedness is that it summarises the net effect of all computational parameters varying within the learning system. As the heritability statistic measures variation in the performance values, the method is robust to increases in the number of parameters that vary in the learning systems, and which underlie any transfer effect. Furthermore, the range of variation and direction of heritability trends can be used to evaluate whether the transfer will be beneficial or not, though with a limitation - the method needs to transfer for at least a few generations to compute heritability and see what trends emerge therein. However in more real-time applications which possibly involve thousands of generations and computations are costly, this metric will still be helpful because only after a very few trials it can be deduced if there is negative transfer and if that's the case, further transfer can be stopped or some remedial measures can be taken.

5.6 Results and Analysis – truncation selection

In this instantiation of the transfer framework, the selection operator was changed to a deterministic (i.e. selecting only the fittest members) - truncation selection. The aim was to assess how the model fares under a new selection scenario. The results reported in this section follow three lineages each with a twenty generation duration that were increasingly optimised on the English past tense task using a truncation selection operator. The change in performance was traced across generations on this task, and the change in heritability; but also, crucially, the same measures are repeated when each succeeding past-tense-optimised generation was instead trained on the other four target tasks. Table 5.4 summarises the experiment design for lineages under this setting, i.e. replications 4, 5 and 6 (denoted by 'R').

No. of replications	3 (R_4 – R_6)
No of Generations per replication	20
Size of population	Breeding = 100; Non-breeding= 100 Total R_4+R_5+R_6 across generations= 12,000 ANNs per task
Size of Datasets	Training= 500 { 508(for past tense)} Generalisation= 500
Training Mode	Batch

Max. training epochs	100 (Past tense, categorisation & categorisation with exceptions) 500 (Auto & Arbitrary)
Early Stopping Criterion, maxstep (i.e. stop training if training accuracy does not improve till step == maxstep)	20 (English past tense, categorisation and categorisation with exceptions) 50 (Auto & Arb)
Initial weight update (Rprop learning rate)	Values from genome
Hidden units, Steepness of logistic	Values from genome
Selection Operator	Truncation- applied at the end of training
Crossover	6 crossovers/chromosome; single-point, multi-point & shuffle operators used
Environmental Factor (SES)	Probability value between 60% and 100%
Range of encoded neurocomputational parameters	No. of hidden units (10 – 500); initial learning rate (0.7 – 1.0); slope of logistic activation (0.0625 – 4.0)

Table 5.4: Experimental Design for truncation selection based replications

Figures 5.8 (a) – (e) and 5.9 (a) – (d) show the overall performance accuracy on the full training set and test/generalisation set for all five tasks. Each of these graphs summarise the results from 12,000 networks. A zigzagged line indicates the mean accuracy level of the 100 networks for each population at each generation, while a straight line represents the general trend observed in that replication scenario. The trend line was derived from a linear regression line based on the least squares method, predicting mean performance level from generation number. Regression analysis was used to determine individually reliable trend lines at .05 level, shown in graphs below with a blue star (★) either next to the gradient or near the corresponding legend. In few cases, R^2 values were relatively small, reflecting the non-monotonic changes in performance over generations.

The mean accuracy levels achieved on all tasks were very high and had an upward gradient, all of which were statistically significant with ($p<0.05$). The only exception was the categorisation with exceptions task which displayed a negative gradient in replications 4 and 5, however the accuracy gradients for this task were not reliable. The decrease in accuracy was very small and the populations still maintained very high accuracy levels varying between 98% - 99%. Another interesting observation drawn is the lack of performance variance in categorisation task. The ANN populations achieve almost 100% accuracy on this task in all three lineages including the generalisation accuracy. Additionally, in all three lineages, there is a big/fast change in population mean performance over generations for the auto association and arbitrary association tasks. Thus, although the ANNs were being optimised, or selected for their performance on English past tense acquisition task, yet the biggest performance improvement was in auto and arbitrary association tasks. Thus performance plots in Figures 5.7 and 5.8

depict that ANN twin populations were able to efficiently learn tasks different from what they were being selected for. Similar performance patterns emerged for generalisation ability also. Accuracy levels on categorisation task were nearly 100%, whereas English past tense and auto association had a reliably ($p<0.05$) increasing generalisation gradient, however in former case the gradient experienced slow increase whilst in latter the increase was fast. Categorisation with exception maintained almost stable generalisation trends over generations in all lineages, these trends were not found to be statistically significant though.

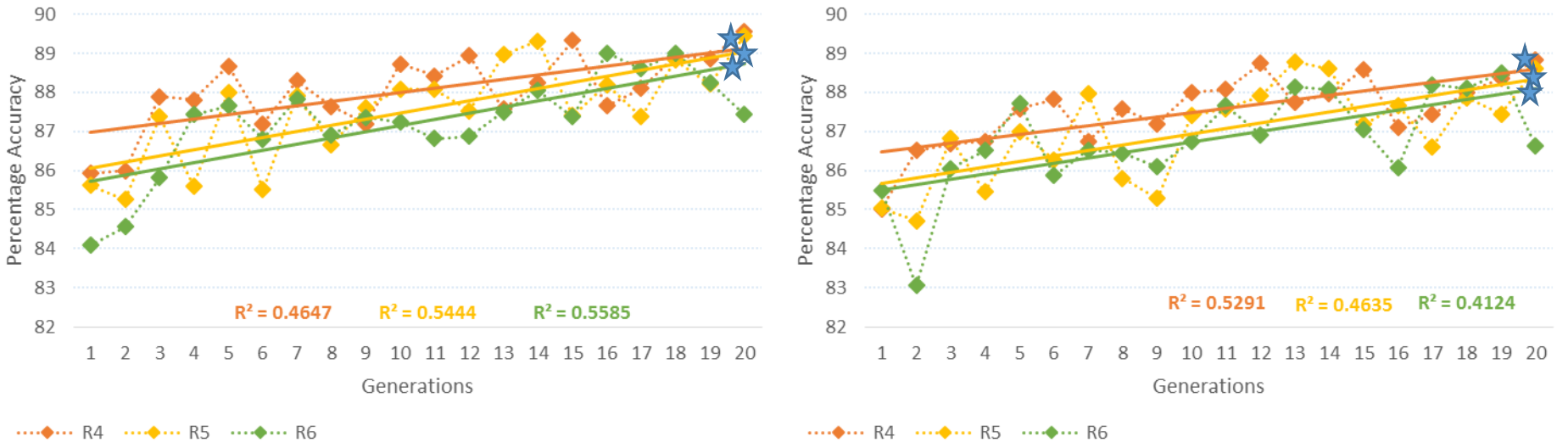


Figure 5.7(a): Mean performance per generation for breeding (left) and non-breeding (right) twin populations on English past tense acquisition task

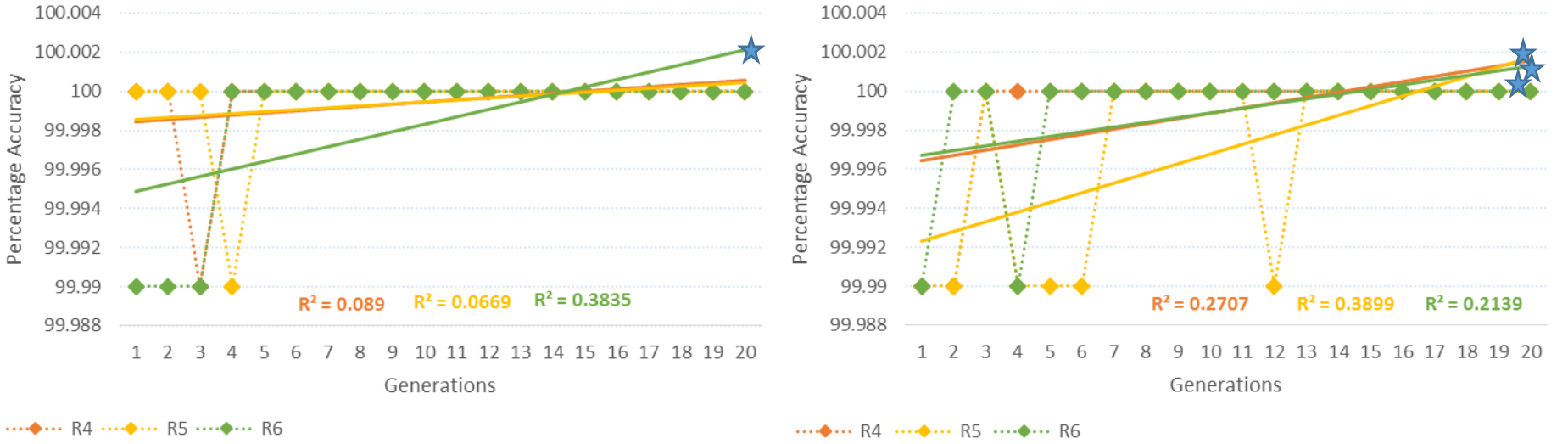


Figure 5.7(b): Mean performance per generation for breeding (left) and non-breeding (right) twin populations on Categorisation task

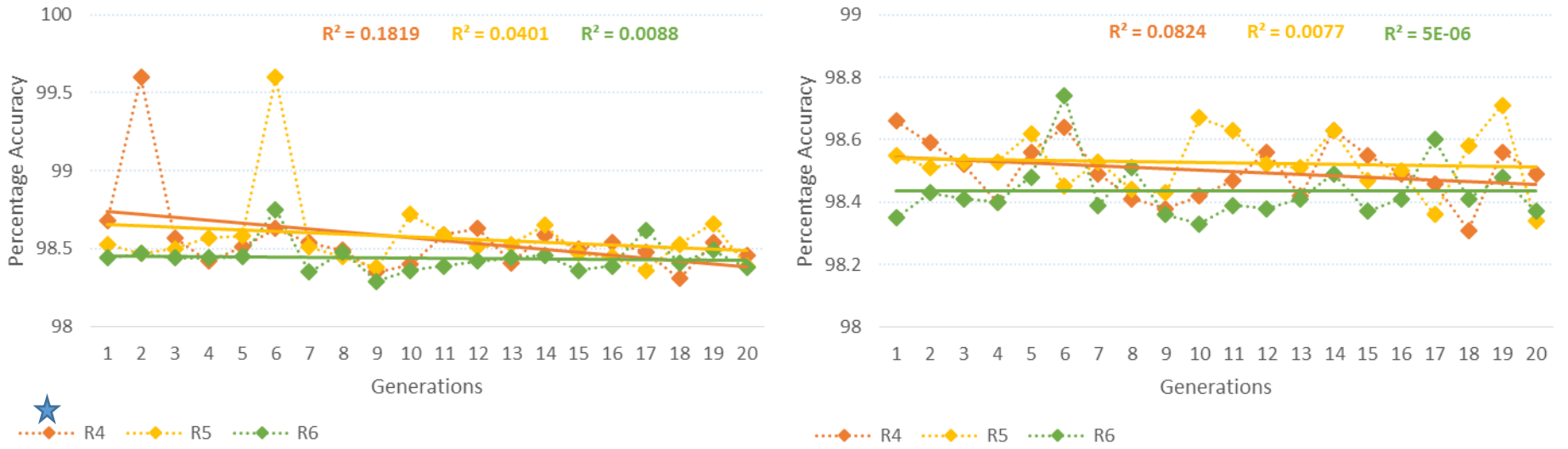


Figure 5.7(c): Mean performance per generation for breeding (left) and non-breeding (right) twin populations on Categorisation with exceptions task

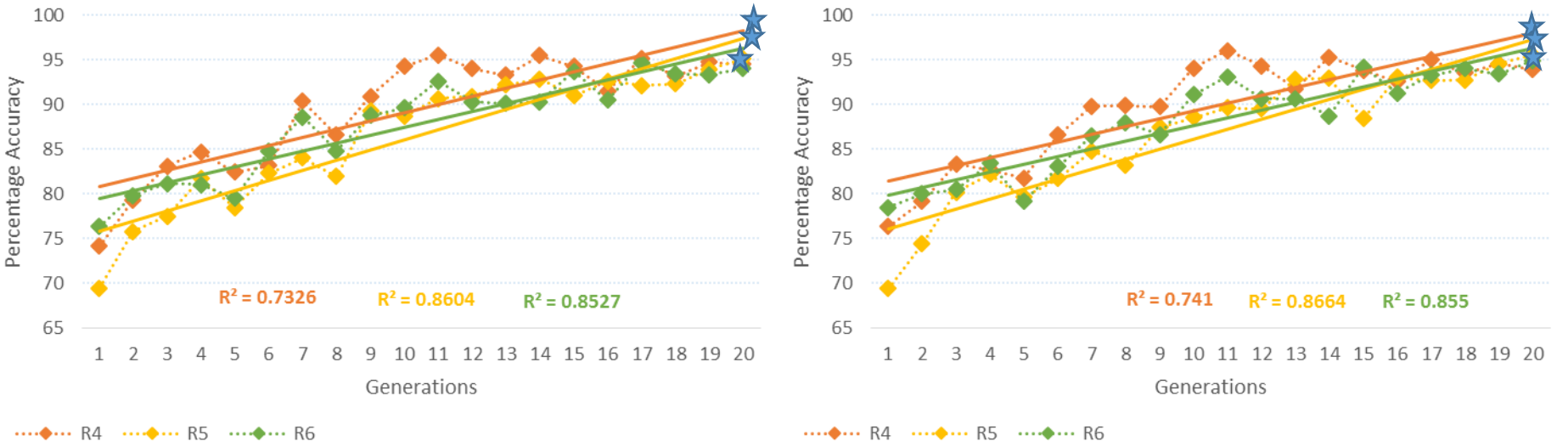


Figure 5.7(d): Mean performance per generation for breeding (left) and non-breeding (right) twin populations on Auto association task

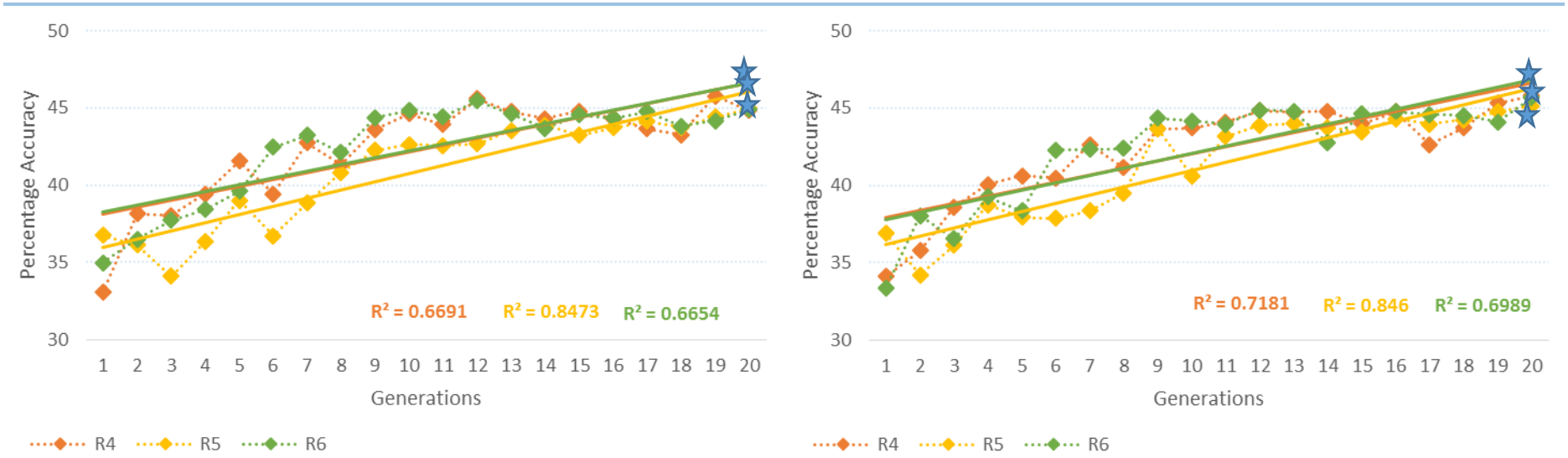


Figure 5.7(e): Mean performance per generation for breeding (left) and non-breeding (right) twin populations on Arbitrary association task

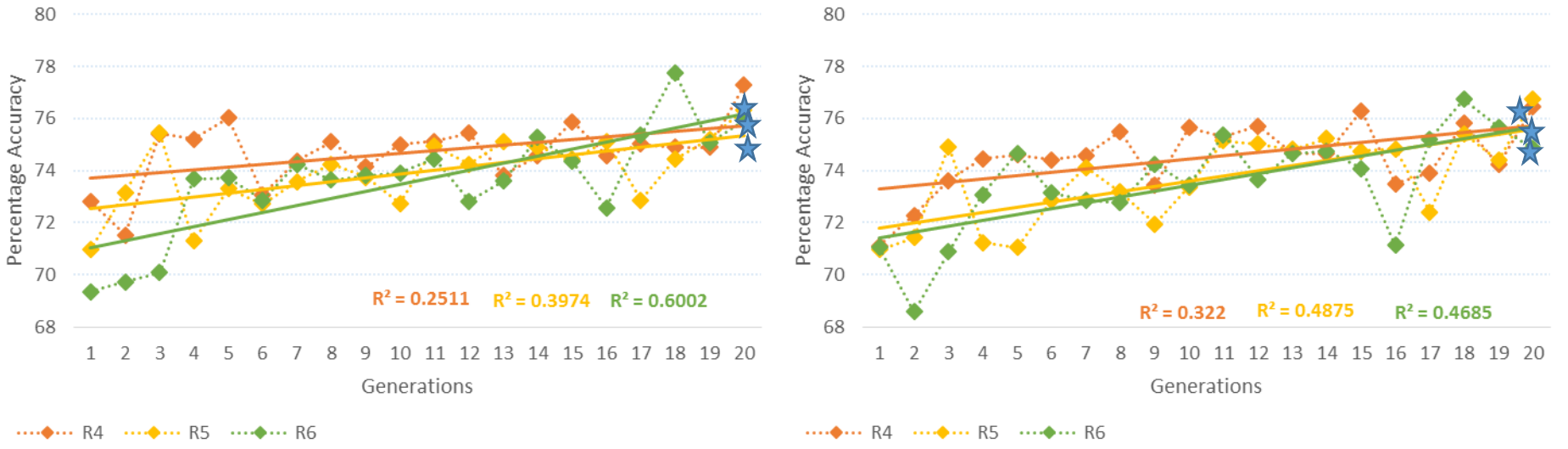


Figure 5.8(a): Mean generalisation accuracy per generation for breeding (left) and non-breeding (right) twin populations on English past tense acquisition task

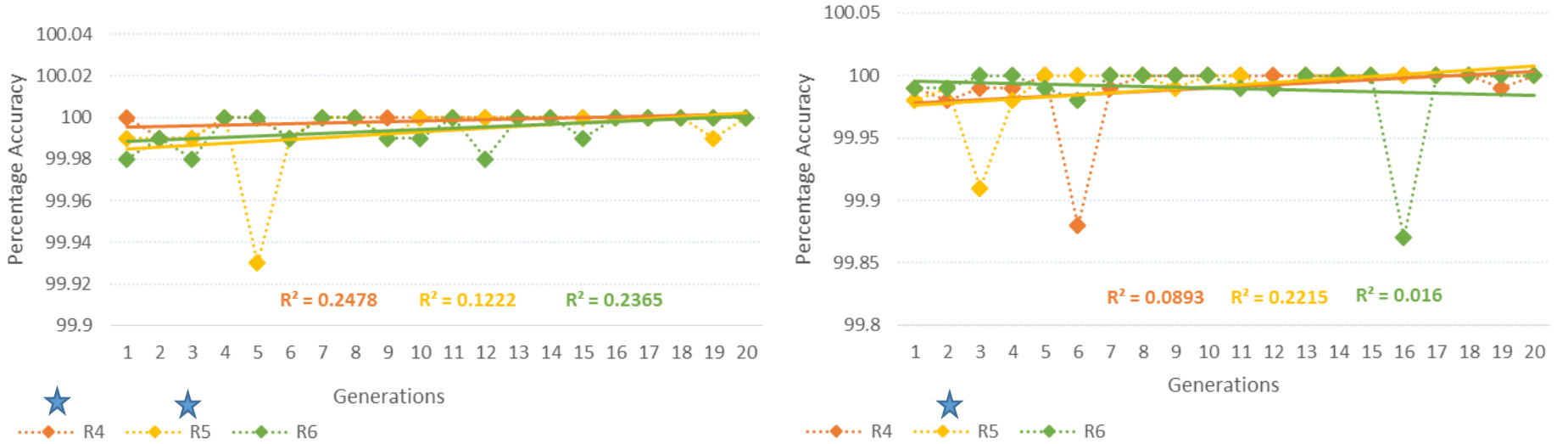


Figure 5.8(b): Mean generalisation accuracy per generation for breeding (left) and non-breeding (right) twin populations on Categorisation task

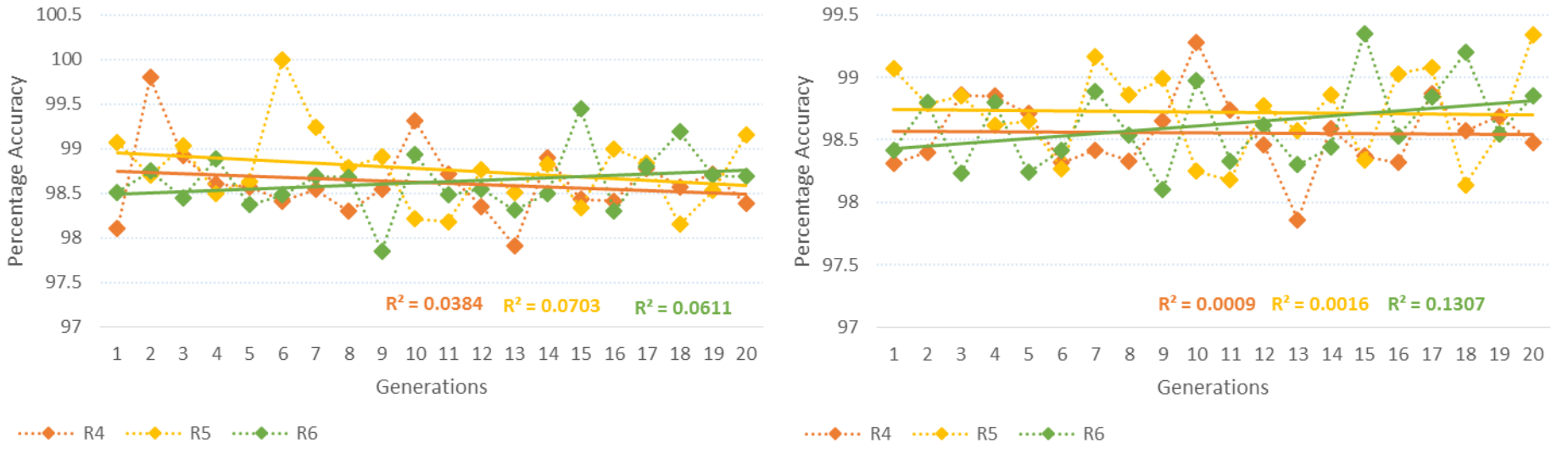


Figure 5.8(c): Mean generalisation accuracy per generation for breeding (left) and non-breeding (right) twin populations on Categorisation with exceptions task

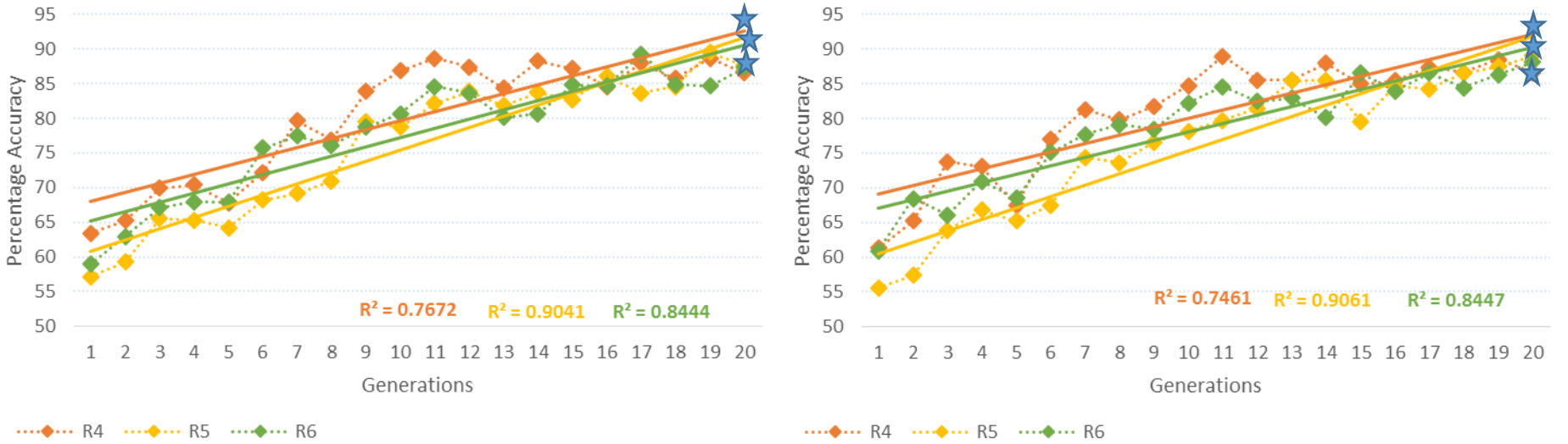


Figure 5.8(d): Mean generalisation accuracy per generation for breeding (left) and non-breeding (right) twin populations on Auto association task

When a population gets optimised on a particular task, the range of its domain relevant parameters should decrease, i.e. the variance in performance should become more due to differences in environmental factors. In these lineages, performance on all tasks shows improvement, which implies that heritability for all tasks should have a decreasing trendline. Figure 5.9 (a) – (e) depicts the heritability plots for all tasks. In case of categorisation task, population performance is at ceiling and has no variance. Therefore, heritability and effects of shared environmental influences are non-computable. Figure 5.9 (b) represents this and the values are not in-fact zero or nil.

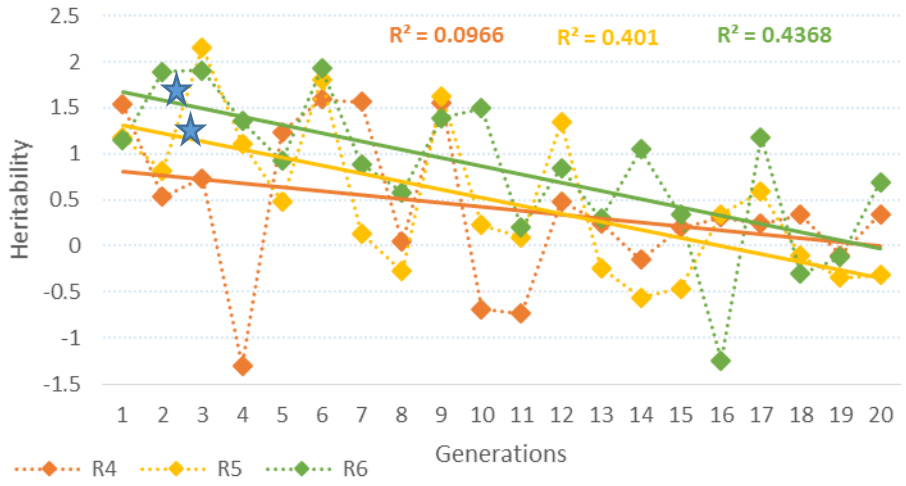


Figure 5.9(a): Heritability or proportion of variance due to genetic (or structural) factors for English PT

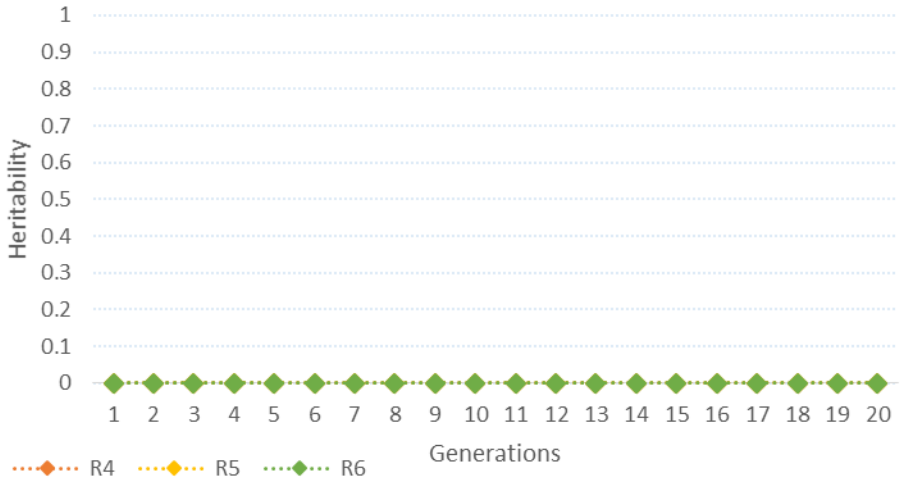


Figure 5.9 (b): Heritability or proportion of variance due to genetic (or structural) factors for Categorisation

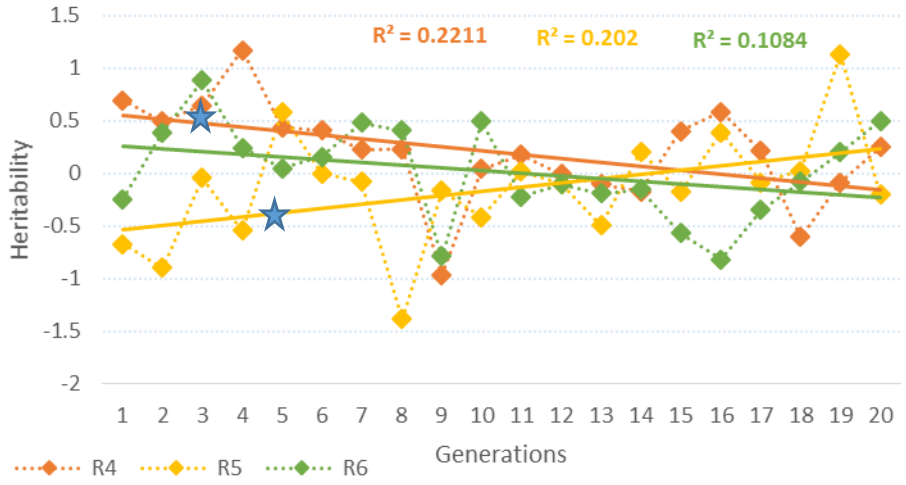


Figure 5.9 (c): Heritability or proportion of variance due to genetic (or structural) factors for Categorisation Exp.

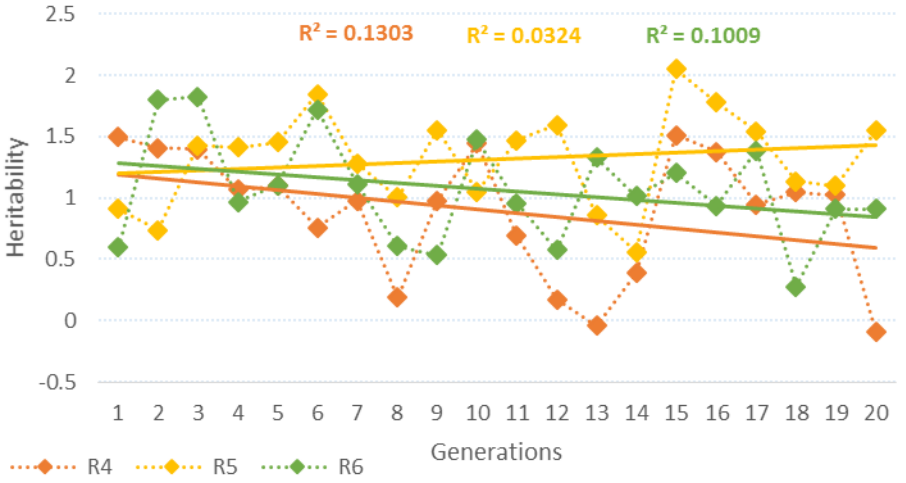


Figure 5.9 (d): Heritability or proportion of variance due to genetic (or structural) factors for Auto association

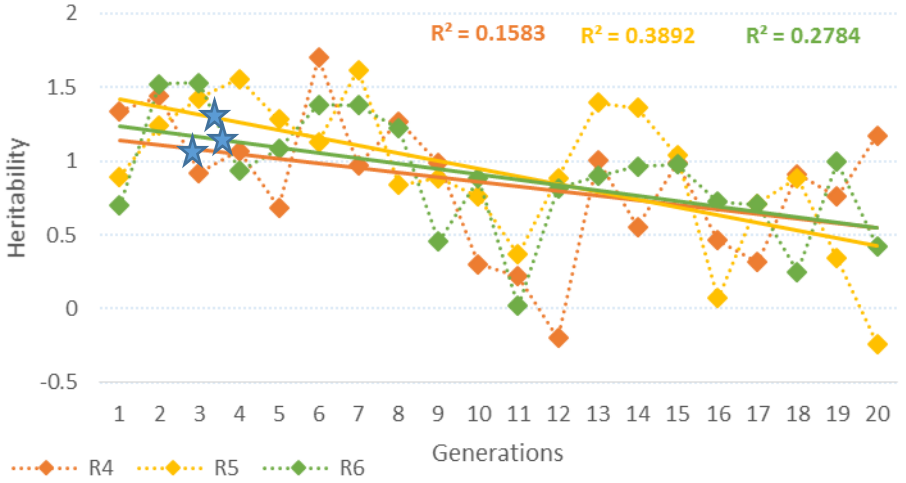


Figure 5.9 (e): Heritability or proportion of variance due to genetic (or structural) factors for Arbitrary association

The source task, English past tense acquisition, had reliably increasing performance accuracy gradients in all lineages and the heritability gradients decreased significantly in lineages 5 and 6. The heritability for English past tense task dropped steadily from very high values in initial generations to almost nil towards the end of each lineage. This indicates that over generations the variance in performance becomes less due to genetic differences. Similar inverse performance-heritability relationship holds for arbitrary association as well, however here the heritability drops from very high values to moderate values, implying that although the range of variation of neurocomputational parameters is domain relevant, still genetic differences substantially affect variations in accuracy levels attained. Additionally, the accuracy-heritability correlations were negative for English past tense and arbitrary-association tasks, thereby substantiating the inverse heritability-optimisation relationship.

As indicated in Figures 5.7 and 5.8, there is no performance variation for categorisation task in all lineages and ergo heritability is non-computable. Heritability for categorisation with exceptions and auto association tasks also has a decreasing gradient in replications 4 and 6, with the values for former varying between moderate to almost nil values, whereas for the latter varying between high to moderate values. Replication 5 however, experiences an increasing heritability gradient for both of these tasks and positive accuracy-heritability correlation, although the performance on auto association task improved substantially in this lineage and accuracy on categorisation with exception maintains nearly steady gradient. This scenario marked by an improving performance being accompanied with an increasing heritability is contradictory to the supposed inverse relationship between the two and this has been addressed later in this section.

The proportion of variance in performance accuracy due to differences in shared environmental factors, i.e. training sets is depicted in Figure 5.10 (a) – (e). The plots (a) and (e) in Figure 5.10 show a reliably increasing gradient for English past tense and arbitrary association tasks, especially in replications 5 and 6, thereby reaffirming that performance variation in these tasks is more due to environmental differences compared to genetic differences. Although the gradients are increasing, the range of variation however is quite small from -0.5 to +0.5 for past tense task and -0.6 to +0.2 for arbitrary association. Categorisation task was marked by nil values for shared environmental factors which is in line with its no performance variations, which renders shared-environmentability non-computable (refer Figure 5.10 (b)).

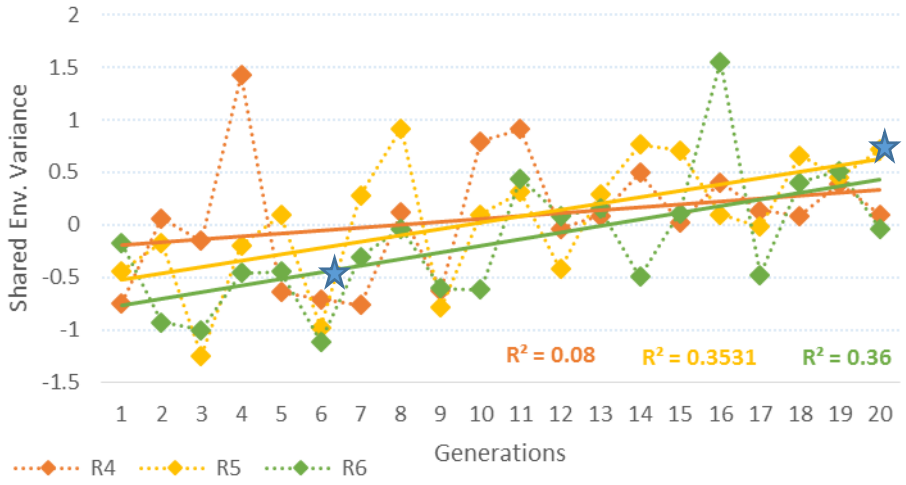


Figure 5.10(a): Proportion of variance due to shared environmental factors – English PT

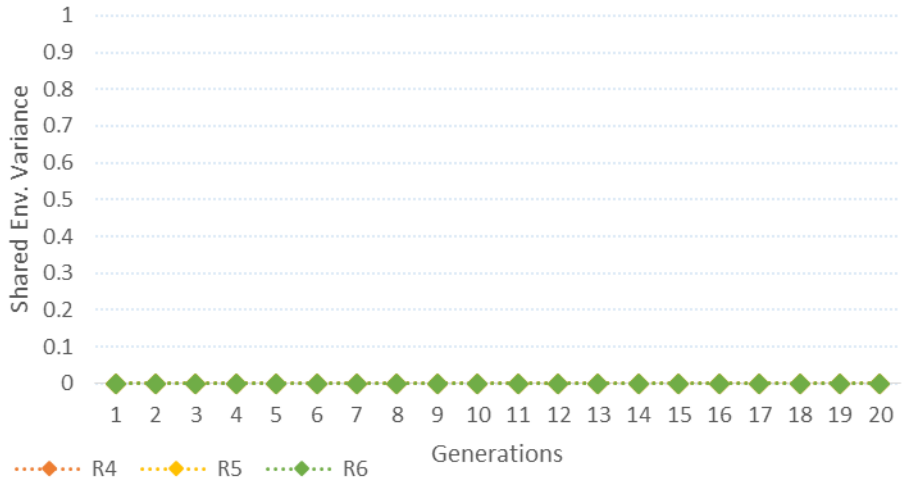


Figure 5.10(b): Proportion of variance due to shared environmental factors – Categorisation

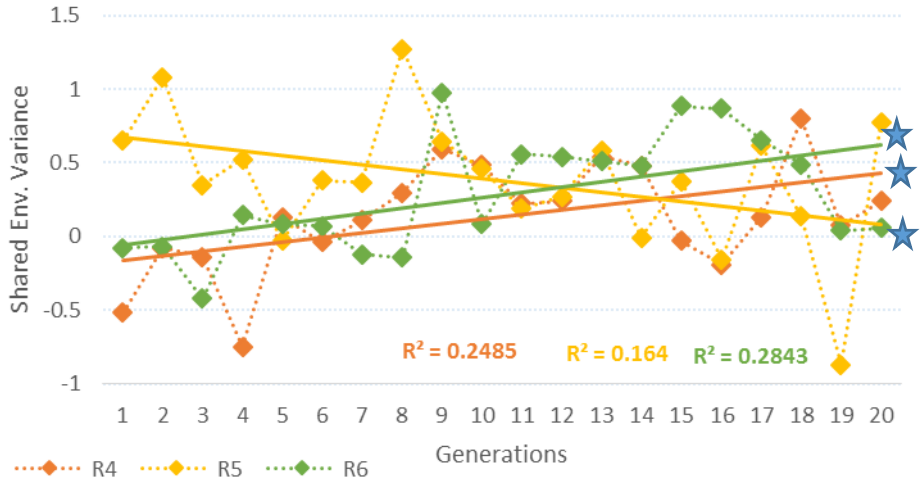


Figure 5.10(c): Proportion of variance due to shared environmental factors – Categorisation Exp.

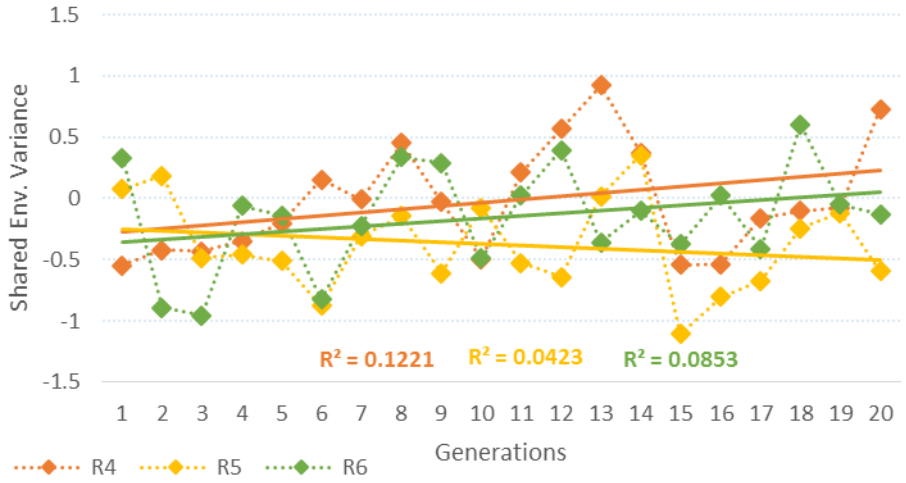


Figure 5.10(d): Proportion of variance due to shared environmental factors – Auto association

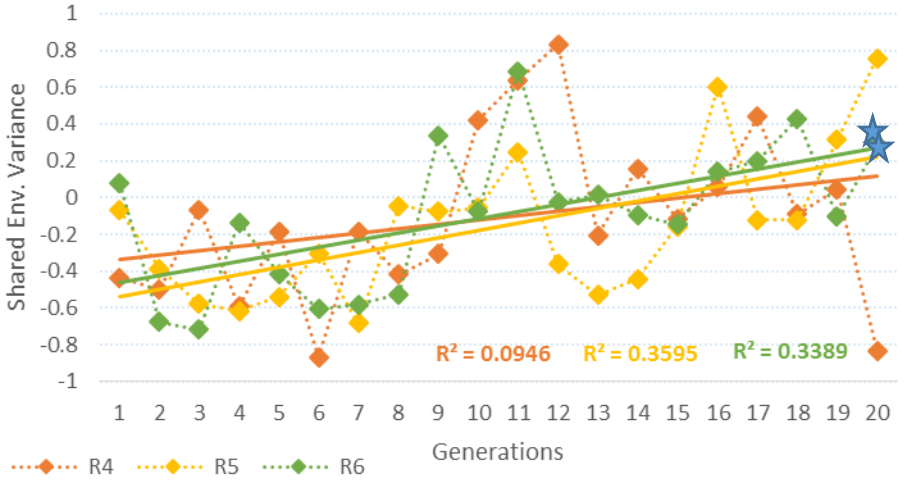


Figure 5.10(e): Proportion of variance due to shared environmental factors – Arbitrary association

Gradients for shared environmental variance changed significantly in replications 4, 5 and 6 for categorisation with exceptions tasks. For remaining tasks, the effects of shared environmental influences weren't found to be reliable.

The variance in performance not accounted for by genetic or shared environmental factors must be attributed to differences in non-shared environment (which also includes error of measurement) i.e. unique initial weights of ANNs. Figure 5.11 shows the results for proportion of variance due to differences in initial weight values of ANNs.

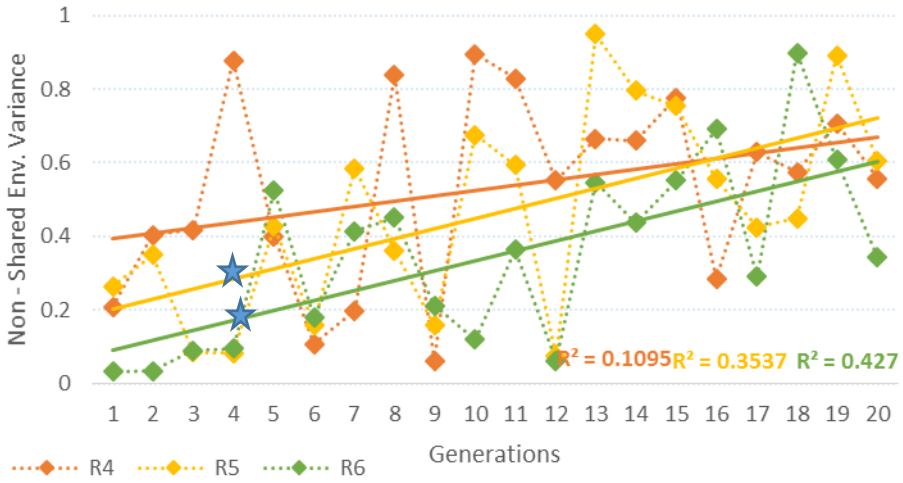


Figure 5.11(a): Proportion of variance due to non-shared environmental factors – English PT

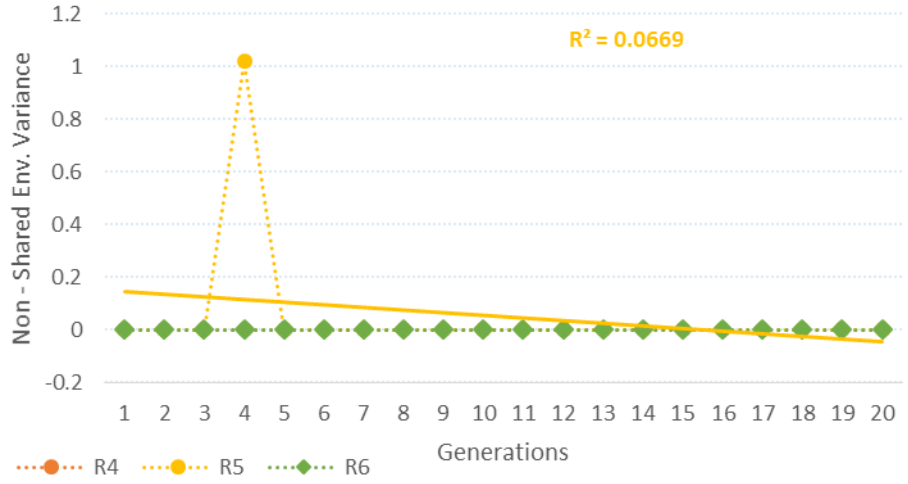


Figure 5.11(b): Proportion of variance due to non-shared environmental factors – Categorisation

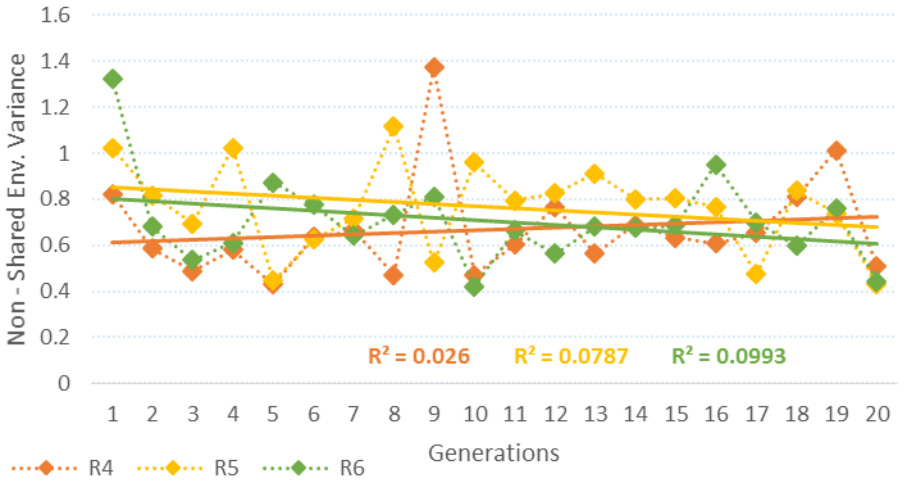


Figure 5.11(c): Proportion of variance due to non-shared environmental factors – Categorisation Exp.

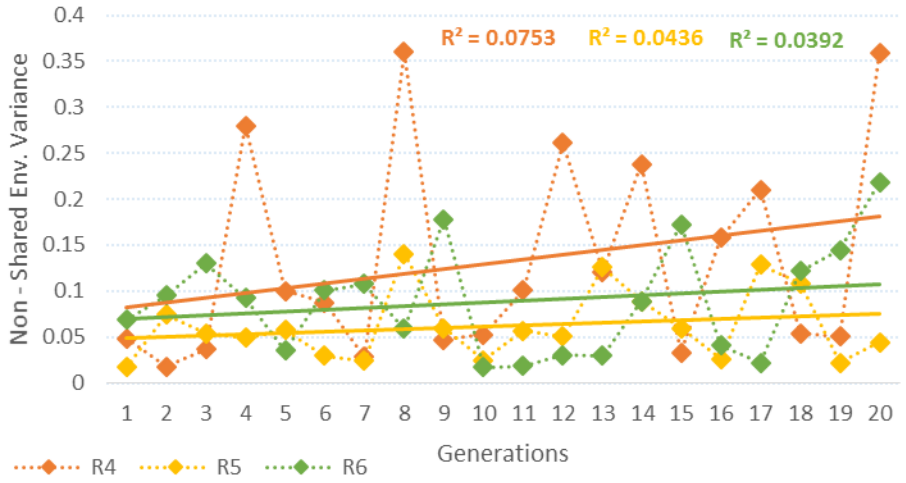


Figure 5.11(d): Proportion of variance due to non-shared environmental factors – Auto association

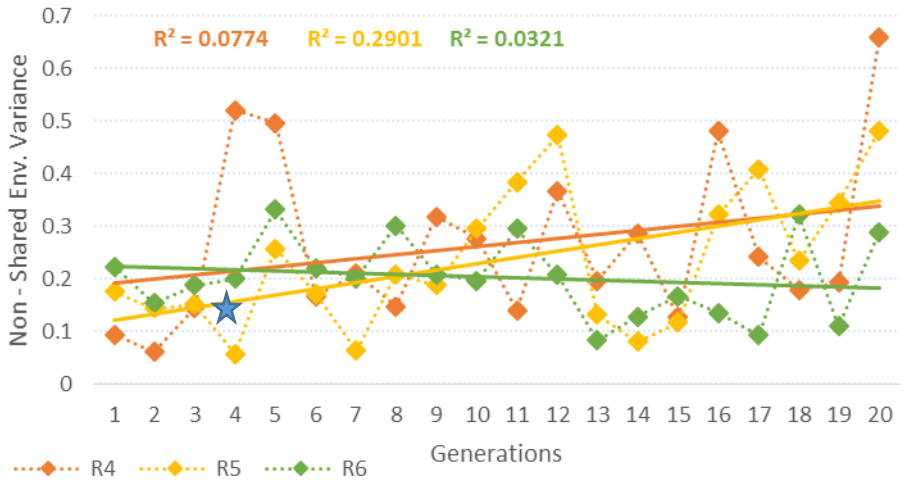


Figure 5.11(e): Proportion of variance due to non-shared environmental factors – Arbitrary association

It is known that the learning speed and fast convergence of many feed forward neural networks depend to some extent on their initial values of weights and biases (refer to Chapter 2, Section 2.5.3.2), which in turn implies that differences in initial weight values should lead to variations in final behavioural outcome as well. Figure 5.11 (a) reiterates this by depicting steadily reliably increasing gradients in lineages 5 and 6. The variation in performance due to weight differences have small to moderate values at the beginning of lineages but then increase to moderate to high values towards the end of lineages. Categorisation task once again showed no variance due to non-shared environmental factors mainly due to lack of any behavioural variance in the first place. Categorisation with exception task, had non-significant trends, nevertheless, the range of variation was always quite high thereby suggesting that initial weight values were an important factor accounting for behavioural variance. In case of the arbitrary

mapping task, replication 5 had slightly increasing gradient at the lower end of spectrum whilst other lineages showed no reliable modulating effect of initial weights. Thus non-shared environmental factors are important in determining behavioural variance especially in case of English past tense acquisition and categorisation with exceptions. However, for auto and arbitrary association tasks, initial weights only moderately affect behavioural variance, which is in fact mostly modulated by differences in genetic factors as can be seen from heritability plots in Figure 5.9.

Finally, since all tasks experienced performance improvement/good performance over generations in all replications, it can be inferred that truncation selection targets the neurocomputational parameters with values in a domain-relevant range. Figure 5.12 (a) – (c) and Figure 5.13 depict the changes in the mean values and the range of variation of these parameters over generations respectively.

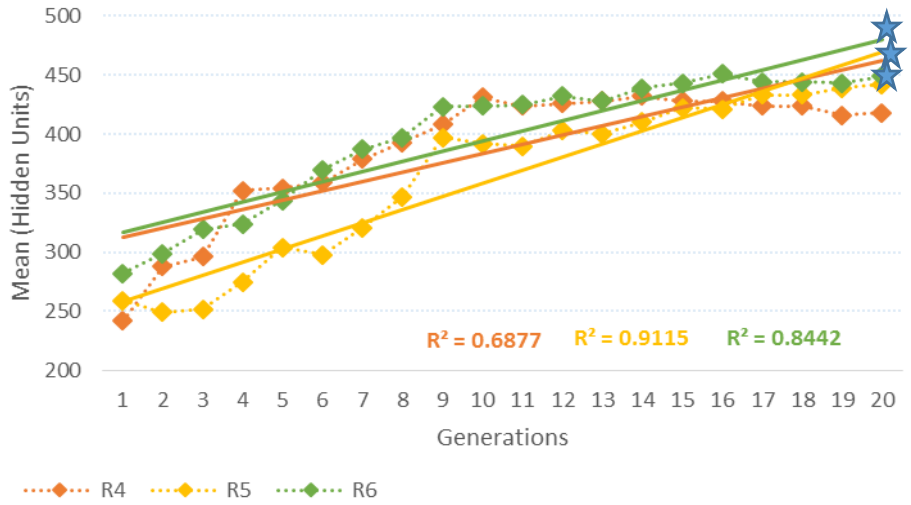


Figure 5.12(a): Change in the mean value of the number of hidden units per generation

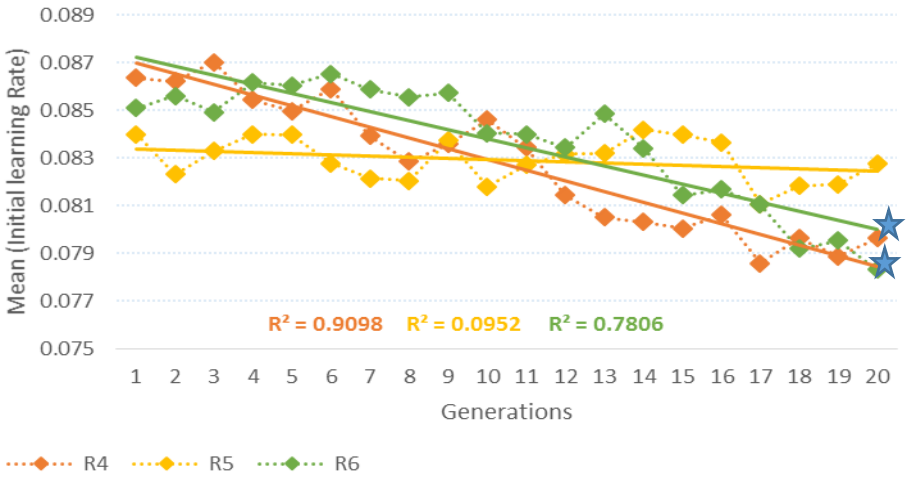


Figure 5.12(b): Change in the mean value of the initial learning rate per generation

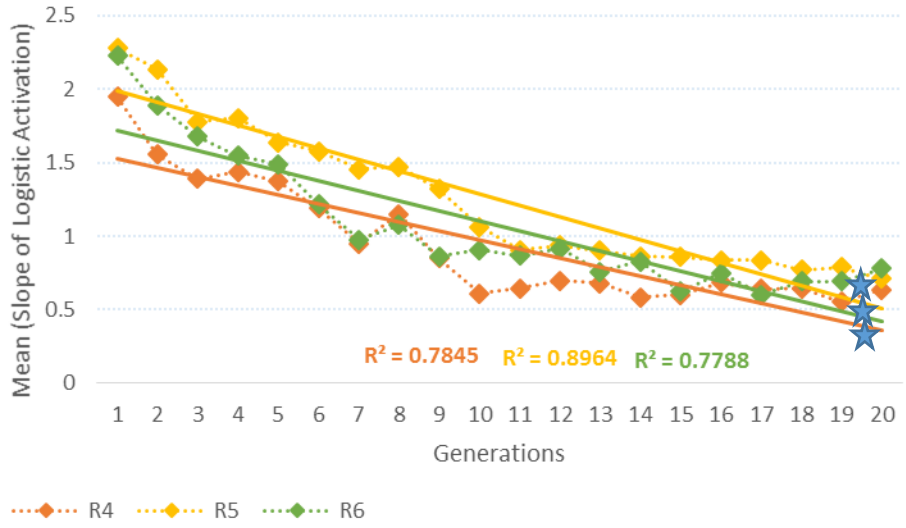


Figure 5.12(c): Change in the mean value of the slope of logistic activation per generation

There is an increase in mean values of number of hidden units and decrease in the mean values of slope of logistic activation in all three lineages. The range of variation of number of hidden units and the slope of logistic activation becomes narrower with generations. In the former case, the range moves towards the higher end of the spectrum while for the latter the range gradually settles at the lower end of range. The range of variation for initial learning rate does not show much variation. However, the mean values for initial learning rate have decreasing gradients in replications 4 and 6, however in replication 5 the gradient is quite constant. It is thus evident that truncation selection is targeting networks with more capacity and good learning ability i.e. increasing hidden units and neither too steep nor too shallow slope of logistic activation function.

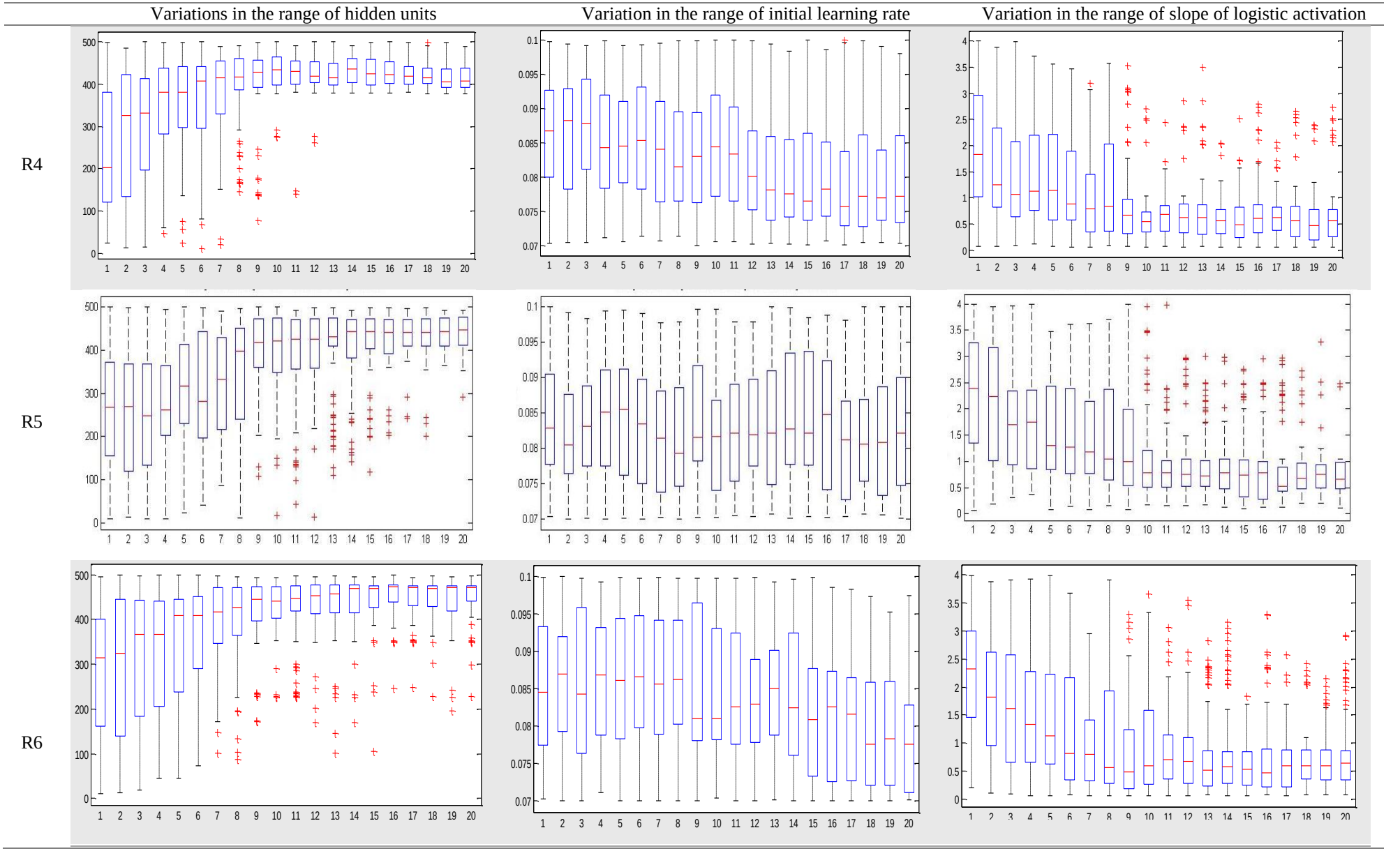


Figure 5.13: Range of Variation of Intrinsic parameters across Generations

The increase in number of hidden units provides networks with increased capacity to learn whereas the drop in values of slope of logistic activation from a very high value to a lower and more favourable values gives networks the ability to learn. These two parameters and their chosen/selected parameter range provides networks with enough capacity coupled with good ability to learn. The parameters' range of variation thus makes them domain-relevant i.e. these work well and benefit learning of all tasks ergo the improved accuracy for all tasks. This is an interesting finding since although these parameters were optimised based on performance on the source task, these optimised ranges turned out to be suited/beneficial for the remaining heterogeneous target tasks as well.

As discussed previously the heritability of categorisation with exceptions and auto association task increases in lineage 5 despite the improvement in accuracy. Mean values of initial learning rate of RPROP decreases in lineage 4 and 6 although the overall range of variation remains quite stable. However in lineage 5, the mean values of learning rate have a steady non-significant gradient centring on 0.083. This non decreasing trend in mean learning rate in lineage 5 could potentially cause increasing heritability for auto association and categorisation with exceptions tasks in lineage 5. This observation is supported by the opposite trends seen in lineages 4 and 6 for initial learning rate accompanied by decreasing heritability trends in those replications. Given that the remaining two parameters maintain the same trends for all three lineages, it is evident that range of variability for learning rate in lineage 5 does not meet the requirements posed by auto association and categorisation with exceptions tasks and ergo results in increasing heritability gradient.

5.6.1 Evaluating benefits of transfer

Replications 4, 5 and 6 were characterised by a quasi-regular source task and a deterministic selection operator used in conjunction with sexual reproduction. The accuracy levels achieved in these replications were quite good on all five tasks. As discussed previously in Section 5.5.1, the success of the framework depends on the following key features:

- Ability to learn different heterogeneous tasks whilst retaining performance on source task
- Avoiding negative transfer by assessing task relatedness
- Ability to find domain relevant range of variation for neurocomputational parameters

To analyse the performance of the proposed approach along these lines, two questions were formulated and the observations made from reported results were used to answer them. These are presented below:

Q1. Did the framework enabled the ANN twins' populations to learn multiple heterogeneous tasks?

The answer is, yes – the ANN twin populations successfully learnt i.e. performed accurately on all five heterogeneous tasks. Performance accuracy and generalisation ability plots in Figures 5.7 and 5.8, show that not only did the gradients increased over generations but also the accuracy levels achieved were considerably higher compared to RW selection based replication results. For instance, training accuracy on English past tense task varied between 74%-80% and that on auto-association task varied between 85%-75% in RW selection based lineages, in contrast, in truncation selection based lineages the accuracy on English past tense varied between 85%-90% and on auto-association accuracy gradient varied between 75%-95%. The mostly decreasing heritability gradients in Figure 5.9 also corroborate that ANN twin populations are getting optimised on various learning tasks despite being selected for English past tense acquisition. This further implies that truncation selection is targeting neurocomputational parameters within a range of variation which tends to make them domain relevant. An evidence of benefitting effect of transfer is the noticeably higher accuracy levels achieved in these lineages. We can thus conclude that transferring the '*ability to learn*' whilst using truncation selection (or selection of fittest members) tends to benefit different kinds of heterogeneous tasks even when being used with a quasi-regular natured source task.

Q2. Was the proposed method able to avoid negative transfer by assessing task relatedness and having a domain relevant range of variation for neurocomputational parameters?

The answer is yes - the accuracy and generalisation results in Figures 5.7 and 5.8 depict that all tasks had increasing accuracy gradients in all replications. The only exception was categorisation with exceptions task which had some decreasing gradients. However as discussed previously, although the gradients were decreasing in few lineages, nonetheless the accuracy levels were consistently very high. Thus it is safe to assume that negative transfer has been avoided in all cases and transfer has been successful. Further Figures 5.12 and 5.13 also show that truncation selection is clearly targeting parameters varying within specific section of entire range, thereby skewing the range of variation to certain ends of

spectrum for each parameter. Since, in truncation selection based lineages, accuracy levels achieved on all tasks are high, it implies that this change/shift in range of variation makes these neurocomputational parameters act in a domain relevant capacity and thus makes transfer a success. Similarly, heritability had decreasing gradients for all tasks in most lineages, implying optimisation. Also focusing on the actual range of variation of heritability values, we can see patterns emerging wherein heritability values for English past tense acquisition and categorisation vary within similar range and that of auto and arbitrary association vary within similar range. A noteworthy point is that although the trends observed for heritability gradients were quite different compared to those observed under RW selection setting, nevertheless the range of variation for heritability values was similar. In either case, the direction of heritability gradient informs whether or not transfer will be beneficial. Thus, heritability could be used in capacity of task relatedness assessment matric, wherein relatedness is not measured as per definitions given in Section 4.2. Instead it informs whether given tasks require same neurocomputational parameters (analogous to generalist genes) varying between similar ranges, in which case transfer will be beneficial.

5.7 Discussion

The results obtained by experimenting with over 120,000 neural networks in various settings have uncovered some interesting corollaries. The first observation is evolution (via selection) and learning (i.e. ANN training) interact throughout lineage and result in different overt behaviours. The aforementioned interaction is of a circular nature wherein selection provides ANN populations with the capacity and ability to learn and thus constrains the behavioural outcome i.e. accuracy levels. On the other hand, the performance levels attained after training (i.e. learning) determine fitness which in turn regulates what type of networks get chosen for breeding next generation members and thus in a way indirectly limits what type of intrinsic factors future generations will have.

Further, the type of selection operator being used, namely stochastic or deterministic, modulates the accuracy levels achieved by ANN populations. Performance accuracy levels achieved with truncation selection were much higher compared to accuracy levels obtained with RW selection. This implies that fitness based deterministic selection is a wise choice to make, which is true however, looking back at Chapter 3, Section 3.7.1, it is evident that this

approach has a slight downside as well. The irregular verb performance gradients in replications 4, 5 and 6 were positioned at considerably lower accuracy levels than those obtained under RWS setting. A potential explanation for this is that selecting only the fittest networks might result in missing certain types of mappings especially if there is class imbalance in the data set. This occurs because global fitness is usually driven by majority class fitness. However, stochastic selection enables selection of networks which possibly cover diverse areas of the training set and ergo are able to handle class imbalance.

Similar to the behavioural trends obtained in Chapter 3, it is evident that the effect of selection (owing to shift in range of intrinsic properties) on different tasks is consistent throughout the replication. For instance, accuracy gradients for a task might keep getting better, or stay invariable at a certain level or keep worsening over generations. Either way, once a behavioural trend emerges it continues that way throughout that replication, there isn't any reversal in that trend, much like Waddington's epigenetic landscape discussed in Chapter 3. This behaviour is not really wished for in machine learning especially if performance starts worsening. It then becomes similar to being stuck in a local minima and there should be a way to reverse the trend. This is where the analyses of proportion of variance due to genetic and environmental factors become more relevant. These analyses revealed which of these neurocomputational or environmental factors caused most behavioural variance and consequently informed us which of them is exploited most by ANNs for acquiring a certain task. Thus training could be biased towards the more important/contributing factor to boost performance accuracy.

Next, the results revealed that heritability acts as an identifier of task relatedness. For clarity, the term task relatedness is not used as described in Chapter 4, Section 4.2. Instead in this context it means that the said tasks target same neurocomputational parameters varying within similar ranges of variation. Consequently the chances of improvement in accuracy are enhanced if selection is acting on one of these tasks and thus it is easier to predict if transfer will be successful and thereby avoid negative transfer. A downside of heritability as a metric of task relatedness is that the method needs to be tried for a few generations in order find the emerging behavioural and heritability trends. However, in more real-world applications which possibly involve thousands of generations and computations are costly, heritability as an indicator of task relatedness would still come handy because after only a few trials trends start to emerge and they will remain that way throughout the lineage. Thus if negative transfer

occurs, further transfer could be stopped or some steps (like biasing training) could be taken to prevent that from happening.

5.8 Summary and contribution of chapter

This chapter presented a behavioural genetics inspired transfer approach capable of performing heterogeneous transfer. The proposed evolutionary computational approach enabled populations of ANNs to acquire five heterogeneous tasks of cognitive nature. These tasks differed in terms of degree of similarity between input-output mappings and the presence of structure and regularity in mapping. Over 120,000 ANNs were trained as part of experimental evaluation process spanning 6 replications, comprising two different selection metrics. The transfer method builds on the premise of generalist genes (Kovas and Plomin, 2007) according to which the same set of genes are responsible for diverse learning and cognitive abilities. Additionally, research in psychology shows that genes and environment interact continuously throughout development to shape differences in overt behaviours. Considering the aforementioned principles, in this work the effects of genes were implemented via variations in neurocomputational parameters of ANNs encoded in an artificial genome and the effect of environmental factors were simulated via filter applied to the training sets (thereby simulating shared environment) and unique initial weights of ANNs (thus simulating non-shared environmental factors). In order to factor in the generalist genes effect, the chosen neurocomputational parameters had general computational functions and no specific relation to any given problem domain. The combination of genes + environment provides ANNs with the ability to learn any behaviour/task and the method transferred the '*ability to learn*' from source to multiple target tasks. In addition to performing heterogeneous transfer, the method also includes heritability, wherein the direction of change and the range of variation of heritability values act as an indicator of task relatedness. Heritability is a useful statistic because it is scalable across potentially very large numbers of computational parameters (and their interactions) that contribute to the variation in learned high-level behaviours, or in this case, the outcome of learning for a set of ANNs. However, in the current simulations, relatively few parameters were encoded in the genome and permitted to vary across populations and between generations. Having lots of computational parameters gives better chance of finding domain-relevant properties. The proposed method enables inclusion of any number of

neurocomputational parameters. In a pilot study/experiments reported in (Kohli et al., 2013) we used five parameters and the heterogeneous transfer approach worked well. As part of future extension of this work, more complex genome will be used.

Chapter 6 Experimental evaluation of BG inspired Transfer Learning framework: switching source tasks

6.1 Overview

This chapter presents the experimental evaluation of the BG inspired transfer framework. In the previous chapter the experiments focussed on two different types of selection operators and examined their impact on success of transfer. Next in line of analysis is the effect of source task on transferability, especially since the chosen tasks are heterogeneous. Therefore, this chapter focuses on the experiments analysing the effect of source task on transfer. This chapter is organised as follows: Section 6.2 explains the experiment design and Sections 6.3-6.6 describe the effects of switching source tasks on transfer, wherein each section is dedicated to a different source task. Section 6.7 presents the discussion of results and finally summary and contribution of chapter is presented in Section 6.8.

6.2 Experiment Design

To investigate the effectiveness of the transfer approach with diverse source tasks, further four replications of 20 generations each were carried out, wherein each lineage had a different source task. The heterogeneous tasks used were the same and have been explained in Chapter 5, Sections 5.2 and 5.3. The selection operator was kept the same in all four replications – truncation selection, because it resulted in better performance accuracy in all previous replications it was used. Table 6.1 summarises the experiment design used in each of these four replications (denoted by 'R').

No. of replications	4 (R_7 – R_{10})
No of Generations per replication	20
Size of population	Breeding = 100; Non-breeding= 100 Total R_7+R_8+R_9 +R_{10} across generations= 16,000 ANNs per task
Size of Datasets	Training= 500 { 508(for past tense)} Generalisation= 500
Training Mode	Batch
Max. training epochs	100 (Past tense, categorisation & categorisation with exceptions) 500 (Auto & Arbitrary)

Early Stopping Criterion, maxstep (i.e. stop training if training accuracy does not improve till step == maxstep)	20 (English past tense, categorisation and categorisation with exceptions) 50 (Auto & Arb)
Initial weight update (Rprop learning rate)	Values from genome
Hidden units, Steepness of logistic	Values from genome
Selection Operator	Truncation - applied at the end of training
Source Task	R_7 : Arbitrary association; R_8 : categorisation w/exception; R_9 : auto association; R_{10} : categorisation
Crossover	6 crossovers/chromosome; single-point, multi-point & shuffle operators used
Environmental Factor (SES)	Probability value between 60% and 100%
Range of encoded neurocomputational parameters	No. of hidden units (10 – 500); initial learning rate (0.7 – 1.0); slope of logistic activation (0.0625 – 4.0)

Table 6.1: Experimental Design for replications 7 – 10: analysing effects of switching source task

6.3 R_7, Source task: arbitrary association

In this replication we switched source task from quasi-regular English past tense acquisition to arbitrary association – a task characterised by completely random mappings. Due to the lack of any systematic mapping between input and output, there isn't anything in particular that the networks could possibly extract and '*learn*'. Consequently the only feasible tactic to acquire such a task is to perform rote learning. Rote learning or cramming as it is more commonly known as, should require networks with abundant capacity so that networks are able to store the random input-output mappings. The randomness of this task makes it an interesting choice of source task especially since it raises the question – is it possible to *learn* anything which is essentially random? How will optimising on randomness affect performance on other target tasks? The experiments in this lineage reveal answers to these questions. Figure 6.1 depicts the performance accuracy on full training set and generalisation accuracy achieved on each of the five tasks.

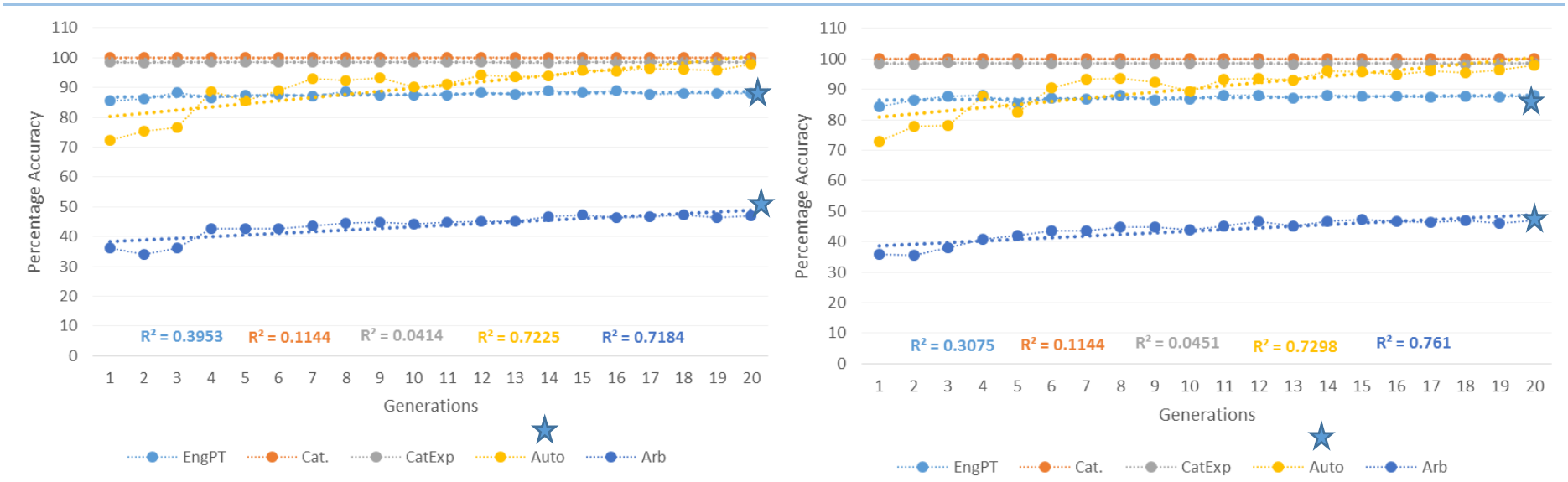


Figure 6.1 (a): Mean performance per generation for breeding (left) and non-breeding (right) twin populations with Arbitrary association as source task

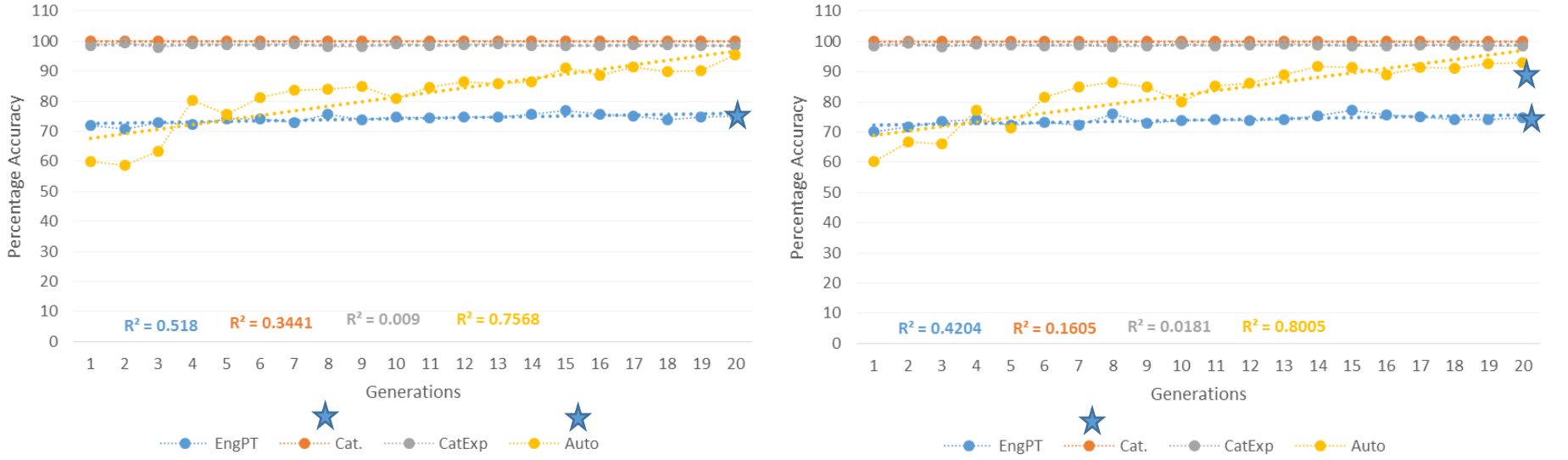


Figure 6.1(b): Mean generalisation accuracy per generation for breeding (left) and non-breeding (right) twin populations with Arbitrary association as source task

From Figure 6.1 it is evident that accuracy levels achieved on all five tasks are good. Accuracy on source task, i.e. arbitrary association improves significantly and reaches nearly 50% accuracy levels towards the end of lineage which is quite noteworthy. Learning pattern mappings depends to some extent on the interaction between regularity in mappings and frequency of occurrence (of pattern types i.e. class balance) and this source task lacks both. In addition, networks are exposed to filtered training dataset which contains between (60%-100%) of patterns only depending on network' SES based filter. Thus, random input-output mappings and filtered training sets make it less likely for networks to achieve accuracy levels of more than 50% on this task. Similarly, the target tasks also displayed reliably improving performance gradient especially English past tense acquisition and auto association. Categorisation and categorisation with exceptions once again had very high accuracy levels and approximately stable gradients.

Similar results were obtained for generalisation ability tests also wherein all tasks achieved high accuracy levels and had improving performance gradients. These results suggest that optimising a population of ANN twins on arbitrary association task and transferring the resulting 'ability to learn' to acquire other learning tasks has proven beneficial.

From previous results we expect that improving performance gradients should be accompanied with decreasing heritability gradients (in most cases). To investigate this, we calculated and plotted the proportion of variance in performance attributed by genetic and environmental factors as shown in Figure 6.2. Since, the population members demonstrated ceiling effects in categorisation task, thus there were no performance variations. Therefore, the heritability and environmentability values were non-computable. The R^2 values for heritability and environmentability for categorisation task are marked N/A to reflect this.

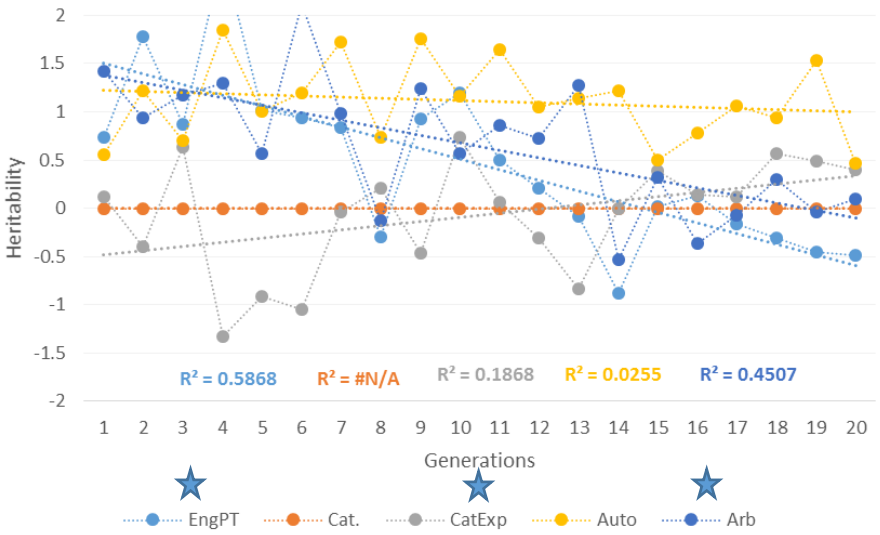


Figure 6.2(a): Proportion of variance due to genetic factors i.e. heritability – Arbitrary association source task

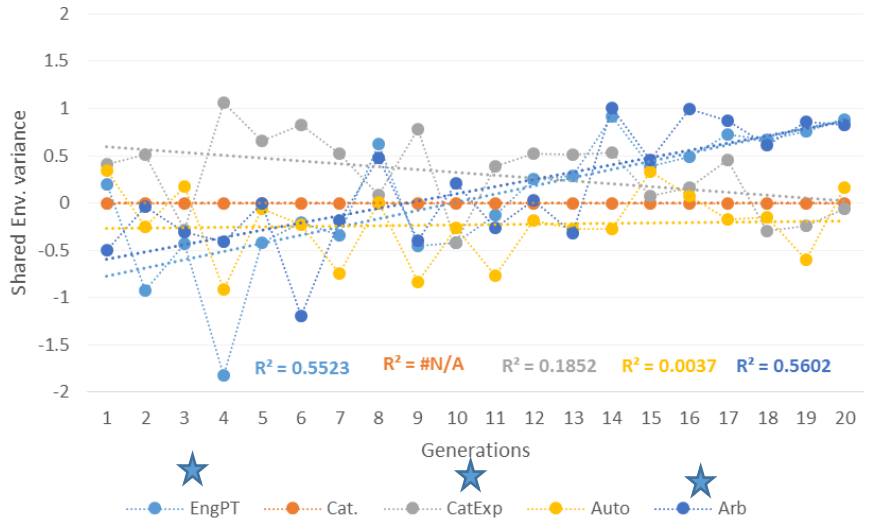


Figure 6.2 (b): Proportion of variance due to shared environmental factors – Arbitrary association source task

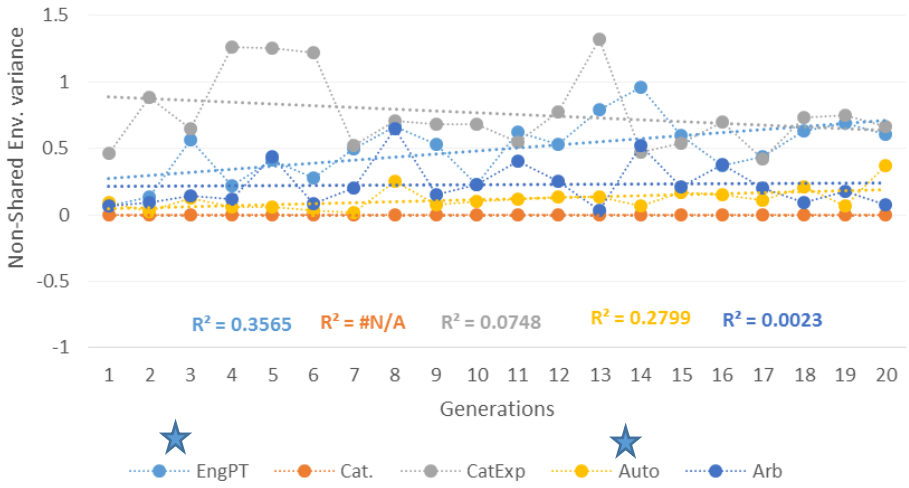


Figure 6.2 (c): Proportion of variance due to non-shared environmental factors – Arbitrary association source task.

The heritability for arbitrary association (i.e. source task) and English past tense acquisition task decreases considerably from very high values to approximately nil. This signifies optimisation in both these cases, which is in fact also substantiated by the negative accuracy-heritability correlation values. Auto association task also experiences a non-reliable (or statistically non-significant) and gradual drop in heritability through the generations. This is indicative of ANN population' strong genetic dependence for learning auto association task and that small genetic differences result in considerable performance variations. Categorisation task had no performance variation (refer Figure 6.1) and correspondingly has nil heritability. Categorisation with exceptions is the only task with reliably increasing heritability gradient with values increasing from negative to moderate.

Based on the heritability results we can see three groups emerge in terms of closeness of heritability values, first one comprising arbitrary association and acquisition of English past tense tasks (these results corroborate with those obtained in truncation selection results), the second group has categorisation and categorisation with exceptions task and auto association falls in the third group. These groups signify task relatedness in terms of their genetic/neurocomputational dependence on learning given task. Additionally, it also helps us deduce tasks for which the targeted neurocomputational range of variation acts in a domain relevant capacity. All of these act as pointers for determining transfer success.

The proportion of variance not accounted for by genetic factors depends on variations in environmental factors. Figure 6.2 (b) depicts the proportion of variance due to differences in shared environmental factors i.e. filtered training sets. Arbitrary association and English past tense acquisition tasks have steadily increasing gradients thereby indicating that variations in accuracy levels achieved are considerably attributed to differences in filtered training sets of networks. The gradient for categorisation task is positioned at zero throughout owing to no variation in accuracy. Categorisation with exceptions task begins with moderate values for shared environment influence but this reliably decreases over generations ending with almost nil contribution. This suggests that initially variations in performance were more due to differences in training sets however this effect diminishes over generations.

Figure 6.2 (c) depicts the proportion of variation due to differences in non-shared environmental factors i.e. initial weights of ANNs. Differences in initial weights modulate performance variations reliably in case of auto association and English past tense. However, for the rest the effects were non-significant. From these results it is evident that although

performance accuracy improves/maintains at a good level for all tasks, yet the factors contributing to performance variation (and thus more important for acquiring the task) are different. Calculating these proportion of variances informs us about possible biases that could be introduced whilst training to improve accuracy if need be for a given task.

In the final leg of analysis for this task, we analyse which neurocomputational parameters and the range of variation is being targeted by selection. Figure 6.3 and 6.4 depict the changes in the mean values of intrinsic parameters and the range of variation over generations. The first observation from these figures is the significant increase in number of hidden units and reduction in the slope of logistic over generations. The mean value for initial learning rate also decreases though more steadily with no noticeable change in the range of variation. These results indicate, that despite having a different source task, selection is still targeting greater number of hidden units and shallower slope of logistic activation function. These parameters act in a domain relevant capacity and provide networks with capacity and ability to acquire different tasks even if they are heterogeneous with respect to source task.

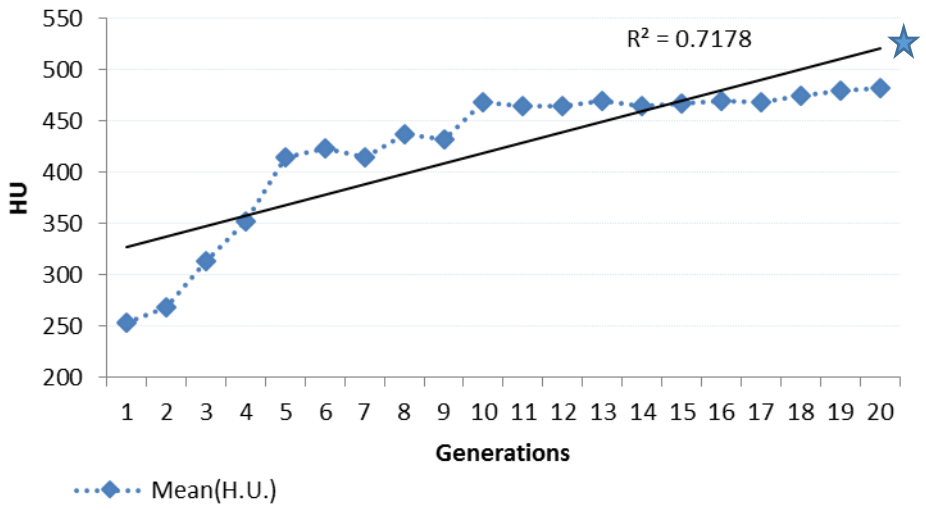


Figure 6.3 (a): Change in the mean value of the number of hidden units per generation

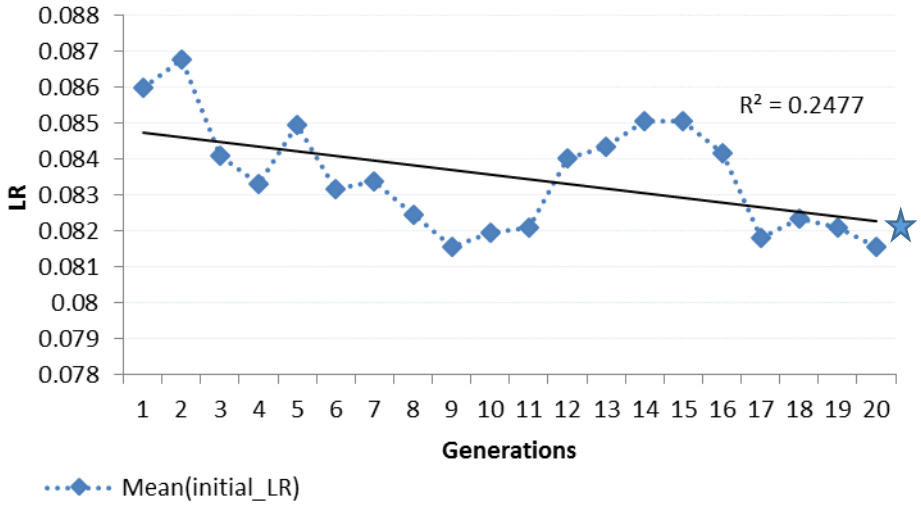


Figure 6.3 (b): Change in the mean value of the initial learning rate per generation

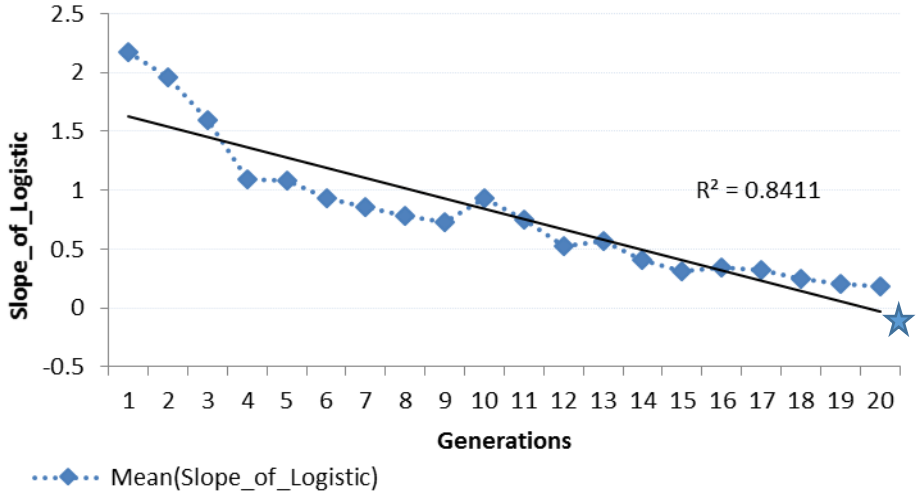


Figure 6.3 (c): Change in the mean value of the slope of logistic activation per generation

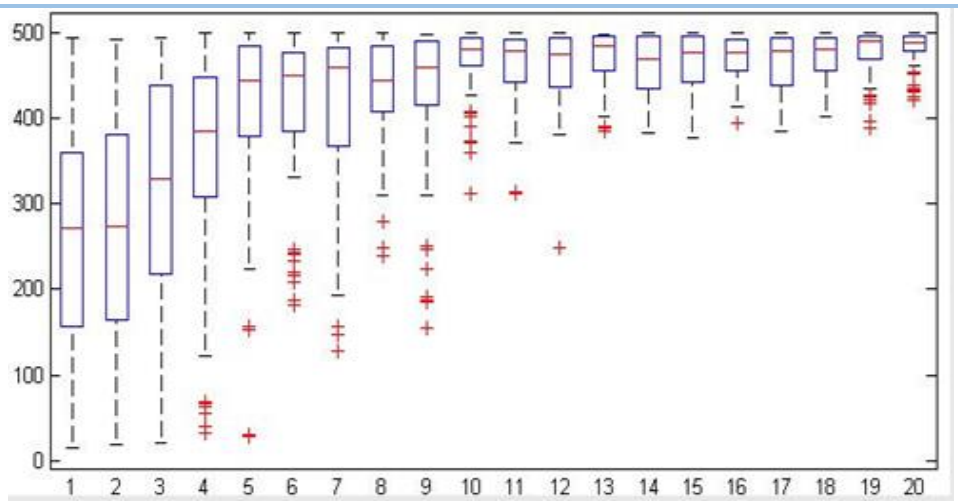


Figure 6.4 (a): Variations in the range of number of hidden units over generations

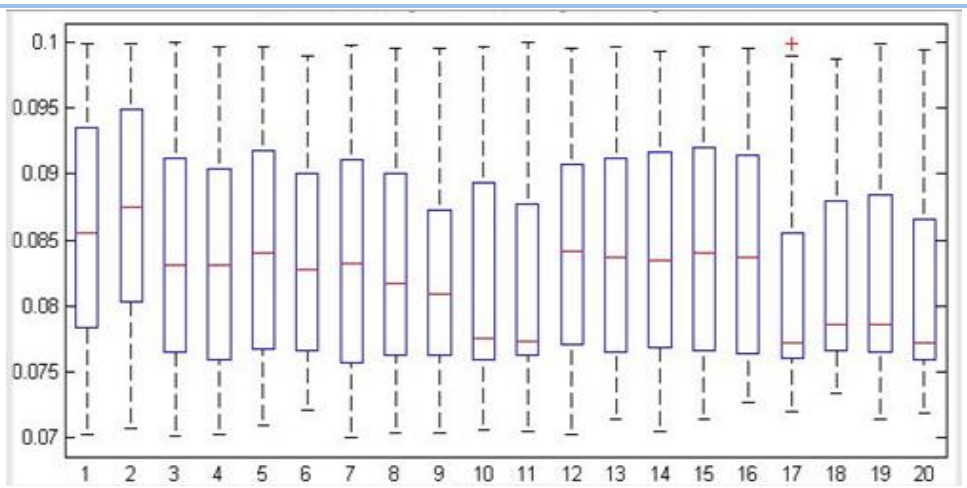


Figure 6.4 (b): Variations in the range of the initial learning rate over generations

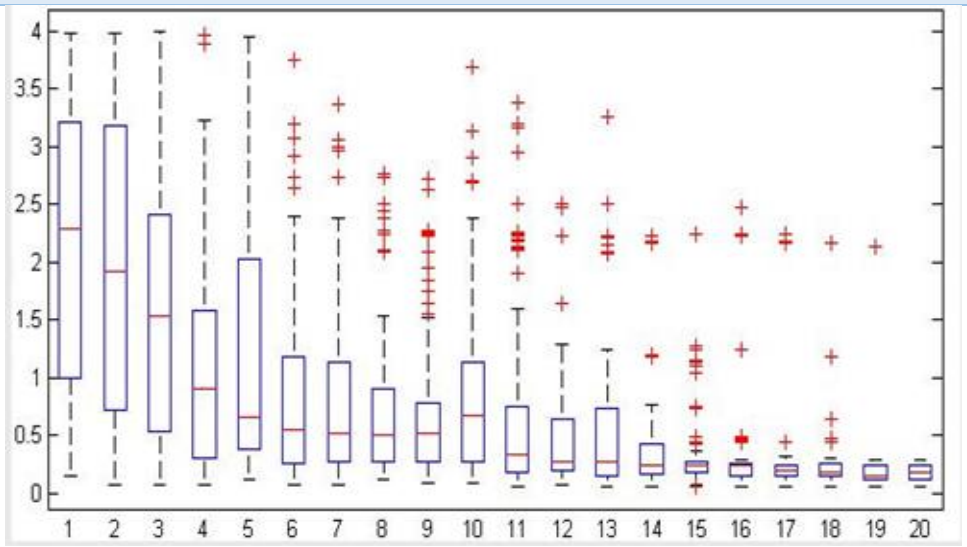


Figure 6.4 (c): Variations in the range of the slope of logistic activation over generations

6.3.1 Evaluating benefits of transfer

Finally, to analyse the benefits of transfer the following two questions were addressed.

Q1. Did the framework enabled the ANN twin' populations to learn multiple heterogeneous tasks?

Yes - the ANN populations acquired different heterogeneous tasks successfully.

Q2. Was the proposed method able to avoid negative transfer by assessing task relatedness and having a domain relevant range of variation for neurocomputational parameters?

Yes – the proposed method avoided negative transfer. The range of variation of intrinsic parameters targeted by selection made them act in a domain relevant capacity and hence transferring 'the ability to learn' through shared genome and shared environment

allowed the ANN populations to learn different heterogeneous tasks through a common learning framework.

6.4 R_8, Source task: categorisation with exceptions

In this replication we switched source task from arbitrary mappings to categorisation with exceptions. This task is similar to consistent categorisation task, however it has one distinguishing feature, the membership of some patterns in a particular category come about by extension. To elaborate, the networks have to learn to assign input patterns to different categories based on their similarity to a prototype pattern for each category. However, a small set of input patterns are exceptions to this rule. Based on some chosen condition, these exceptional patterns are assigned to a category which is different from the one corresponding to the more similar prototype pattern. Thus in order to learn and optimise performance in this task, the population of networks have to understand and learn the more pervasive similarity based mappings but also the counter intuitive exceptions mapping.

Figure 6.5 shows the training and generalisation accuracy achieved by ANN twin populations in this lineage on all five tasks. The accuracy achieved on target tasks i.e. English past tense acquisition, arbitrary association and auto association decreases significantly over generations whereas the accuracy was stable through the lineage for categorisation task. The performance on the source task itself had a non-reliable (statistically non-significant) static gradient varying within 98.6% to 98.5%. Similar trends occurred for generalisation tests also.

The observation drawn from these results suggests that transferring ability to learn with categorisation with exceptions as the source task has not been successful and has led to negative transfer instead. To investigate the reasons behind this we compute the proportions of behavioural variances due to genetic and environmental factors. Figure 6.6 depicts these results which exhibit some counter-intuitive trends.

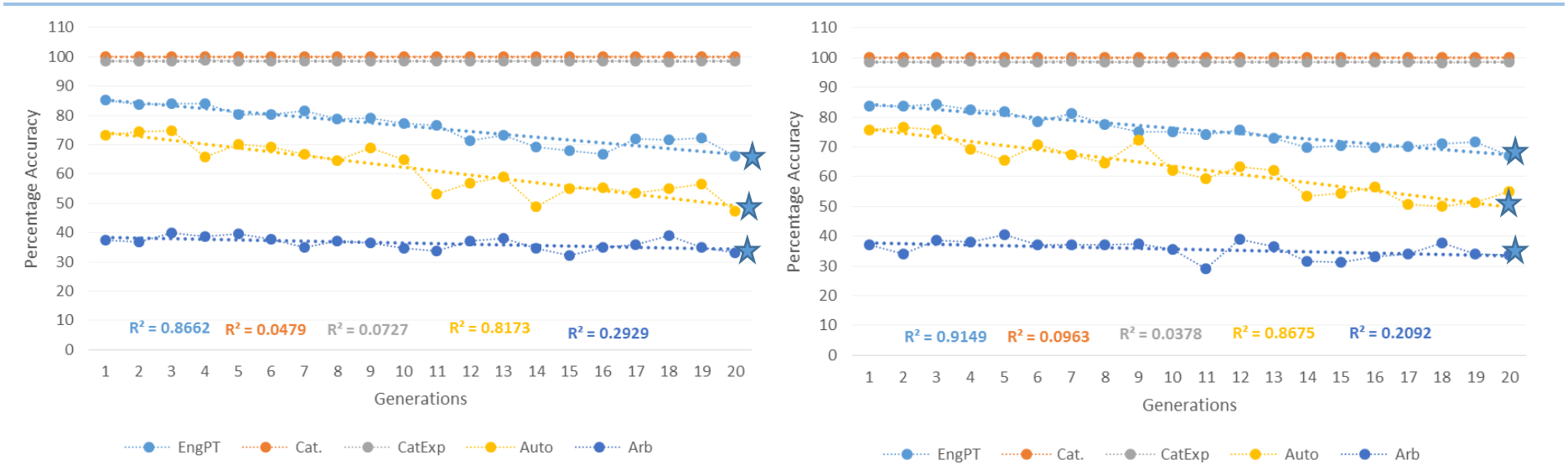


Figure 6.5 (a): Mean performance per generation for breeding (left) and non-breeding (right) twin populations with Categorisation w/exceptions as source task

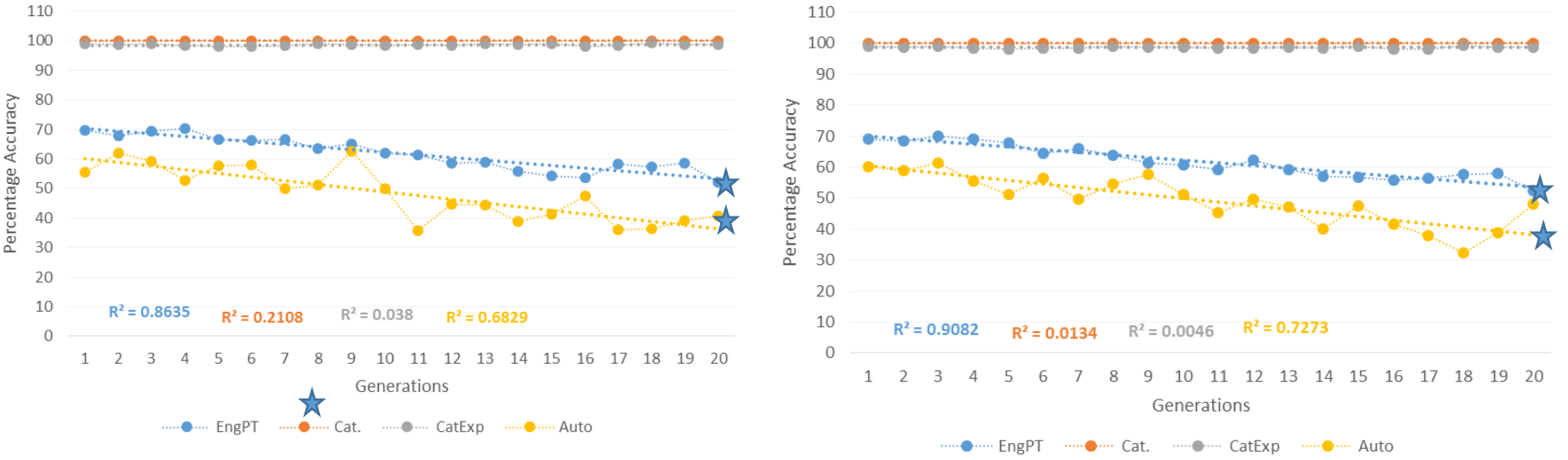


Figure 6.5 (b): Mean generalisation accuracy per generation for breeding (left) and non-breeding (right) twin populations with Categorisation w/exceptions as source task

The first in the line of counter-intuitive trends is the increasing heritability gradient for source task i.e. categorisation with exceptions. The ANN population members are being selected based on their fitness on the source task and the selection metric being used is deterministic and thus selects only the fittest members, consequently the behavioural accuracy should ideally improve over generations. Additionally, selection should skew the range of variation of certain intrinsic parameters being targeted by selection thereby making the range narrower which ultimately should have led to reduction in heritability values. However as we notice from the performance plots this is not the case. Since there is not any real variance in performance accuracy, selection is random and thus there isn't any real optimisation in this lineage. The heritability gradients for all tasks except English past tense were non-significant. However, the accuracy-heritability correlation for the source task, categorisation with exceptions was negative, thereby implying an inverse heritability-optimisation relationship.

Continuing with the counter-intuitive trend is the gradient for English past tense acquisition and arbitrary association tasks which showed a reliable decrease throughout lineage despite having a decreasing accuracy gradient. However the accuracy-heritability correlation of 0.41 suggests that the inverse relationship between optimisation and heritability does not hold true in this case. The remaining heritability gradients were not reliable.

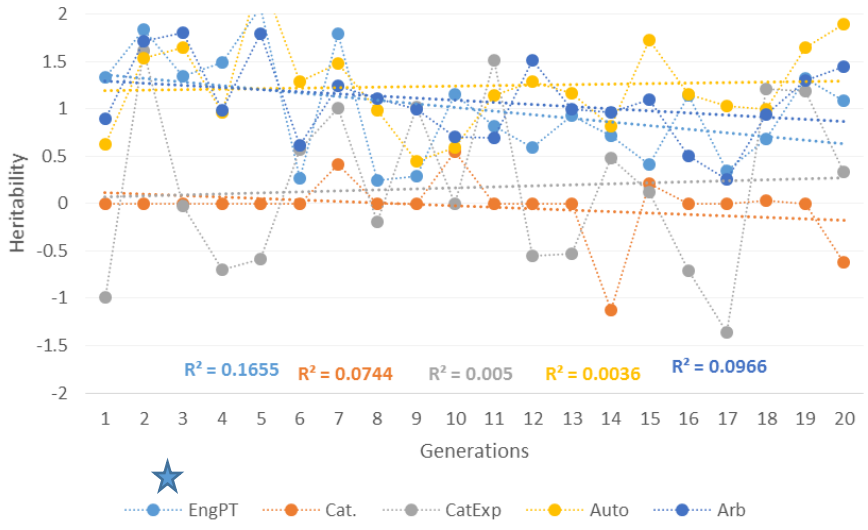


Figure 6.6 (a): Proportion of variance due to genetic factors i.e. heritability – Categorisation w/exceptions source task

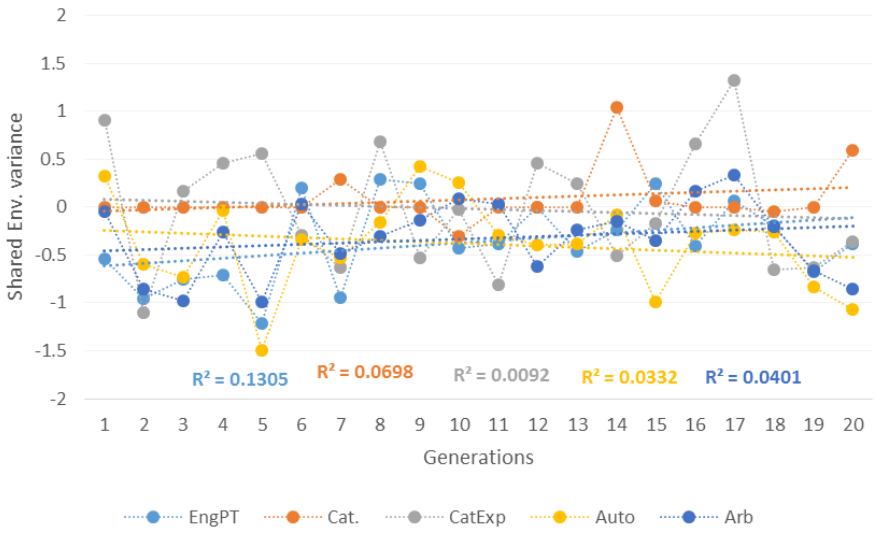


Figure 6.6 (b): Proportion of variance due to shared environmental factors – Categorisation w/exceptions source task

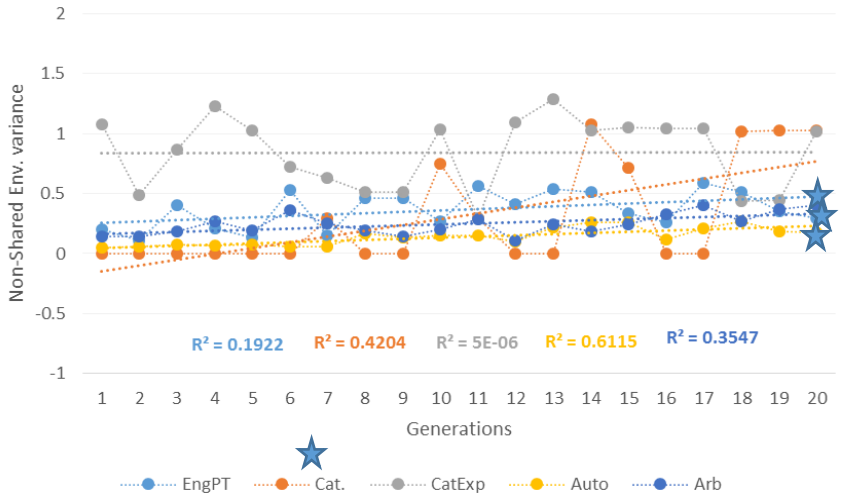


Figure 6.6 (c): Proportion of variance due to non-shared environmental factors – Categorisation w/exceptions source task.

Figure 6.6 (b) represents the proportion of behavioural variance due to differences in shared environment i.e. filtered training sets. It is quite evident from the graph that shared environmental factors do not account for any noticeable performance variation in any task. However, non-shared environmental factors as shown in Figure 6.6 (c) account for significant behavioural variance in all tasks except the source task. This shows that ANNs are highly dependent on their initial weight values for acquiring the said task and even a small difference in weight values leads to huge differences in performance. This high dependence on weight values could partly explain the lower dependence (or contribution of) on intrinsic factors which in turn results in increasing heritability.

Finally, we evaluate the changes in the genome occurring whilst selection is acting on categorisation with exceptions task. Figures 6.7 and 6.8 depict the changes in the mean values of intrinsic parameters and the range of variation through the lineage. The first observation drawn from these graphs is the increase in the number of hidden units over generations. This increase in capacity, as we know from previous results, acts in a domain-relevant capacity especially for English past tense acquisition and arbitrary association tasks. This explains why these two tasks have a decreasing heritability gradient despite experiencing worsening performance accuracy. The next observation is the increase in the mean values of learning rate and slope of logistic activation function. However the spread of values remains quite uniform over the original range in case of learning rate whilst the range of variation becomes skewed towards the higher end of range for slope of logistic activation. This increase explains why the performance worsens in this replication for some tasks. The slope of logistic activation is becoming steeper with generations and this is taking away the ability of networks to learn as it approximates the threshold thereby generating a unit with near binary response features which retards learning.

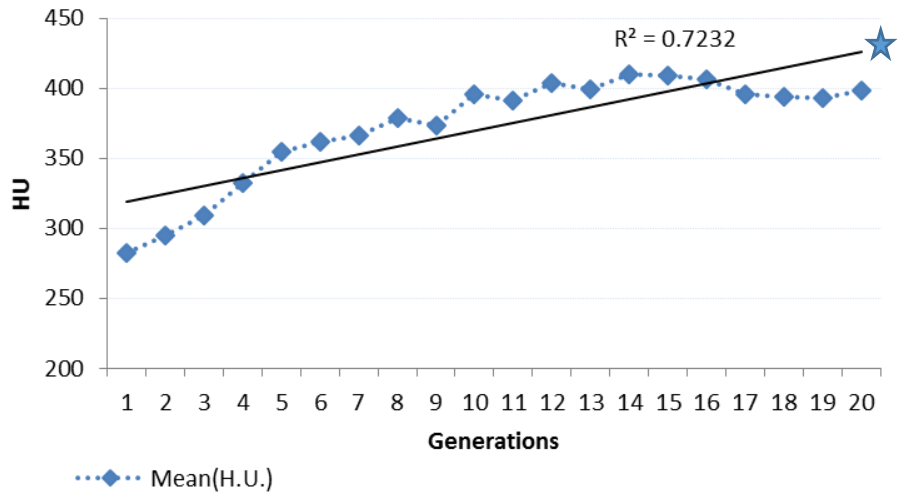


Figure 6.7 (a): Change in the mean value of the number of hidden units per generation

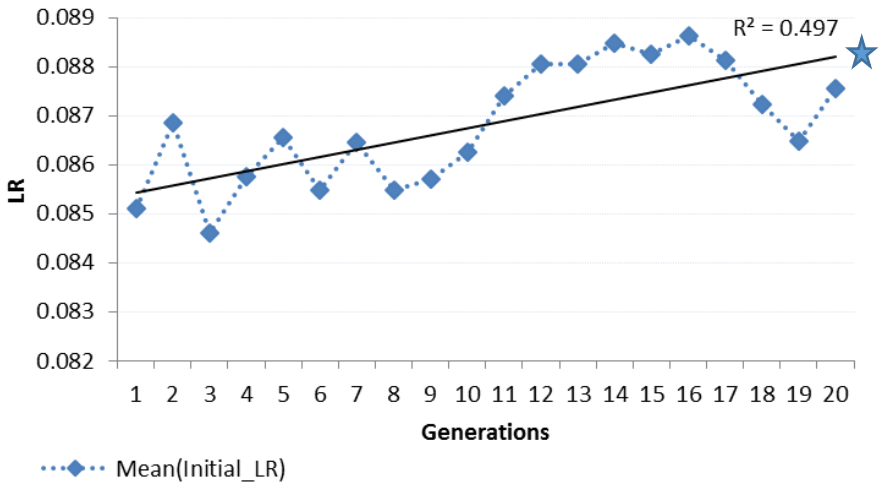


Figure 6.7 (b): Change in the mean value of the initial learning rate per generation

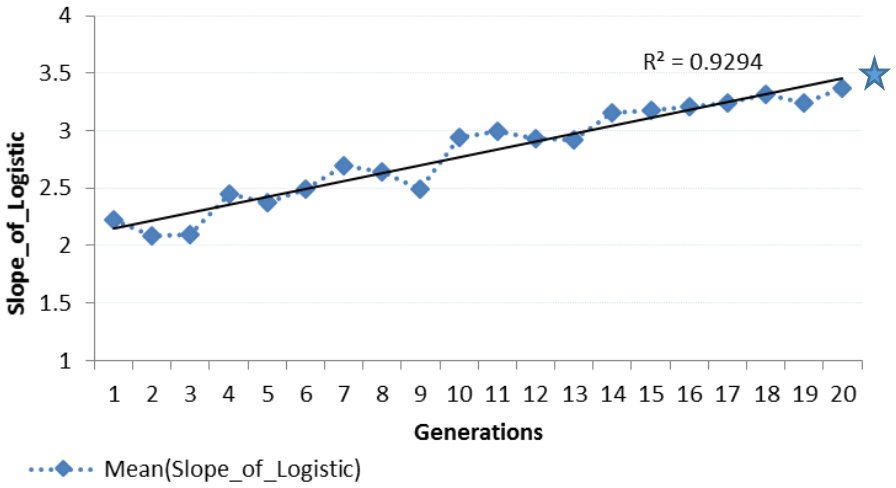


Figure 6.7 (c): Change in the mean value of the slope of logistic activation per generation

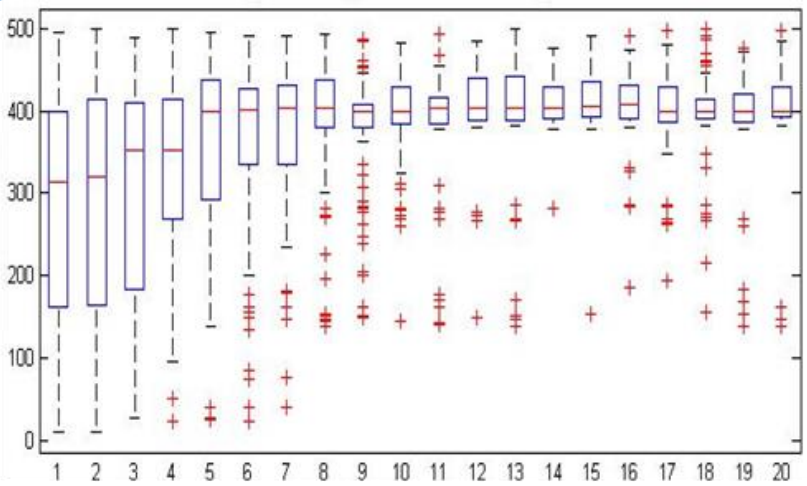


Figure 6.8 (a): Variations in the range of number of hidden units over generations

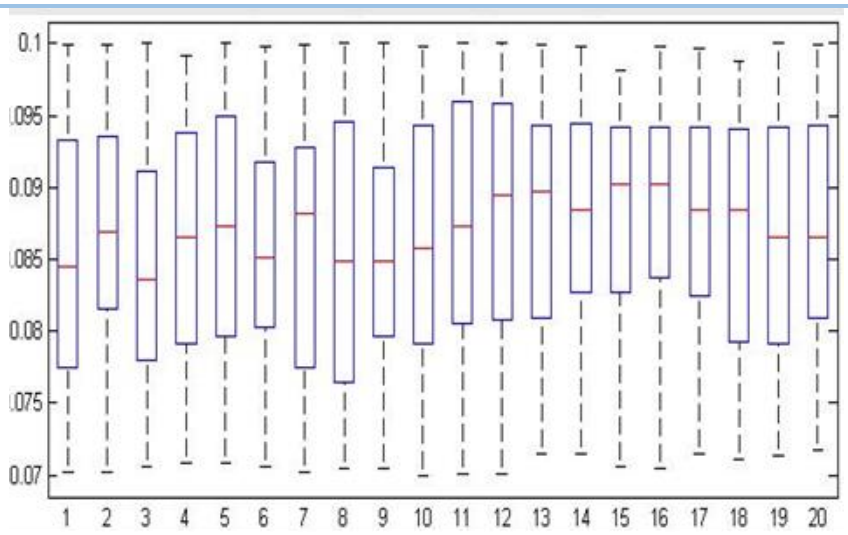


Figure 6.8 (b): Variations in the range of the initial learning rate over generations

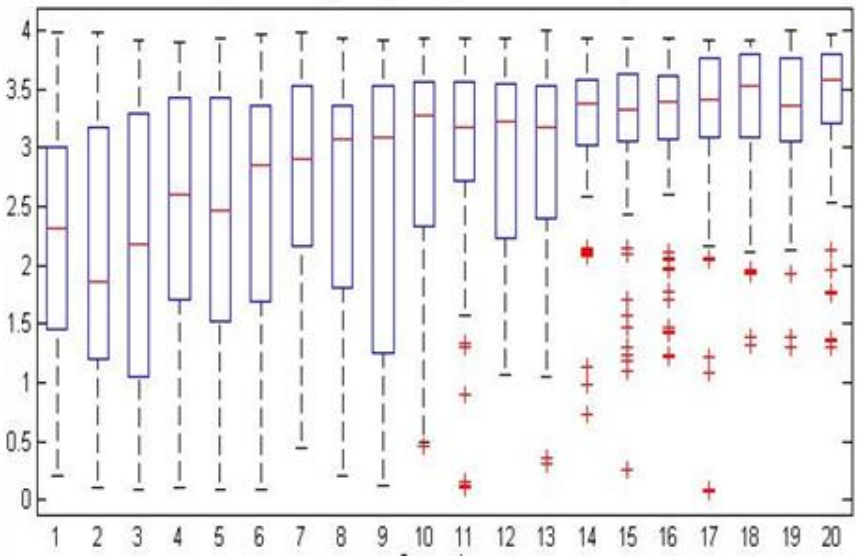


Figure 6.8 (c): Variations in the range of the slope of logistic activation over generations

So the increase in number of hidden units which acts in a domain relevant capacity results in decreasing heritability for some tasks, however it is unable to counter the retarding effects of increase in the values of slope of logistic and learning rate which consequently results in decreasing performance over generations. Overall it can be inferred that range of variation of intrinsic parameters being targeted by selection whilst acting on categorisation with exceptions task isn't beneficial for transfer.

6.4.1 Evaluating benefits of transfer

To analyse the benefits of transfer, the following two questions were addressed.

Q1. Did the framework enabled the ANN twin' populations to learn multiple heterogeneous tasks?

No – the performance worsened for most target tasks. In fact there wasn't any improvement in the source task either, but the populations maintained their high levels of accuracy throughout the lineage. This is owing to no variation in performance and as a result selection acts randomly i.e. it is unable to favour/target parameters varying within a specific range of variation. Consequently there is no real scope of optimisation for this particular task and ANNs' performance is as good as it can be.

Q2. Was the proposed method able to avoid negative transfer by assessing task relatedness and having a domain relevant range of variation for neurocomputational parameters?

No – negative transfer occurred in this instantiation of transfer framework. The range of variation of intrinsic parameters targeted by selection made them impractical for acquiring most learning tasks. In fact networks relied more on their initial weight values for acquiring the source task compared to genetic dependence.

6.5 R_9, Source task: auto association

In the next instantiation of the framework, auto association was chosen as the source task. This task is representative of cognitive imitation and normally learning it does not require description and understanding of underlying mechanisms. Instead networks just need the representation of stimulus that is target of imitation. This infers that ANNs have to learn to produce the exact same output code as the one presented in the input layer. Figure 6.9 describes the mean performance accuracy and generalisation accuracy achieved by ANN populations in the current lineage.

Figure 6.9 demonstrates that in the current replication, ANN populations achieved high accuracy levels on all tasks. English past tense acquisition, auto association and arbitrary association were marked with steeply and significantly increasing gradients whilst categorisation and categorisation had almost invariable (but with subtle increase) and statistically non-significant gradient varying above 98% accuracy levels for the latter and cent percent for the former. Thus, the first conclusion drawn is that using auto association as source task has resulted in effective transfer and negative transfer has been avoided in all cases.

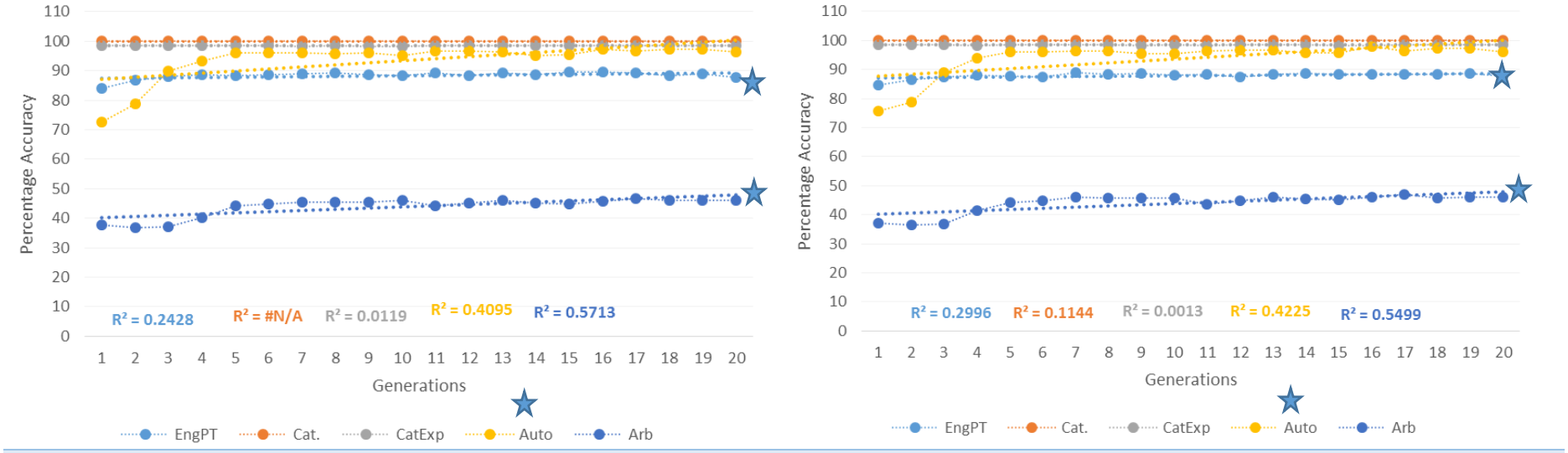


Figure 6.9 (a): Mean performance per generation for breeding (left) and non-breeding (right) twin populations with auto association as source task

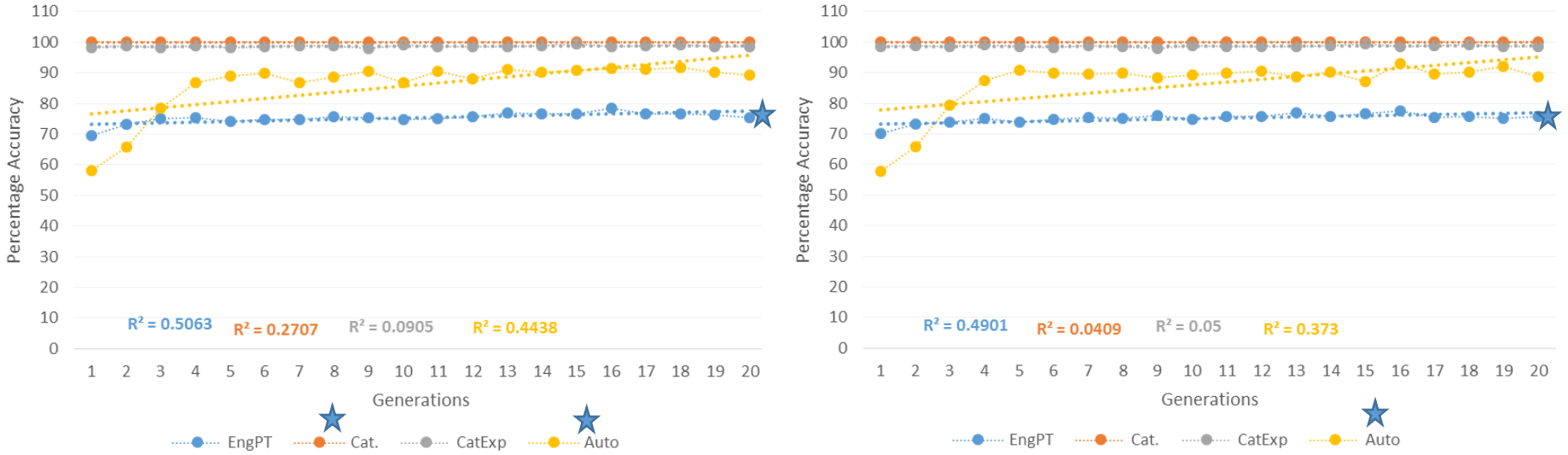


Figure 6.9 (b): Mean generalisation accuracy per generation for breeding (left) and non-breeding (right) twin populations with auto association as source task

Since the performance results clearly indicate optimisation occurring for all tasks in this lineage, one should expect to see decreasing heritability gradients complementing the optimisation. Figure 6.10 depicts the proportion of behavioural variance accounted for by genetic and environmental factors.

Figure 6.10 (a) depicts that heritability for all tasks barring categorisation decreases reliably from relatively moderate-to-high values to close to zero values. The heritability for categorisation task is fixed at nil due to no performance variation. The decreasing heritability gradients and the corresponding increasing behavioural accuracy gradients reiterate that optimisation and heritability are in fact inversely related. Due to lack of performance variation in categorisation task, the heritability and environmentability values were non-computable and hence the R^2 values are shown as N/A in graphs below.

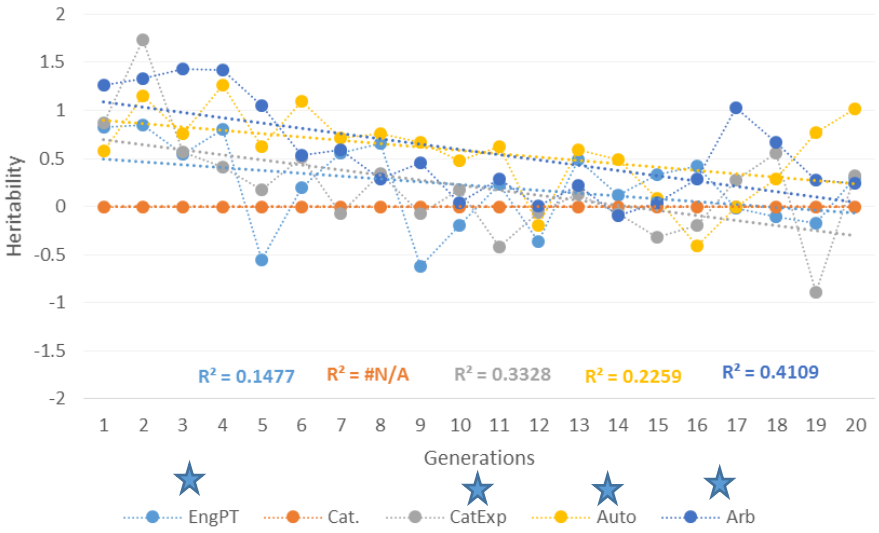


Figure 6.10 (a): Proportion of variance due to genetic factors i.e. heritability – auto association source task

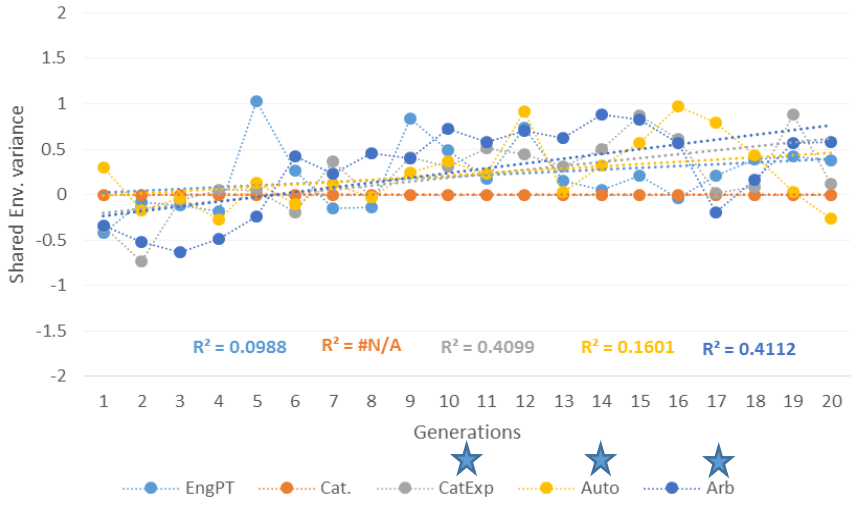


Figure 6.10 (b): Proportion of variance due to shared environmental factors – auto association source task

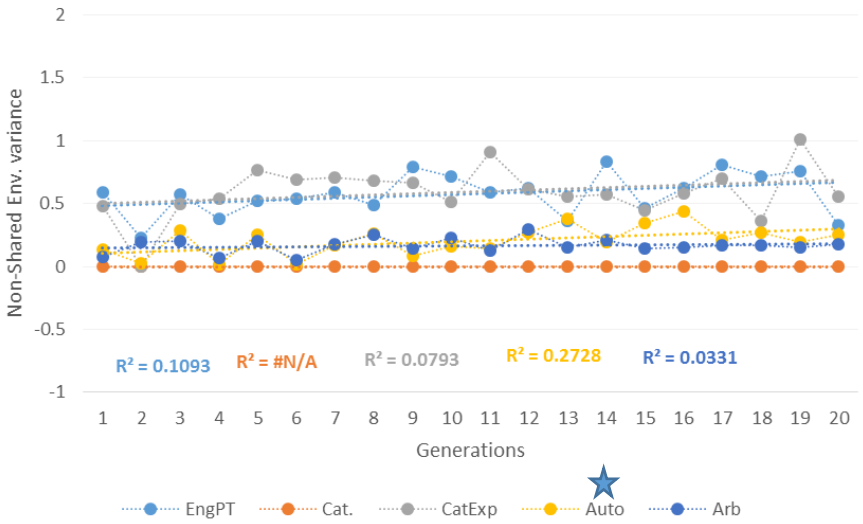


Figure 6.10 (c): Proportion of variance due to non-shared environmental factors – auto association source task.

Figure 6.10 (b) depicts the proportion of behavioural variance due to differences in training sets. The shared environmental gradients exhibit an increasing trend for all tasks except categorisation for which gradient is positioned at zero, although these effects were reliable only for auto-association, arbitrary-association and categorisation with exceptions tasks. For the remaining tasks, differences in filtered training sets account for moderate to high behavioural variations particularly towards the end of lineage. Figure 6.10 (c) shows that differences in initial weight values do not lead to significant behavioural variances in any task except the source task, wherein the effects were reliable. Although non-significant, for English past tense acquisition and categorisation with exceptions differences in weight values result in substantial performance variations throughout the lineage.

Finally, decreasing heritability gradients for all tasks indicate that the range of variation of intrinsic parameters being targeted by selection is acting in a domain relevant capacity. Figure 6.11 and Figure 6.12 depict the changes in the mean values of the parameters and the change in the range of variation over generations. There is a sharp increase in the mean value of number of hidden units and a significant decrease in the mean value of slope of logistic activation function as well. Mean for initial learning rate also experiences a drop over generations.

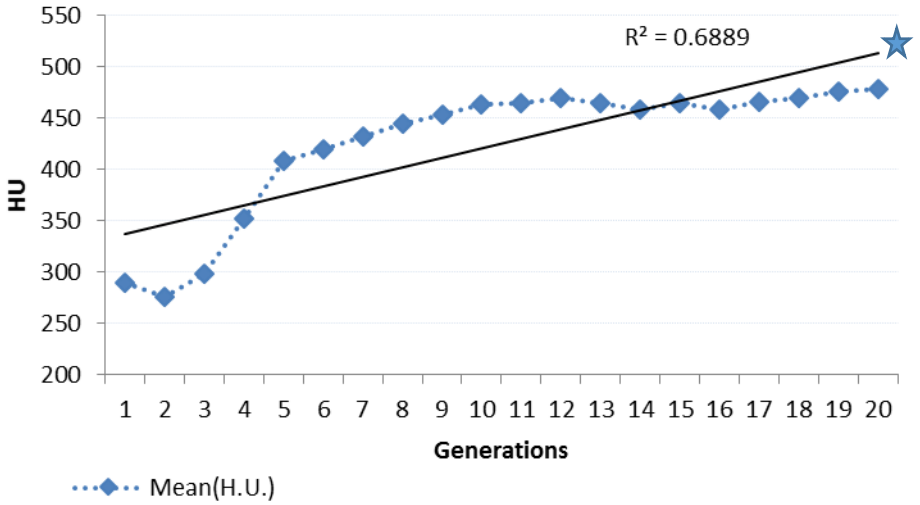


Figure 6.11 (a): Change in the mean value of the number of hidden units per generation

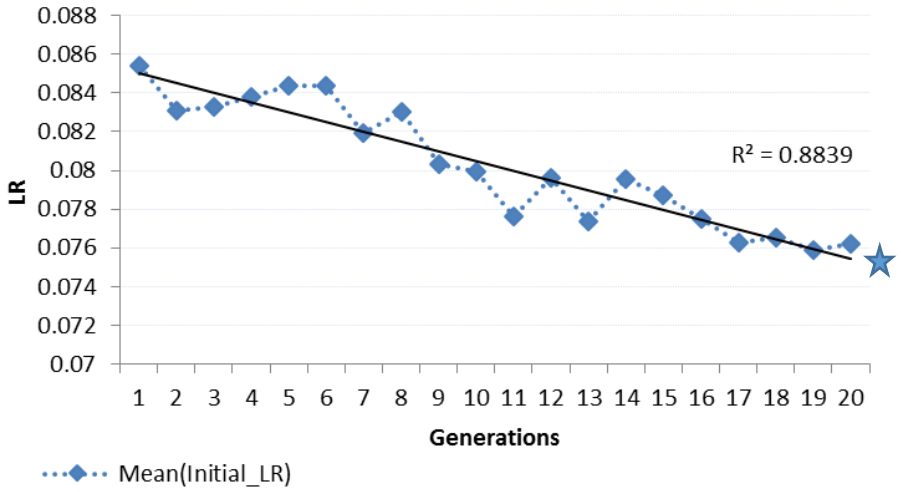


Figure 6.11 (b): Change in the mean value of the initial learning rate per generation

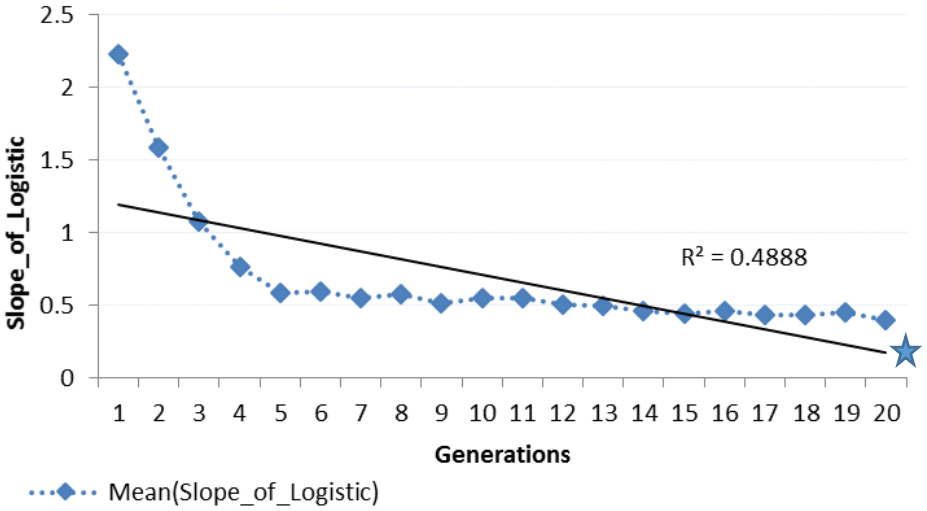


Figure 6.11 (c): Change in the mean value of the slope of logistic activation per generation

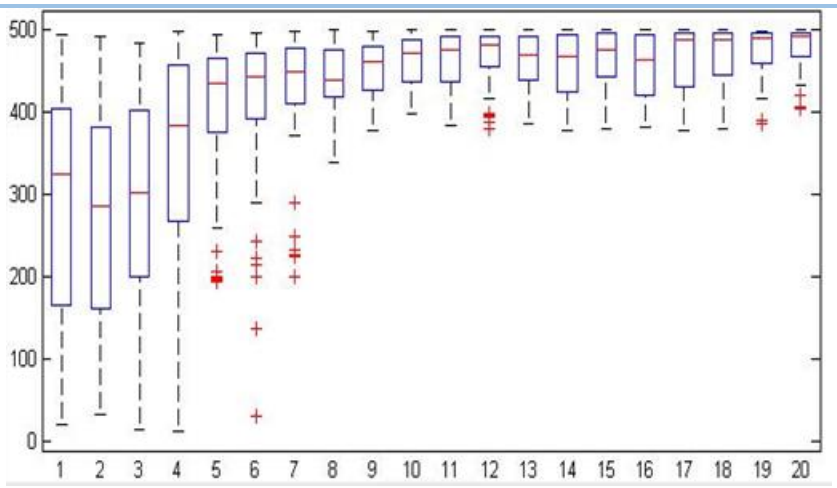


Figure 6.12 (a): Variations in the range of number of hidden units over generations

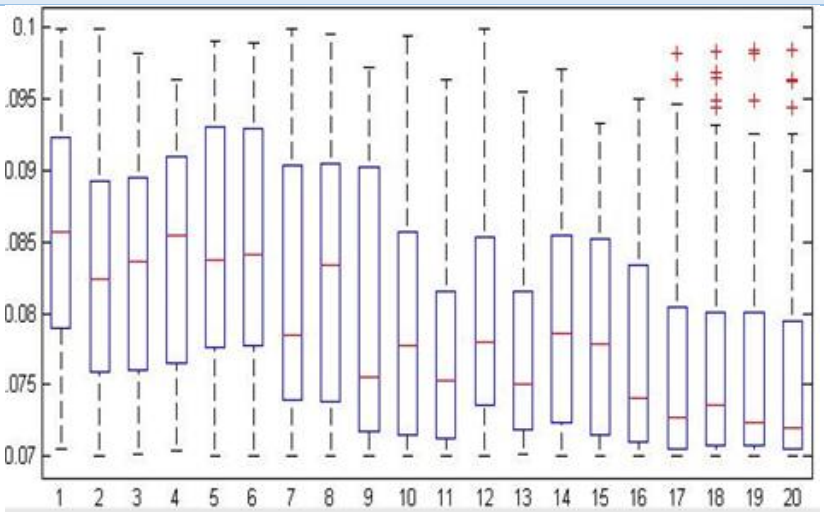


Figure 6.12 (b): Variations in the range of the initial learning rate over generations

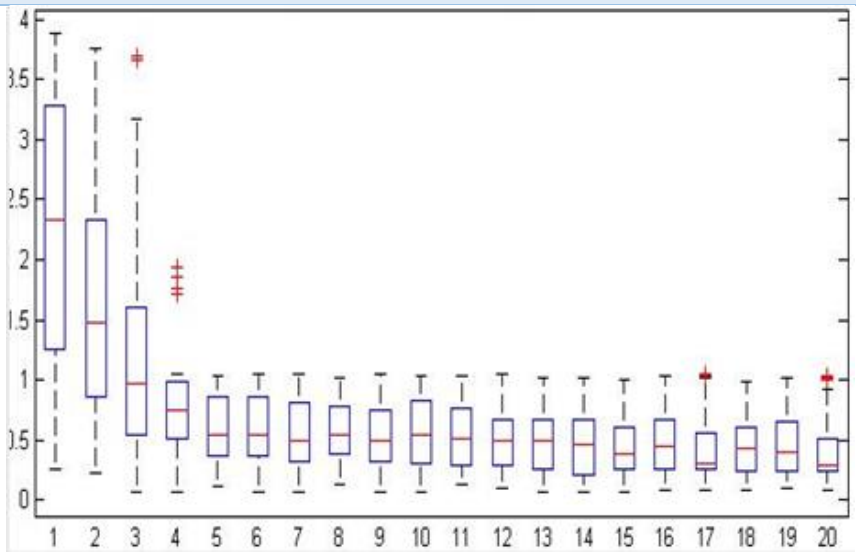


Figure 6.12 (c): Variations in the range of the slope of logistic activation over generations

The range of variation of number of hidden units and slope of logistic activation function becomes considerably small over generations and settles at the higher end of spectrum for the former and at lower end of spectrum for the latter. In this replication networks have ample capacity in terms of large number of hidden units and good ability to learn by virtue of neither too steep nor too shallow slope of logistic activation function and suitable learning rate. This combination of ample capacity and good ability proves beneficial for all kinds of tasks and thus makes transfer a success.

6.5.1 Evaluating benefits of transfer

To analyse the benefits of transfer, the following two questions were addressed.

Q1. Did the framework enabled the ANN twin' populations to learn multiple heterogeneous tasks?

Yes – the ANN populations achieved high accuracy levels on all tasks and had improving accuracy gradients. Therefore our method enabled the populations to acquire multiple heterogeneous tasks.

Q2. Was the proposed method able to avoid negative transfer by assessing task relatedness and having a domain relevant range of variation for neurocomputational parameters?

Yes – negative transfer did not occur in this instantiation of transfer framework. The range of variation targeted by selection i.e. increased capacity provided by greater number of hidden units and good ability provided by neither too steep nor to shallow slope of logistic activation function acted in domain relevant capacity and proved beneficial for acquisition of all tasks.

6.6 R_{10}, Source task: categorisation

Finally in the last replication consistent categorisation was chosen as the source task. This task involves grouping things based on prototypes. The population of artificial neural networks had to learn to assign input patterns to different categories based on their similarity to the prototype pattern for each category. The results from previous replications have repeatedly shown that ANN populations achieve very high accuracy levels on this task with mostly no variation and irrespective of how the genome is changing. Optimising populations on categorisation task by applying selection based on fitness and then transferring the ability to learn seems interesting

since – (a) all networks usually are 100% accurate on this task even when they weren't being optimised on it and thus there is not really room for any further optimisation and (b) from previous results we know that networks don't seem to depend on a particular range of variation of their neurocomputational parameters in order to perform well in this task. To analyse the aforementioned points we tested the framework for 20 generations with categorisation as the source task.

Figure 6.13 contains the mean performance accuracy and generalisation accuracy achieved by ANN populations through the lineage. The graphs in Figure 6.13 depict that accuracy for categorisation, categorisation with exceptions and English past tense acquisition are maintained at nearly fixed levels throughout the lineage, although these trends were not statistically reliable. However, performance accuracy on auto association and arbitrary association experiences a significant decrease over generations. Similar trends were observed for generalisation performance whereby accuracy levels for the former three tasks were maintained fixed albeit at lower accuracy levels and only the gradient for auto association task experienced a significant drop.

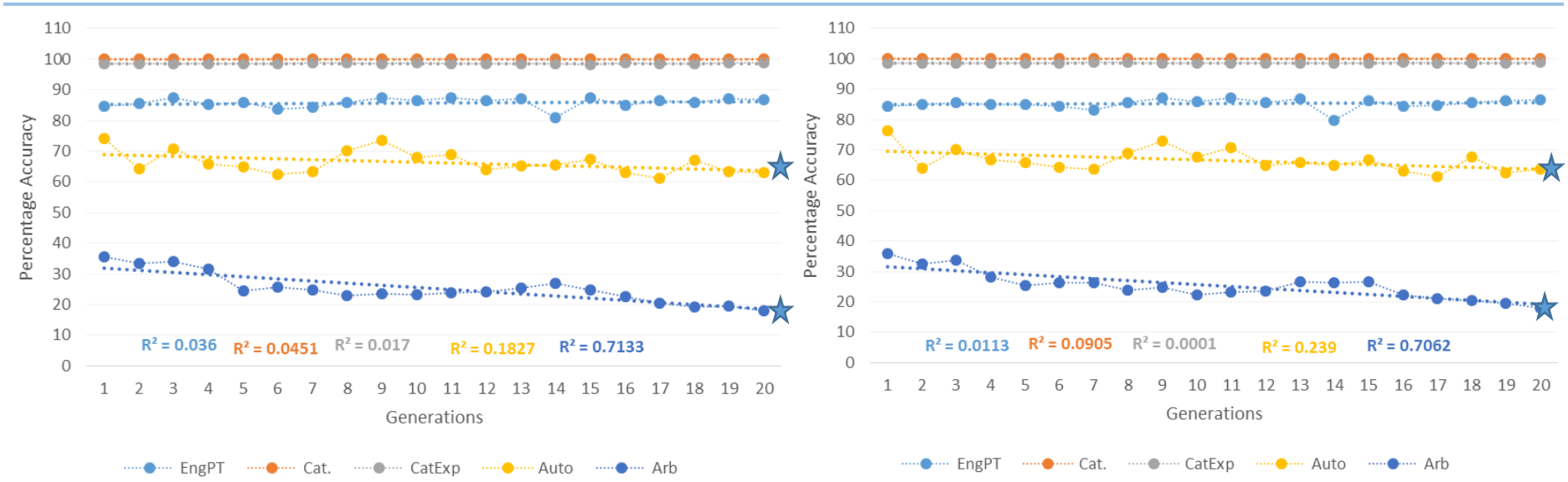


Figure 6.13 (a): Mean performance per generation for breeding (left) and non-breeding (right) twin populations with categorisation as source task

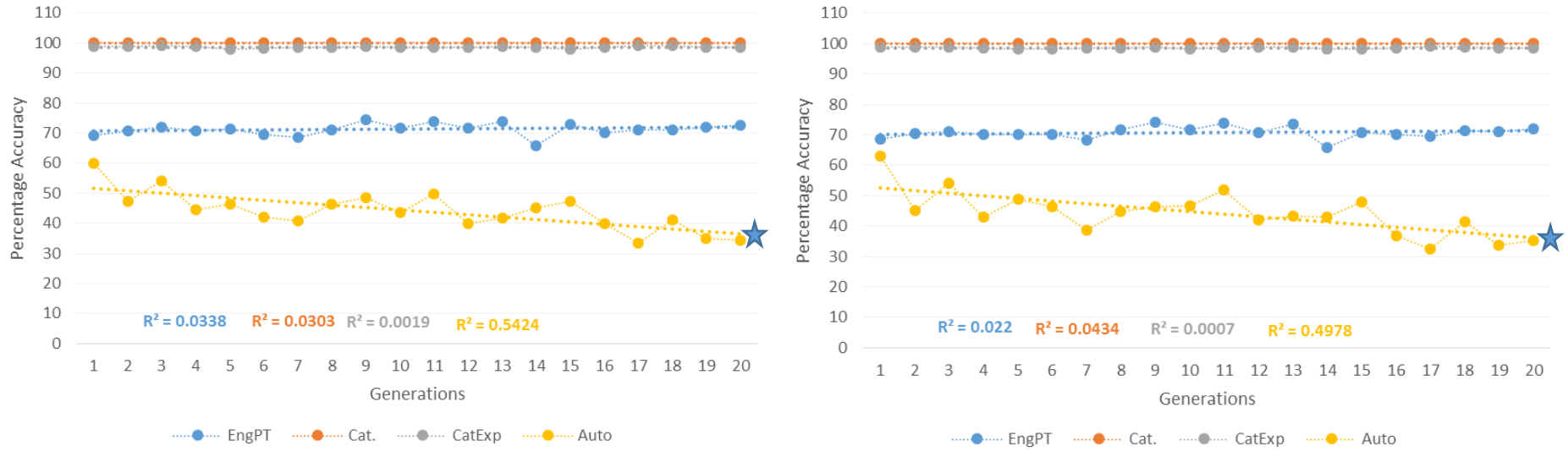


Figure 6.13 (b): Mean generalisation accuracy per generation for breeding (left) and non-breeding (right) twin populations with categorisation as source task

The estimates of heritability and environmental factors provide more insight into the causes for observed performance trends. Figure 6.14 shows the proportion of variance accounted for by genetic and environmental factors respectively.

The heritability and environmentability for the source task is non-computable since there is no variation in performance accuracy and therefore R^2 value is shown as N/A in graphs below. Heritability for categorisation with exceptions task is also mostly varying close to zero with a slightly increasing gradient. Arbitrary association and English past tense acquisition have an increasing heritability gradient positioned at high values (above 1.0) for the former and at moderate-to-high values (+0.5, +1.0) for the latter. However, none of the aforementioned gradients were significant. Auto association task on the other hand, has a counter-intuitive heritability gradient, exhibiting significantly decreasing trend despite a worsening performance accuracy over generations.

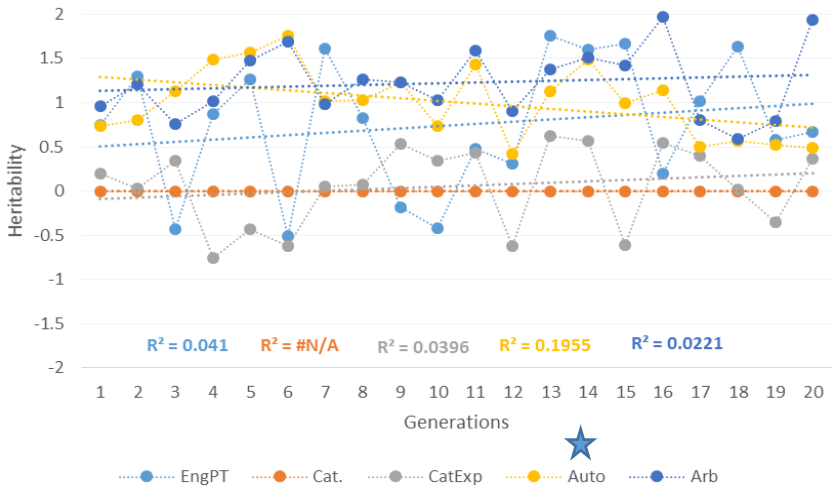


Figure 6.14 (a): Proportion of variance due to genetic factors i.e. heritability – categorisation source task

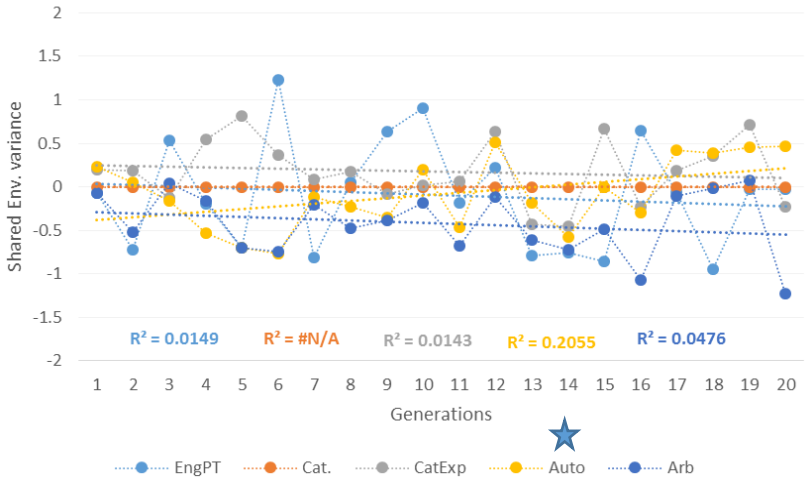


Figure 6.14 (b): Proportion of variance due to shared environmental factors – categorisation source task

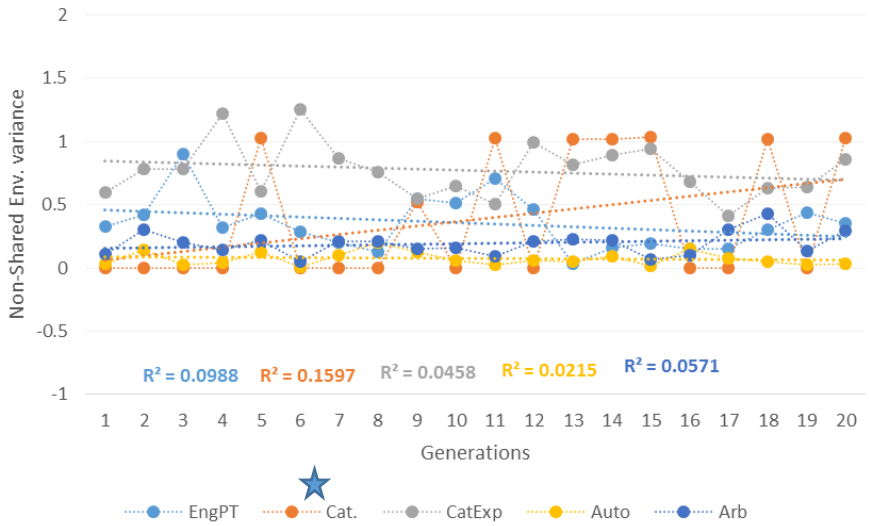


Figure 6.14 (c): Proportion of variance due to non-shared environmental factors – categorisation source task.

The gradients of behavioural variation due to differences in training sets, as shown in Figure 6.14 (b) are fairly invariable at low values for most tasks. Therefore we can infer that differences in filtered training sets do not result in any significant differences in accuracy levels achieved by ANN population members, with exception to auto-association task. On the other hand, differences in initial weigh values of ANNs i.e. non-shared environmental factors have significantly increasing gradient for the source task inferring once again that ANNs depend mostly on their weight values to acquire this particular task. For the remaining tasks the effects of non-shared environmental effects were non-significant.

Finally the mostly nil heritability for source task raises some questions about intrinsic parameters being targeted by selection based on fitness and its impact on the range of variation of parameter values. Figure 6.15 depicts the changes in the mean values of intrinsic parameters over generations and Figure 6.16 represents the changes in the range of variation itself.

Figure 6.15 depicts a drop in the mean value of hidden units over generations and an increase in the mean of slope of logistic activation function. The mean for initial learning rate remains fairly invariable throughout the lineage.

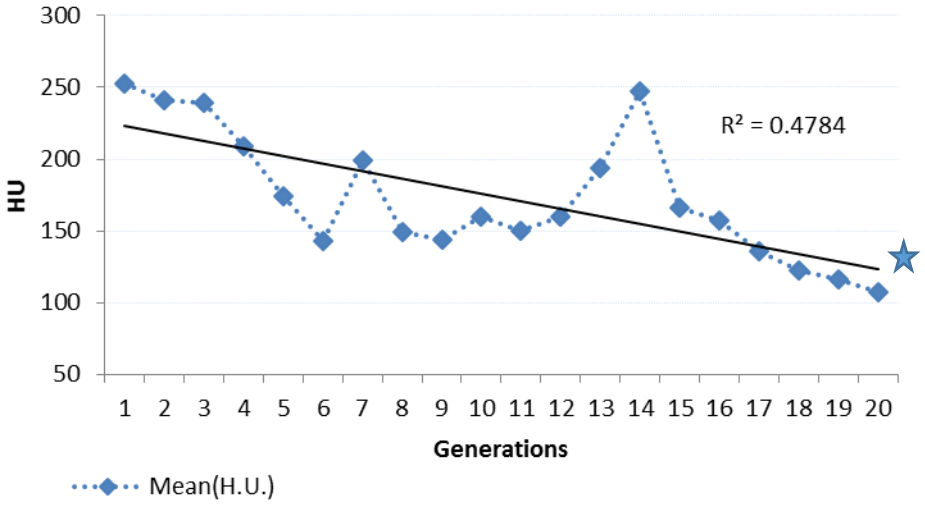


Figure 6.15 (a): Change in the mean value of the number of hidden units per generation

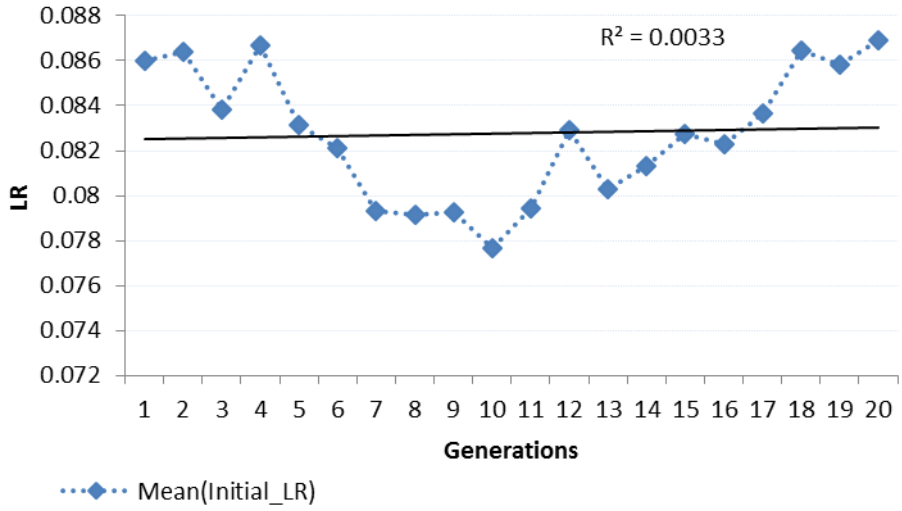


Figure 6.15 (b): Change in the mean value of the initial learning rate per generation

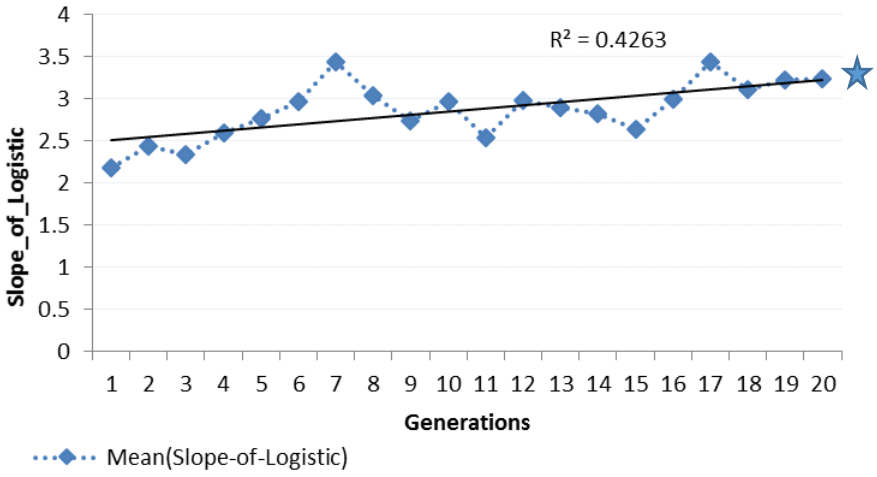


Figure 6.15 (c): Change in the mean value of the slope of logistic activation per generation

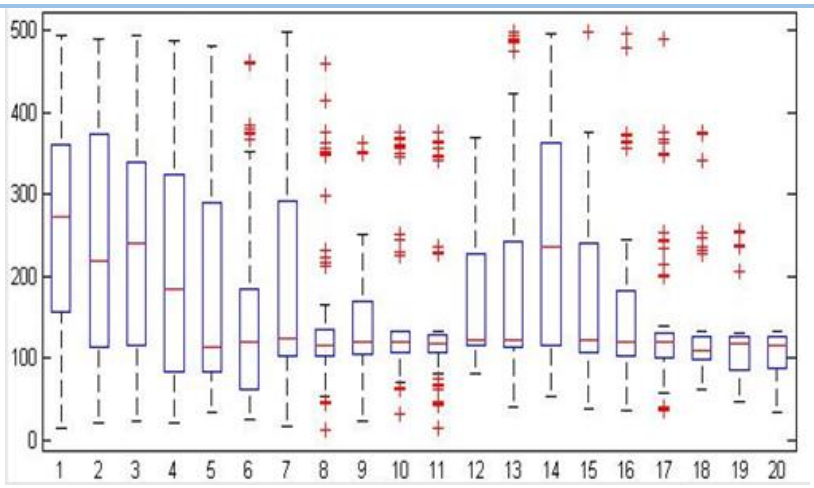


Figure 6.16 (a): Variations in the range of number of hidden units over generations

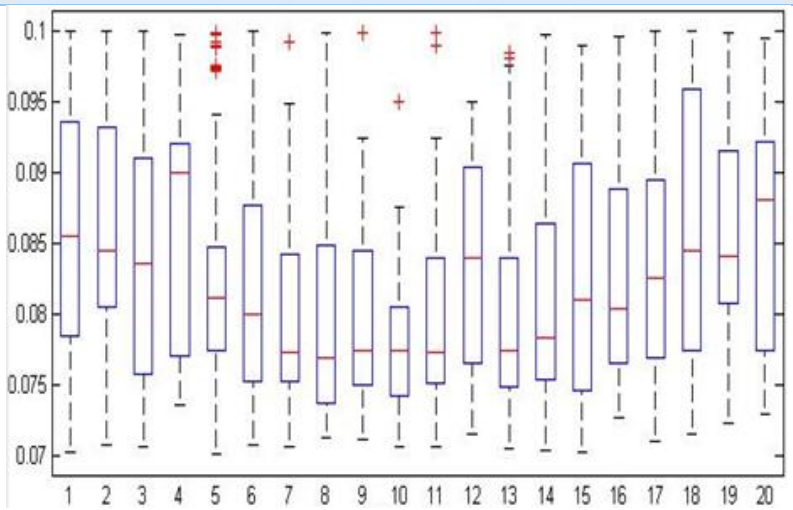


Figure 6.16 (b): Variations in the range of the initial learning rate over generations

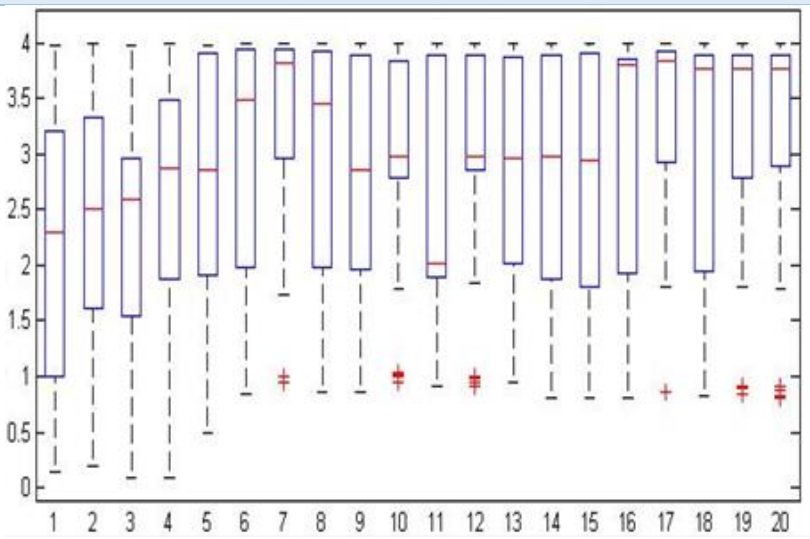


Figure 6.16 (c): Variations in the range of the slope of logistic activation over generations

Although trends emerge in the gradients over generations, yet focusing on Figure 6.16 we can see that the actual range of variation although gets skewed over generations but remains fairly spread out for all three intrinsic parameters. This is an important observation because: from past results it is evident that intrinsic parameters tend to act in domain relevant capacity when the number of hidden units increases with generations and the slope of logistic function decreases through the lineage. Both of the said parameters are trending in opposite directions in this lineage and yet ANN populations are able to maintain their accuracy levels in three out of five tasks. Additionally, the direction of change in the range of variation of neurocomputational parameters is opposite to the range of variation obtained in the last replication which had auto association as the source task and yet the heritability of auto association task decreases despite worsening performance gradient.

Both of the aforementioned observations might have an explanation in the fact that although the range of variation shifts in a direction opposite from the domain relevant range yet it hasn't become much smaller i.e. the parameter values are still fairly spread out throughout the replication. This implies that over generations there are ANNs which have neurocomputational parameters set to the right values needed to at least maintain performance on some tasks. Clearly they are unable to do so for auto and arbitrary association. The decrease in heritability gradient of auto association task could be due to this quite wide range of variation i.e. as evident from Figure 6.16, even in later generations there are networks which have good capacity/ability and these networks depend on their neurocomputational parameters to try and acquire the task. Although the number of such networks with good/suitable genes might be less and that is why mean population accuracy decreases. However, because such networks with good capacity/ability exist in the population, so differences in intrinsic parameter values result in differences in accuracy levels attained and thus heritability decreases over generations despite decreasing accuracy levels. Arbitrary association task does not replicate this heritability trend probably since most networks do not possess the intrinsic properties needed for the task and thus do not depend on genetic properties for learning.

6.6.1 Evaluating benefits of transfer

To analyse the benefits of transfer, the following two questions were addressed.

Q1. Did the framework enabled the ANN twin' populations to learn multiple heterogeneous tasks?

Yes and No – the ANN populations achieved high accuracy levels on categorisation, categorisation with exceptions and English past tense acquisition. The accuracy gradients for these tasks remained fairly invariable through the replication. However the performance on auto and arbitrary association tasks worsened over generations. Thus categorisation as the source task did not result in optimisation of any task, the performance was either maintained at a certain level or it worsened.

Q2. Was the proposed method able to avoid negative transfer by assessing task relatedness and having a domain relevant range of variation for neurocomputational parameters?

No – negative transfer occurred for at least two target tasks in the current replication. The range of variation targeted by selection i.e. decrease in capacity and lack of learning ability due to steep slope of logistic made the range of variation unsuited for certain tasks. The remaining three tasks maintained their high accuracy levels through the lineage but there was no real improvement in performance. This might be because the ANNs perform extremely well in the source task from the very beginning and hence there is no real room for improvement. This might explain why selection targets counter-intuitive intrinsic values.

6.7 Discussion

The results obtained by experimenting with over 80,000 neural networks spanning four replications, each with a different source task have uncovered some stimulating outcomes. The nature of source task evidently becomes an important modulator of transferability only when ANNs find the source-task challenging i.e. requiring some learning effort. The results (especially of R_7 and R_9) revealed that despite the heterogeneous nature of source tasks, the premise of 'generalist-genes' holds valid. In R_7, the source task was arbitrary-association and in R_9, the source task was auto-association. Both of these tasks vary considerably in terms of their input-output mappings and level of difficulty, yet both of these tasks targeted neuro-computational parameters varying between similar ranges of variation. These replications were characterised by increasing number of hidden-units and decrease in range of initial learning-rate and the slope-of-logistic activation function. This combination provided ANN populations with both, the capacity as well as plasticity/ability to learn different mappings. Consequently, the transfer was success and negative transfer was avoided in both these replications.

However, the source-tasks in replications 8 and 10, categorisation and categorisation with exceptions, exhibited ceiling effects and ANNs did not rely on their neurocomputational properties for learning. There was no actual optimisation even in source tasks in these lineages (again due to ceiling effects) and thus these replications were marked by some instances of negative transfer. Nonetheless, if some constraints were to be included in the source tasks like increased size of training data set or reducing the number of training epochs, these ceiling effects could be contained and transfer might have been positive and beneficial.

6.8 Summary and contribution of chapter

This chapter presented the experimental evaluation of a behavioural genetics inspired transfer approach capable of performing heterogeneous transfer. The focus was on examining the effect of source task on transferability. Over 80,000 ANNs were trained as part of experimental evaluation process spanning 4 replications, comprising four different source tasks. These tasks differed in terms of degree of similarity between input-output mappings and the presence of structure and regularity in mapping and thus posed different computational requirements. The results reported in this chapter validate the premise of 'generalist-genes' based transfer framework. Each replication was marked by a different source task and yet the evolution targeted (especially in R_7 and R_9) neuro-computational parameters varying within similar range of variation, implying that same set of genes/intrinsic parameters are responsible for diverse learning and cognitive abilities. The results also showed that the combination of genes + environment provides ANNs with the ability to learn any behaviour/task and the method transferred the '*ability to learn*' from source to multiple target tasks. In addition to performing heterogeneous transfer, the range of variation of heritability values acted as an indicator of task relatedness in these replications as well.

The results were encouraging and have showed the effectiveness of this transfer method under varied conditions. However, a direction for extension of this work is to transfer from multiple source tasks and more importantly apply this method to more realistic (i.e. real world) applications.

Chapter 7 Conclusions and Future Work

7.1 Overview

This Chapter provides a summary of the thesis and a discussion of its contributions (Section 7.2). It also analyses its limitations and offers insights for future work (Section 7.3).

7.2 Summary and contribution of thesis

The work presented in this thesis offers a new perspective to the evolution of ANNs that exhibit intelligent behaviours. Evolutionary neural networks is a widely researched field with numerous successful methods and applications. A comprehensive literature survey presented in Chapters 1 and 2 covering the research efforts made in the said field revealed that despite being hugely successful, there are still some open questions in this area, such as the need for a more generic approach that would not be constrained by task specifics and would be able to work for more than one task, or a method that could combine evolution and learning without resulting in catastrophic interference/forgetting to name a few. Based on an analysis of previous research efforts, in this thesis a novel neuroevolutionary approach based on principles of behavioural genetics was presented. The approach evolves the ANN's general 'ability to learn' and combines evolution and learning within a single framework. The research presented in this thesis can be grouped into three main parts, which are summarised below:

BG inspired neuroevolutionary framework: Chapter 2 of this thesis presented a BG-inspired neuroevolutionary approach for evolving a population of ANNs. The approach combined evolution and ontogenetic adaptation (i.e. learning) within a single framework. It is generic, scalable and evolves a population of ANNs that can acquire any number of learning tasks. The approach parallels between the intrinsic properties of ANNs and genes and between training datasets and connection weights of ANNs and the environment; wherein the genes modulate and constrain learning and the environment provides learning bias. The interaction of the two provides ANNs with a general 'ability to learn' any specific skill. The approach evolves the ANN's 'ability to learn'. To model the ability to learn such that it is evolvable, the formational properties of ANNs (i.e. genes) were encoded into a genome within a fixed range of variation. Additionally a filter was applied to the training datasets that determined the quality of each networks training (shared) environment. The connection weights of ANNs were considered to

be representatives of unique environments which varied with the environment (i.e. task) within which the ANN system was placed. A local gradient-based search method was incorporated in this evolutionary approach in order to help learning. The ability to learn was evolved through a Darwinian approach by a fitness-based selection criterion (based on mean performance accuracy). Thus, the approach presented in this thesis combined evolution and learning in a systematic way which is not masked by problem specifics and can be applied to any given set of tasks. The key insight to this approach is that it makes it possible to gauge the net effect of a neurocomputational parameter set if (i) the parameters are allowed to vary in a population and (ii) the quality of training set is allowed to vary as well.

Applying BG inspired framework to model English past tense acquisition: In the second phase of work done (presented in Chapter 3), the neuroevolutionary approach was adapted to model children's acquisition of English past tense verbs and to capture individual differences. This work captures the interaction of evolution and learning when placed within a single yet dual natured task. The model used a population of ANN twins to disentangle effects of genetic and environmental influences on behavioural differences. The past tense acquisition model was tested using two different selection operators – stochastic (RW) and deterministic (truncation). The application of selection on developmental performance of ANN twins on a quasi-regular task differentiates this work from others reported in literature. The experimental evaluation of this model focused on individual differences in performance and showed that the effect of applying selection on an individual's performance leads to divergent behaviours subject to initial conditions. The results also highlighted that once selection starts steering a specific aspect of quasi-regular task, it behaves like Waddington's epigenetic landscape as in, that trend continues throughout the lineage. This phenomenon is sometimes also referred to as 'restriction of fate' (Nishida, 1997). The findings corroborate the usefulness of the method within an evolutionary setting and provide the basis for future work to capture population-level differences within a developmental setting.

Extending the framework to model transfer learning: Lastly, the BG-inspired approach was extended to transfer learning with a special focus on heterogeneous transfer scenarios (work presented in Chapter 4, 5 and 6). The transfer model used ANNs as computational models capable of learning heterogeneous tasks in an evolutionary setting. From a neuroevolution perspective this represented a scenario wherein the population members are capable of learning tasks different from those they have been selected for. The formational parameters of ANNs

were considered as representatives of genes, the training datasets corresponded to the shared environments whilst the unique connection weights of ANNs captured non-shared environments. Therefore, the approach embodies the effects of both genetic and environmental influences thereby imitating learning as it occurs in humans more closely. The neurocomputational properties (i.e. genes) shape and constrain learning whereas the training dataset and connection weights of ANNs provide the learning bias. The interaction of these two entities provides ANNs with their general 'ability to learn' or 'learning predisposition'. Thus, by having the same quality of training datasets and same neurocomputational parameters but different connection weights, the approach transferred the general 'ability to learn' across numerous heterogeneous tasks. The approach also identified the two key factors that modulate the performance of model – type of selection operator and nature of source task. The approach was tested on different combinations of genetic and environmental influences. Overall the transfer model was tested on 10 replications, each with a 20 generation duration and included ANN populations with over 200,000 members. Through the transfer of 'ability to learn', the model enabled population of ANNs to acquire five different heterogeneous tasks successfully, thereby demonstrating that it is possible to store and reuse acquired knowledge without catastrophic forgetting/interference. Though, it is worth mentioning that ANNs did not learn five tasks at once or one after the other, rather the tasks are acquired independently. The experiments also revealed the role of the type of selection operator used in modulating overall performance of model, wherein having a deterministic selection operator results in a more accurate model but at the same time it can also result in loss of some classes. Stochastic selection on the other hand results in less accurate performance but covers all the classes even in case of class imbalance. The experiments also demonstrated that learning and evolution interact and guide each other, whereby fitness-based selection determines what genes, i.e. learning ability next generation members will have. Fitness, in turn is derived from mean performance accuracy (which results from learning/training), governs what members get chosen for breeding next generation. In addition, the results revealed the role of heritability as an identifier of task relatedness and thereby avoiding negative transfer or catastrophic interference. Finally, the analysis of heritability and environmentability revealed the factors causing most behavioural (performance) variance for each task and therefore in cases where negative transfer occurs, training could be biased towards the most variance-causing factors to boost accuracy. Although this idea was not implemented, it will be incorporated in future extension of this work.

The key findings of this project are:

- The proposed neuroevolutionary approach is systematic, generic and adaptable. It does not depend on problem specifics and thus can be applied for any given set of task(s), i.e. enables incremental learning. Also it imitates learning as it happens in humans more closely.
- It combines evolution and ontogenetic adaptation (i.e. learning) within a single framework. In other words, the method is capable of storing and reusing the acquired knowledge whilst learning new tasks.
- It helps in understanding and synthesising the evolutionary pressures (genetic or environmental) leading to high-level intelligence.
- It scales the neuroevolutionary approach to evolve cognitive behaviours such as language acquisition and enables lifetime learning as well.

As with any research, this work also has limitations and scope for improvements that are discussed in the next section which covers directions for future work.

7.3 Directions for future research

This section presents the various directions for future research, grouped by five main categories, each of which is discussed below. These are: (i) ANN properties; (ii) ANN operators and learnability; (iii) Interpreting environment and its effects; (iv) Extending framework for other applications and (v) Enhancing robustness of neuroevolutionary framework.

Group 1: ANN properties

More complex ANN architectures **-** Chapter 2 presented the neuroevolutionary framework wherein the connectionist model employed a 3-layer feedforward artificial neural network. However, it is known that the type of ANN used can significantly influence its learning abilities (Risi and Togelius, 2015) and ergo the size, architecture, and complexity as dimensions of ANNs could be varied to test model's robustness and performance. Through the last decade or so, the field of neuroevolution has witnessed growing use of numerous advanced neural networks such as: feedforward deep neural networks (Hinton et al., 2012), deep learning recurrent neural network, better known as the Long short term memory (LTSM) network (Li

and Wu, 2015), continuous time recurrent neural network (CTRNN) (Zhang et al., 2014), neural Turing machines (NTMs) (Greve et al., 2016), modular networks (Schrum and Miikkulainen, 2014), spiking neural networks (Pavlidis et al., 2005) and compositional pattern-producing networks (CPPNs) (Zeng et al., 2016) to name a few. Therefore, the first extension of the work proposed in this thesis would be to use/test the framework with different complex neural network architectures. Crucially, the approach used in this project is robust to increases in the size, complexity and number of parameters in the architectures.

More complex and bigger genome **-** in this work, three neurocomputational parameters were encoded into the genome and optimised to achieve best learning. The learning ability was determined by the cumulative effect of the following parameters, the number of hidden units, initial learning rate and the slope of logistic activation. However, using more complex network architecture would imply having greater number of neurocomputational properties targeting different aspects of networks like network construction, processing dynamics, network maintenance, adaptation and response. Therefore, employing a more complex network architecture would in turn entail having a bigger and more complex genome, capable of encoding many diverse network properties. An additional advantage of encoding more properties in the genome is that since the neurocomputational parameters help in explaining the variance in the population, ergo having more properties encoded into genome increases the chances of finding domain-relevant parameters (Thomas, 2016).

Different range of variation of encoded parameters and filtered training sets **-** Using a bigger genome that encodes lots of different intrinsic properties of ANNs will entail use of different ranges of variation depending on intrinsic properties being encoded. In Chapters 5 and 6, the neuroevolutionary framework was extended to model transfer learning, however, the range of variation for encoded parameters was kept the same. In future, the range(s) could be modified with respect to different source (or evolutionary) tasks and could also be modified with respect to level of desired variability in population i.e. wide range (which incorporates the full range of variation) or the narrow range (which includes restrictions on the range of possible values). This will in turn constrain learnability and consequently affect behaviour/performance (Thomas et al., 2009). Likewise, another option is to experiment with different range of variations for environmental influences, by having different filtering ranges that could be wider or narrower (Thomas et al., 2009). This would facilitate investigating the effects of having different combinations of genetic and environmental variation on behavioural outcome of this

framework. In human behaviour variations in environmental influences over generations have been proposed to lead to bifurcation of sub-populations with different degree of sensitivity to the environment during development (i.e. plasticity) such as in Belsky et al.' differential susceptibility hypothesis (Belsky et al., 2007).

Bigger population size **-** the choice of population size has been the subject of various studies (Jansen et al., 2005). In this work, a consistent population size of 100 ANN twins per generation was used. However, the use of more complex genome should be accompanied with bigger population size. Having a bigger population size would ensure more variability in population and adequate exploration of fitness space and consequently will enhance the chances of finding more networks that can accurately perform phylogenetic evolution and ontogenetic adaptation.

Experimenting with different weight initialisation techniques **-** effective weight initialisation is associated with performance characteristics such as the time needed to successfully train the network and the generalisation ability of the trained network (Adam et al., 2014). The approach presented in this thesis used a weight initialisation method proposed by (Bottou, 1988). Although this method served well in this framework, nevertheless there are various other weight initialisation techniques reported in literature (refer Adam et al., 2014; Murru and Rossini, 2016; Qiao et al., 2016; Yam and Chow, 2000). Thus, in future one avenue would be to try different weight initialisation techniques and assess their impact on performance of this model when applied to different scenarios such as language acquisition or transfer learning.

Group 2: ANN operators and learnability

Type of selection operators – the experiments presented in Chapter(s) 3 and 5, demonstrated that different selection operators led to very different behavioural performances. If deterministic selection resulted in higher cumulative accuracy, then stochastic selection was able to take into account sparsely represented classes (i.e. handled class-imbalance). Thus each selection type has its own merits and demerits. There are plenty more selection mechanisms proposed in literature (Goldberg and Deb, 1991; Sastry et al., 2014) and it would be informative to test the framework's performance whilst using other selection techniques, particularly with respect to the canalisation of parameter sets over generations.

Reward-based early selection – In this work, selection was driven by fitness, defined by mean performance and was applied at the end of training. This implies that fitness was mainly modulated by capacity and not necessarily learnability of the networks. Hence, both slow and fast learners got equal opportunities for getting chosen. However, selecting fast learners might be advantageous as it would not only lessen the training time but might also improve the overall fitness of the population. Therefore, the next focus would be to apply some reward-based fitness strategy (Maniezzo, 1994) which differentiates between slow and fast learners, for instance, selecting networks that can reach some pre-set accuracy level within pre-set number of epochs. This would allow early selection of networks that have better learnability.

Co-operative/interactive learning – in this work, the approach for learning and adaptation draws inspiration from human cognition and incorporates main elements affecting it, yet there is one crucial difference. Learning, in this approach occurs in isolation. There is no interaction amongst the networks, whereas learning in humans includes interactions between individuals within a generation and between generations, and is affected by social and cultural influences. Therefore, another extension of this work would be to include the effects of interactions and group/social dynamics in learning and adaptation. Notable research advanced nature-inspired methods incorporating social and cultural dynamics into machine learning with promising results. Methods include swarm intelligence/optimisation (Xu et al., 2015), collaborative learning (Chandra, 2015), and cultural dynamics swarms (Ali et al., 2016; Reynold and Peng, 2004) amongst others.

Group 3: Interpreting the environment and its effects

Epigenetics as representative of shared environmental influence – connectionist modelling has demonstrated that the structure of the learning environment interacts with the system's internal constraints in terms of the neurocomputational properties of ANNs to shape developmental trajectories in behaviour (Thomas et al., 2009). The neuroevolutionary framework presented in this thesis construed socio-economic-status, or SES, as a shared environmental influence. SES is a well-known environmental measure that predicts significant individual differences in cognitive and language development domains (Thomas et al., 2013). However, shared environmental influences refer to all non-genetic influences that make family members (or in this case – twins) similar to one another (Plomin et al., 2013) and thus there are other ways for representing shared environmental influences in simulations presented in this

thesis as well. One of the viable options to pursue is epigenetics. It is a field of research focused on study of heritable changes in gene expression that does not involve changes in underlying DNA sequences, i.e. a change in phenotype without a corresponding change in its genotype (Sadikovic et al., 2008). Epigenetic effects can be heritable, which implies that a parent's experiences, in the form of epigenetic tags, are passed down to future generations (Epigenetic tags are chemical additions made either to the DNA or its affiliated proteins, the histones. These tags mediate the expression of the genes upon which they have been placed, either by activating or inhibiting them). This epigenetic inheritance has added an extra dimension to how evolution is viewed. Normally changes in genome occur slowly, through process of random mutation and natural selection, requiring many generations for a genetic trait to become common in the population. By contrast, the epigenetic effects change rapidly in response to some stimulus from environment. Additionally, what is sometimes termed the epigenome maintains plasticity as the environment continues to change implying that it allows an organism to continually adjust its gene expression with respect to a changing environment, without altering its DNA sequence (Chong and Whitelaw, 2004).

Research focussing on modelling/simulating epigenetic effects in connectionist networks is still in the fairly early stages. However, some significant research efforts have been made by (Turner et al., 2015; Turner et al., 2016) wherein they proposed an artificial epigenetic network which is essentially a recurrent neural network capable of dynamically modifying its topology so as to decompose automatically, and thus solve dynamical problems. Their research has given an inspired starting point to extend work in this direction.

Simulating gene-environment (G-E) correlations – In this framework, the quality of environment was sampled independently of properties of genome. There was no gene-environment correlation. However, behavioural genetic research has shown that genetic propensities are in fact often correlated with individual differences in experiences (Plomin et al., 2013). In simpler terms, it implies that what appears to be an environmental effect can actually reflect a genetic influence because these experiences are predisposed by genetic differences among individuals. Gene-environment correlations are of three types (as discussed in Chapter 2, Section 2.4.4): *Passive* – this refers to scenario wherein children passively inherit from their parent's family environment that are correlated with their genetic propensities. For example, musically gifted children are more likely to have musically gifted parents who provide them with (musically) good genes and an environment conducive to the development

of musical ability. *Evocative* (or reactive) – this occurs when individuals on the basis of their genetic propensities, evoke reactions from environment (or other people). For instance, musically talented kids are more likely to get picked out at school and given special opportunities to further enhance their musical talent. *Active* – this refers to the association between an individual's genetic propensities and the environmental niches that the individual selects. To illustrate, even if no one does anything specifically to enhance their musical ability, gifted children might actively seek out musical environment by having musically inclined friends or otherwise creating their own musical environments (like joining a band) (Plomin et al., 2013).

Computationally speaking, one probable way of simulating the aforementioned scenarios could be: *Passive* - ANNs in this work inherit genes (i.e. neuro-computational properties) from their parents. Thus if say an ANN performed extremely well in a particular task, it is more likely to pass these domain-relevant genes to its offspring ANN. Similarly, environment conducive to learning and acquisition of this particular task can be generated by say, using parent ANN's trained weight distributions to initialise offspring ANN's unique weights. This way offspring ANNs have both inherited genes (neurocomputational properties) and a suitable environment for acquisition of a specific task. *Evocative* – in this case, a preconditioning phase (consisting of pre-training ANNs on a very small sample of actual dataset) can be used to determine 'gifted' networks for each 'class or sub-task' and then during actual training the learning algorithm identifies these gifted networks for each class/sub-task and allocates them for learning that specific portion of dataset only, thereby further enhancing their expertise. This technique is somewhat similar to Mixture-of Experts (Yuksel et al., 2012) methods wherein networks learn different aspects and then their individual solutions are combined to represent overall solution. *Active* - this can be emulated by pre-training ANNs on a very small sample of training datasets and then using the preconditioning/pre-training results (per class/sub-task) to generate a probability distribution (again per class/sub-task) that in turn will be used to generate training subset for that particular network. Thus if a network performed well in say class 1 compared to class 2, then its probability distribution will mirror this result and its actual training subset will be more likely to have greater number of samples of patterns belonging to class 1.

Group 4: Extending the framework for other applications

Extending the framework for evolving neural network ensembles (NNE) – A neural network ensemble (NNE) is a very effective way to obtain a good prediction performance by combining

the outputs of several independently trained artificial neural networks. This concept which was first proposed by Hansen and Salamon (1990) has since been applied extensively and successfully in numerous applications, especially those involving pattern classification (Fu and Zhang, 2013). Research in the field has shown that an ANN ensemble offers several advantages over a monolithic ANN. First, it can perform more complex tasks than any of its component ANNs. Second, it can make the whole system easier to understand and modify. Finally, it is more robust than a monolithic ANN, and shows graceful performance degradation in situations where only a subset of ANNs in the ensemble performs correctly. There have been many studies which suggest that ensembles, if designed appropriately, generalise better than any single individuals in the ensemble do (Anastasiadis and Magoulas, 2006; Yao and Islam, 2008).

The key to successful ensemble methods is to include individual members that perform better than random guessing and produce uncorrelated outputs. This also implies that individual ANNs in the ensemble must be accurate as well as diverse. The creation of an ensemble is often divided into two steps: first, generate individual ensemble members; and second appropriately combine individual members' outputs to produce the output of the ensemble (Anastasiadis and Magoulas, 2006). Over the years numerous techniques for creating NNE have been proposed, however there are few methods which have been widely acknowledged as more accurate and efficient. These methods include diversity based NNE creation, mixture-of-experts, negative correlation and more recently distillation, each of which is discussed briefly below.

Diversity – this refers to a class of methods for creating ensembles that focus on creating classifiers that disagree on their decisions. In the context of neural networks, these methods comprise techniques for training with different network topologies, different initial weights, different learning parameters and/or learning different portions of the training set (Anastasiadis and Magoulas, 2006).

Mixture-of-Experts – this is a learning procedure for systems composed of many separate ANNs, each of which learns to specialise in a subset of complete training cases (Jacobs et al., 1991). This method is usually implemented with set of experts (ANNs) and a gating network which cooperate with each other to solve learning problems. The gating network is responsible for learning the appropriate weighted combination of specialised experts for any given input (Yuksel and Wilson, 2012).

Negative correlation learning - this method attempts to train individual networks in an ensemble and combines them in the same learning process. All individual member networks are trained simultaneously and interactively through correlation penalty terms in their error functions. This method creates negatively correlated networks to encourage specialisation and cooperation among the individual networks (Liu and Yao, 1999). These methods have also been lately used in an evolutionary perspective, for instance (Fu and Zhang, 2013) proposed an evolving NNE classifier based on regularised negative correlation learning algorithm.

Distillation – this technique mainly supports deep neural network ensembles. This method involves approximating a deep neural network through a smaller neural network by training it (i.e. smaller network) to reproduce the output of the bigger network without the loss of generality (Hinton et al., 2015; Mosca and Magoulas, 2016). Distilled models are more portable than the original ensemble, because these have a smaller footprint, in terms of computational and memory requirements. Research has shown that this method of distillation can also be interpreted as a regularisation technique, and that the distilled model is capable of improving the generalisation of the ensemble (Mosca and Magoulas, 2016).

Although the aforementioned methods are the more widely used for ensemble creation, numerous other methods have also been proposed such as cooperative coevolution approach for designing neural network ensembles (García-Pedrajas et al., 2005), combining NNE and multi-population swarm intelligence to construct improved neural network ensemble (Zhao et al., 2015) and cross-entropy error function based ensemble (Kusumoputro, 2016) to name a few.

Testing transfer learning model on real world datasets – in the current project the transfer learning model was applied/tested on five tasks from the cognitive domain. However, transfer learning techniques have recently been applied successfully in numerous real world applications and several datasets have been published for transfer learning research (Pan and Yang, 2010). Some of the benchmark transfer learning datasets, as described in (Pan and Yang, 2010) include the following: *text mining dataset*: comprising three datasets, 20 newsgroups, SRAA and Reuters – 21578 have been pre-processed for transfer learning setting. The data in these data sets are categorised into a hierarchical structure and data from different subcategories under the same parent category are considered to be from different but related domains. The task is to predict the labels of the parent category. *Email spam filtering dataset*: data provided

by the 2006 ECML/PKDD discovery challenge. *WiFi localization over time periods data set*: provided by the ICDM-2007 Contest and finally the *Sentiment classification data*: This data set contains product reviews downloaded from Amazon.com from four product types (domains): Kitchen, Books, DVDs, and Electronics. Each domain has several thousand reviews, but the exact number varies by domain. Some researchers have already tested their transfer learning approaches on these datasets successfully (refer Pan and Yang, 2010). Ergo it can be deduced that the transfer learning methods when designed aptly for real-world applications can actually improve the performance significantly compared to the non-transfer learning methods.

Group 5: Enhancing robustness of neuroevolutionary framework

Countering effects of negative transfer or decreasing behavioural gradients – in Chapter 3 Section 3.7.1 and Chapter 5 Section 5.7, it was discussed that once a behavioural or performance trend starts emerging (in a replication), it continues along the lineage of that specific replication. This implies that currently there is no way of reversing this trend, much like Waddington's epigenetic landscape discussed in Chapter 3. This occurrence is not desirable in cases when the performance gradients are declining thereby indicating negative transfer. There are lot of different ways of improving performance accuracy, nonetheless a viable starting would be to begin by implementing an idea discussed in Chapter 5, Section 5.7. It reviewed the option of using the analyses of the proportion of behavioural variance due to genetic and environmental factors to rectify the aforementioned situation mainly because these analyses reveal which of the factors (i.e. genetic or environmental – shared and unique) causes most behavioural variance. It was thus proposed that biasing the training towards the factor leading to maximum accountable behavioural variance could possibly boost performance accuracy. As part of extending this framework, a first step would introduce an implementation to countering the effect of declining gradients, which will ultimately make this framework more robust.

Appendix 1: Datasets Used

Note: The training and generalisation datasets used for each of the five tasks have been embedded here as an excel worksheet. Double-click on any table/paperclip icon below to activate/open that specific dataset.

1. English past tense task

0	1	0	1	0	0	1	1	0
1	0	1	1	1	0	1	0	0
1	0	1	1	1	0	1	0	0
0	1	0	0	1	0	1	1	0
0	1	0	1	1	1	1	0	0
1	1	0	0	1	0	0	0	1
1	1	0	1	0	0	0	0	1
0	1	0	0	0	0	1	1	0
0	1	0	1	1	0	1	1	0
0	1	0	1	0	1	1	0	0
0	1	0	1	1	1	1	0	0
0	1	0	1	0	0	1	0	0
0	1	0	0	1	1	1	0	0
0	1	0	0	1	0	0	0	1

2. Categorisation task

0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0

3. Categorisation with exceptions task

0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
1	1	1	1	1	1	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
1	1	1	1	1	1	0
0	0	0	0	0	0	0
0	0	0	0	0	0	1

4. Auto-association task

1	1	0	0	1	0	0
0	1	1	0	0	0	1
1	1	0	1	0	0	0
0	0	0	1	0	1	1
0	1	1	1	1	0	0
1	1	0	0	1	1	1
1	1	0	0	1	1	0
1	0	0	1	1	0	0
0	0	0	0	1	0	0
0	0	0	1	0	0	1

5. Arbitrary-association task

1	0	1	1	0	0	1
0	0	0	0	0	1	0
0	0	0	0	0	0	1
0	1	0	0	0	1	1
0	1	0	1	0	0	0
1	1	1	1	1	1	0
1	1	1	1	1	0	1
1	1	0	0	1	1	1
1	1	1	1	0	1	0
0	0	1	0	0	0	1

APPENDIX 2

A high-level description of Condor implementation/algorithm followed is explained below taking English past tense as an example:

Procedure Master: This is the part of code that runs on host/submit machine and is sequential in nature.

1. % Generate first population
   ```
   [twinpop1,twinpop2,PhenTw1,PhenTw2] = generate_init_population(count)
   ```
2. % Generate ses and create filtered training sets
   ```
   [ses,fam_data] = createfamdata(inputs,targets)
   ```
3. % initialise individual network folders having specific family data and phenotype
   ```
   initialisetwin1 (or initialisetwin2)
   ```
4. % Create the exe and send ANN training exe to Condor (slave nodes)
   ```
   mcc -m EngPT.m -a./ptCondor
   For networks (slaves) = 1:n do
   ```
   % Submit training file to Condor
   ```
   Condor_submit EngPT.Condor.txt
   ```
   % Collect individual training and generalisation results; combine them (per population) and release slaves
   ```
   % collects results
   [twin1, netindex, indices] = analyseresults(count)
   % computes consolidated results
   genavg = analyseperf(twin1,count)
   end
   ```
5. % Calculate heritability
   ```
   [rmz,rdz,h2,c2,e2] = calc_herit(count)
   ```

% In case of past tense task, also calculate performance accuracy of twinpop1 and twinpop2 according to verb categories. Additionally assess performance on verb categories as twins, i.e. DZs and MZs. This involves running scripts:

```
  Verb_cat_analysis
 % for assessing perf as twinpop1s & 2s
  Twins_verb_cat_analysis
 % for assessing perf as twins, i.e. DZs & MZs
```

6. % Apply selection based on performance
   ```
   Sel_Parents = RW_selection(count)
   ```
7. % Generate subsequent population
   ```
   For generations = 2:n
   [twinpop1,twinpop2,PhenTw1,PhenTw2] =
   generate_new_population(Sel_Parents,twinpop1,count)
   ```
   Repeat Steps 2-6

Procedure Slave: This is the part that runs on Condor and is executed in parallel – all of the steps mentioned below are contained in the tasks respective executable files, for e.g. EngPT.exe

1. % Receive training and data files and folders from Master
   ```
   Condor_submit EngPT.Condor.txt
   Repeat
   ```
2. % Partition filtered training set into training, validation and test sets
   ```
   tr = divide_data(p,t);
   ```
3. % Initialise weights
   ```
   [IW,LW] = init_weights_bias(tr);
   ```
4. Train the network
   ```
   [tr_rprop,IW,LW] = train_rprop(IW,LW,tr,phen,hu);
   ```
5. % Evaluate trained network on Full training set

```
op1 = use_feedforward_nw(IW,LW,P,T);
```

6. % Calculate performance based on mse(msereg)
```
perf_trset = mse(err1);
```
7. % Calculate performance based on classification accuracy on full training set
```
result = class_accu(T,op1);
```
8. % Evaluate network on generalisation dataset
```
op2 = use_feedforward_nw(IW,LW,pg,tg);
```
9. % Calculate performance based on classification accuracy on generalisation/test set
```
result_gen = class_accu(tg,op2);
```
% Save the results in folders provided, send them back to master and release the slave node (core)

The MATLAB code is presented below:

1. Generate initial population

```
function [twinpop1,twinpop2,PhenTw1,PhenTw2] =
generate_init_population(count)
% This function generates the initial population of 200 twins; 100 twinpop1
% 100 twinpop2 [for first generation only]
warning('off','MATLAB:dispatcher:InexactCaseMatch')
initpop = crtbp(100,80);
parent1 = initpop(1:50,:);
parent2 = initpop(51:100,:);
% considering parent1 as *fathers*
for rows = 1:size(parent1,1)
k = 1;
for j = 1:2:size(parent1,2)
chromo1(rows,k) = parent1(rows,j);
k = k + 1;
end
m = 1;
for n = 2:2:size(parent1,2)
chromo2(rows,m) = parent1(rows,n);
m = m + 1;
end
end
clear j k m n rows
% combining the chromosomes into 1 matrix to perform xover
x = 1;
for row = 1:size(parent1,1)
newparent1(x,:) = chromo1(row,:);
x = x + 1;
newparent1(x,:) = chromo2(row,:);
x = x + 1;
end
clear x row
warning('off','MATLAB:dispatcher:InexactCaseMatch')
 % code for generating sperms
sperm1 = xovsp(newparent1,1);
sperm2 = xovsp(newparent1,1);
sperm3 = xovmp(newparent1,1);
sperm4 = xovsprs(newparent1,1);
  %code to generate parent2, say mother and splitting it's genome in two
% equal halves
for rows = 1:size(parent2,1)
k = 1;
for j = 1:2:size(parent2,2)
```

```
chromoI(rows,k) = parent2(rows,j);
k = k + 1;
end
m = 1;
for n = 2:2:size(parent2,2)
chromoII(rows,m) = parent2(rows,n);
m = m + 1;
end
end
clear rows k j m n
% combining the chromosomes into 1 matrix to perform xover
x = 1;
for row = 1:size(parent2,1)
newparent2(x,:) = chromoI(row,:);
x = x + 1;
newparent2(x,:) = chromoII(row,:);
x = x + 1;
end
clear row x
% code for generating eggs
egg1 = xovsp(newparent2,1);
egg2 = xovsp(newparent2,1);
egg3 = xovmp(newparent2,1);
egg4 = xovsprs(newparent2,1);
  clear chromo1 chromo2 chromoI chromoII parent1 parent2 newparent1 ...
    newparent2
 % positional recombination of sperms and eggs to form offspring
for rows = 1:100
x = 1;
for y = 1:40
offspring1(rows,x) = sperm1(rows,y);
offspring2(rows,x) = sperm2(rows,y);
offspring3(rows,x) = sperm3(rows,y);
offspring4(rows,x) = sperm4(rows,y);
 x = x+1;
offspring1(rows,x) = egg1(rows,y);
offspring2(rows,x) = egg2(rows,y);
offspring3(rows,x) = egg3(rows,y);
offspring4(rows,x) = egg4(rows,y);
 x = x+1;
end
end
clear sperm1 sperm2 sperm3 sperm4 sperm5 sperm6 egg1 egg2 egg3 egg4...
    egg5 egg6 rows x y
 % Creating the required population of twins - twinpop1 and twinpop2...
... keeping in mind similarity. Both populations are (100 X 80)...
... Starting from top - Row 1 &2 havetwo DZs and 3rd & 4th have MZs...
... and so on.
 % The code is as follows:
k = 1;
for a = 1:2:100
    pop1(1,:) = offspring1(a,:);
    pop1(2,:) = offspring1(a+1,:);
    pop1(3,:) = offspring2(a,:);
    pop1(4,:) = offspring2(a+1,:);
    pop1(5,:) = offspring3(a,:);
    pop1(6,:) = offspring3(a+1,:);
    pop1(7,:) = offspring4(a,:);
    pop1(8,:) = offspring4(a+1,:);
S1 = size(pop1,1);
w = round(0.5+rand*S1);
```

```
DZ1 = pop1(w,:);
if mod(w,2) == 0
pop1(w-1:w,:) = [];
else
pop1(w:w+1,:) = [];
end
S2 = size(pop1,1);
for b = 1:size(S2,1)
pattern = pop1(b,:);
hd = @(DZ1,pattern)sum(DZ1(:)~=pattern(:));
dist(b,:) = [(80 - hd(DZ1,pattern)) b];
end
ch2 = min(dist(:,1));
for c = 1:size(dist,1)
if dist(c,1) == ch2
ab = c;
break
end
end
 val = dist(ab,2);
 DZ2 = pop1(val,:);
if mod(val,2) == 0
pop1(val-1:val,:) = [];
else
pop1(val:val+1,:) = [];
end
S3 = size(pop1,1);
pop = pop1;
S3 = size(pop1,1);
 twinpop1(k,:) = DZ1;
 twinpop2(k,:) = DZ2;
k = k+1;
w = round(0.5+rand*S3);
MZ1 = pop1(w,:);
if mod(w,2) == 0
pop1(w-1:w,:) = [];
else
pop1(w:w+1,:) = [];
end
twinpop1(k,:) = MZ1;
 twinpop2(k,:) = MZ1;
k = k+1;
end
clear k a DZ1 DZ2 MZ1 MZ2 S1 S2 S3 S4 S5 ab b c ch2 dist hd offspring1...
    offspring2 offspring3 offspring4 offspring5 offspring6 pattern pop...
    pop1 val w
 FieldD = [10 10 10 10 10 10 10 10; 10 0.07 0 0.0625 0.2 0.05 1 1; 500 ...
    0.1 0.5 4 0.6 0.5 1 1; 0 0 0 0 0 0 0 0; 0 0 0 0 0 0 0 0; ...
    1 1 1 1 1 1 1 1; 1 1 1 1 1 1 1 1];
%for converting genotype into phenotype
PhenTw1 = bs2rv(twinpop1,FieldD);
PhenTw2 = bs2rv(twinpop2,FieldD);
clear initpop FieldD
cd NewPop
cd (['npop',num2str(count)]);
save twinpop1
save twinpop2
save PhenTw1
save PhenTw2
clear all
load('PhenTw1.mat')
```

```
clearvars -except PhenTw1
save PhenTw1
clear all
load PhenTw2
clearvars -except PhenTw2
save PhenTw2
clear all
load twinpop1
clearvars -except twinpop1
save twinpop1
clear all
load twinpop2
clearvars -except twinpop2
save twinpop2
clear all
```

2. Generate SES and filtered training sets

```
function [ses,fam_data] = createfamdata(inputs,targets)
% creates family datasets for all networks based on their ses values
% step1: to generate 100 ses values
for i = 1:100
    ses(i,1) = 0.6+rand*0.4;
end
clear i
 save ses
% step 2: to generate family datasets using the ses values generated above
probe = rand(508,1);
for i = 1:100           % no. of networks in population
    datai = [];
    datat = [];
    for j = 1:size(inputs,1)      % no. of patterns in dataset, mostly 500
        temp = 1-ses(i,1);
        if probe(j,1)>temp
            datai(j,:) = inputs(j,:);
            datat(j,:) = targets(j,:);
        else
            datai = datai;
            datat = datat;
        end
    end
    fam_data{i,1} = datai;
    fam_data{i,2} = datat;
end
 clear i
for i = 1:100
diff = size(inputs,1) - size(fam_data{i,1},1);
if diff > 0
fam_data{i,1}(end+1:end+diff,:) = 0;
fam_data{i,2}(end+1:end+diff,:) = 0;
else
fam_data{i,1} = fam_data{i,1};
fam_data{i,2} = fam_data{i,2};
end
end
clear i probe datai datat j temp diff
save fam_data
```

3. Initialise individual network folders having specific family (shared-environment) data and phenotype

```
function initialise
j = 1;
  for index=0:99
      load fam_data
      load PhenTw1
    mkdir(['network',num2str(index)])
    addpath (['network',num2str(index)])
    cd (['network',num2str(index)])
    netip = cell2mat(fam_data(j,1));
    netop = cell2mat(fam_data(j,2));
    phen = PhenTw1(j,:);
    j = j+1;
    save (['netip'],'netip');
    save (['netop'],'netop');
    save (['phen'],'phen');
    cd ..
  end
```

4. Partition filtered training set into training, validation and test sets

```
function tr = divide_data(pn,tn)
% Divides the normalized data into training, validation and testing groups
trratio = 0.75;
valratio = 0.25;
testratio = 0;
totalratio = trratio + valratio + testratio;
%testper = testratio/totalratio;
valper = valratio/totalratio;
q = numsamples(pn);
numval = round(valper*q);
%numtest = round(testper*q);
numtrain = q - numval;% - numtest;
allind = randperm(q);
trind = sort(allind(1:numtrain));
valind = sort(allind(numtrain + (1:numval)));
%testind = sort(allind(numtrain + numval + (1:numtest)));
trainP = pn(:,trind);
valP = pn(:,valind);
%testP = pn(:,testind);
trainT = tn(:,trind);
valT = tn(:,valind);
%testT = tn(:,testind);
%trdata = [{{trainP},{valP},{testP}} {{trainT},{valT},{testT}}];
tr.training.inputs = trainP;
tr.training.targets = trainT;
tr.training.indices = trind;
tr.validation.inputs = valP;
tr.validation.targets = valT;
tr.validation.indices = valind;
%tr.testing.inputs = testP;
%tr.testing.targets = testT;
%tr.testing.indices = testind;
%data_set_name = 'TrainingDataset';
clear trratio valratio testratio totalratio testper valper q numval ...
```

```
    numtest numtrain allind trind valind testind trainP valP ...
    testP trainT valT testT;
    save tr;
end
```

5. Initialise ANN weights

```
function [IW,LW] = init_weights_bias(tr)
% Generates the initial weights and biases
num_ip = size(tr.training.inputs,1);
num_op = size(tr.training.targets,1);
load phen  % the genome with values for all network parameters or genes
num_hid = roundoff(phen(1,1),0);
ai = -2.38/(sqrt(num_ip));
bi = +2.38/(sqrt(num_ip));
al = -2.38/(sqrt(num_hid));
bl = +2.38/(sqrt(num_hid));
IW = ai+(bi-ai)*rand(num_hid,num_ip+1); % initial weights (input-hidden)
LW = al+(bl-al)*rand(num_op,num_hid+1);   % layer weights (hidden-output)
```

6. Log sigmoid function

```
function Yhid = logsigmoid(x)
% This function calculates the activation or output of hidden layer ...
... by considering the effect of *temp. or slope of activation* gene
load phen
Yhid = 1./(1+(exp(-x*(phen(1,4)))));
```

7. Create feed-forward network

```
function [Yout,mynet] = compute_feedforward_nw(IW,LW,inputs,targets)
% This function creates the feedforward network
Pf = inputs;
Tf = targets;
% augment inputs by adding an extra row of ones
Pf = [Pf;ones(1,size(Pf,2))];
% calculate weighted inputs
Wip = IW*Pf;
% Apply logsigmoid transfer function [Layer 1: I/p to hidden]
Yhid = logsigmoid(Wip);  % this is output of hidden layer
% Augment Yhid by adding an extra row of ones
Yhid_new = [Yhid;ones(1,size(Yhid,2))];
% calculate weighted layer input [Layer2: Hidden-Output]
Wlp = LW*Yhid_new;
% Apply log sigmoid activation function to get final network output
Yout = logsigmoid(Wlp);   % Network output
Enet = Tf-Yout;  % Network error (actual - desired)
% save these values in a structure
mynet.input.weights = IW;
mynet.input.augip = Pf;
mynet.input.weghtedip = Wip;
mynet.hidden.output = Yhid;
mynet.layer.weights = LW;
mynet.layer.input = Yhid_new;
mynet.layer.output = Yout;
mynet.network.error = Enet;
```

```
clear Pf Tf Wip Yhid Yhid_new Wlp Enet
save mynet
```

8. Train the network

```
function [tr_rprop,IW,LW] = Train_Rprop_test1(IW,LW,tr,phen,hu)
% Parameters
epochs = 100;
%time = inf;
goal = 1e-03;
%min_grad = 1e-10;
delta_inc = 1.2;
delta_dec = 0.5;
delta0 = phen(1,2);
deltamax = 50;
deltamin = 1e-06;

% for new termination condition
TrainPerf_old = 0;      % idea being that intial value should be random and
large so that we can choose smaller values thereafter
bestnet = {};           % initially an empty structure
step = 0;
maxstep = 20;           % so as to stop training if ValPerf does not
decrease for 20 epochs
best_epoch = 1;

% to calculate classification accuracy
load ('inputs.mat')
load ('targets.mat')

% initialisation
W = getW(IW,LW);

% train
deltaX = delta0*ones(size(W));
deltaMax = deltamax*ones(size(W));
deltaMin = deltamin*ones(size(W));
%gX = zeros(size(W));
gX_old = zeros(size(W));

% calculate network performance and gradient
for epoch = 1:epochs
    [~,mynet] =
compute_feedforward_nw(IW,LW,tr.training.inputs,tr.training.targets);
    [ValPerf] = eval_network(IW,LW,tr);
    %ValP(epoch,1) = ValPerf;
    NetPerf = calcperf(mynet.network.error);
    [G1,G2] = calc_grad(mynet,hu);
    gX = getW(G2,G1);
    % apply RPROP update
    ggX = gX.*gX_old;
    deltaX = min(deltaX*delta_inc,deltaMax).*(ggX>0) + ...
    max(deltaX*delta_dec,deltaMin).*(ggX<0) + ...
    deltaX.*(ggX==0);
    dW = (sign(gX).*deltaX);
    W = W + dW;
    gX_old = gX;
    [IW,LW] = vec2mat(W,IW,LW);
    train{epoch,1} = IW;
```

```
    train{epoch,2} = LW;
    train{epoch,3} = NetPerf;
    train{epoch,4} = ValPerf;
    train{epoch,5} = mynet;
    IW = train{epoch,1};
    LW = train{epoch,2};
    % to calculate classification performance at each epoch
    P = inputs';
    T = targets';
    op1 = use_feedforward_nw(IW,LW,P);
    err1 = T-op1;
    perf_trset = mse(err1);
    [~,final_corr] = class_accu(op1,T);     % perf. on full training set
    result(epoch,1) = final_corr;
    result(epoch,2) = perf_trset;
    TrainPerf = final_corr;
    % to check validation error termination condition
    if TrainPerf > TrainPerf_old
        TrainPerf_old = TrainPerf;
        bestnet = mynet;
        best_epoch = epoch;
        final_op = {TrainPerf_old bestnet best_epoch};
        step = 0;
    elseif TrainPerf <= TrainPerf_old
        TrainPerf_old = TrainPerf_old;
        bestnet = bestnet;
        best_epoch = best_epoch;
        step = step+1;
        if step == maxstep
            final_op = {TrainPerf_old bestnet best_epoch};
            break
        end
    end
    % check the error goal condition
    if NetPerf < goal
        final_op = {TrainPerf_old bestnet best_epoch};
        break
    elseif epoch == epochs
        final_op = {TrainPerf_old bestnet best_epoch};
    end
    %display(epoch)
end
tr_rprop{1,1} = train(1,:);
tr_rprop{2,1} = train(best_epoch,:);
tr_rprop{3,1} = result;
tr_rprop{4,1} = final_op;
%tr_rprop{5,1} = ValP;
tr_rprop{5,1} = result(best_epoch,:);
%clearvars except result tr_rprop final_op IW LW
```

9. Evaluate the network

```
function [op1] = eval_network(IW,LW,tr)
% this function computes the network's validation and test performance
val_op = use_feedforward_nw(IW,LW,tr.validation.inputs);
val_error = tr.validation.targets - val_op;
op1 = meansqr(val_error);
%test_op = use_feedforward_nw(IW,LW,tr.testing.inputs);
%test_error = tr.testing.targets - test_op;
```

```
%op2 = meansqr(test_error);
```

10. Evaluate the network on validation and test set and on full training set

```
function [Yout] = use_feedforward_nw(IW,LW,inputs)
% This function uses the feedforward network for validation and testing
% datasets
Pf = inputs;
% augment inputs by adding an extra row of ones
 Pf = [Pf;ones(1,size(Pf,2))];
 % calculate weighted inputs
Wip = IW*Pf;
 % Apply logsigmoid transfer function [Layer 1: I/p to hidden]
 Yhid = logsigmoid(Wip);  % this is output of hidden layer
 % Augment Yhid by adding an extra row of ones
Yhid_new = [Yhid;ones(1,size(Yhid,2))];
 % calculate weighted layer input [Layer2: Hidden-Output]
Wlp = LW*Yhid_new;
 % Apply log sigmoid activation function to get final network output
Yout = logsigmoid(Wlp);   % Network output
```

11. Calculate mean-squared error

```
function perf = calcperf(a)
% function to determine network's meansqr performance.
perf = mse(a);
```

12. Calculate gradient

```
function[G1,G2] = calc_grad(mynet,hu)
% function to calculate gradients, delE/delW
y2 = mynet.layer.output;
e = mynet.network.error;
w2 = mynet.layer.weights;
y1 = mynet.hidden.output;
y1_new = mynet.layer.input;
p = mynet.input.augip;
 % gradient1 for output-hidden layer
delta1 = y2.*(1-y2).*e;
G1 = delta1*y1_new';
 % gradient 2 for hidden-input layer
delta2 = (w2(:,1:hu)'*delta1.*y1.*(1-y1));
G2 = delta2*p';
```

13. Calculating class accuracy for English past tense task

```
function [final_err, final_corr] = class_accu(act_output,des_output)
% this bit calculates the actual performance
%act_output = a{1,3};
des_output = des_output';
act_output = act_output';
act_output = ApplyThreshold(act_output);
%des_output = targets;
[categorymatrix, CountCorrect, CountError] =
FinalClassificationCode(act_output,des_output);
```

```
% this section calculates the performance improvement 1
error = CountError;
correct = CountCorrect;
count = 0;
red_error = 0;
inc_correct = 0;
for q = 1:508
    actop = categorymatrix{q,1};
    desop = categorymatrix{q,3};
    if (actop == 2 && desop == 3) || (actop == 2 && desop == 4) || (actop
== 3 && desop == 2) || (actop == 3 && desop == 4) || (actop == 4 && desop
== 2) || (actop == 4 && desop == 3)
    count = count + 1;
    end
end
red_error = error - count;
inc_correct = correct + count;
 % this section calculates the performance improvement 2
err = red_error;
corr = inc_correct;
newcount = 0;
red_err = 0;
inc_corr = 0;
for s = 1:508
    actbit = categorymatrix{s,2};
    desbit = categorymatrix{s,4};
    act = categorymatrix{s,1};
    des = categorymatrix{s,3};
    if (act == 5 && des == 2) || (act == 5 && des == 3) || (act == 5 && des
== 4)
    hd = @(actbit,desbit)sum(actbit(:)~=desbit(:));
    answer = hd(actbit, desbit);
        if answer <= 1
        newcount = newcount + 1;
        end
    end
end
red_err = err - newcount;
inc_corr = corr + newcount;
 %% this section calculates the performance improvement 3
err1 = red_err;
corr1 = inc_corr;
newcount1 = 0;
final_err = 0;
final_corr = 0;
for g = 1:508
    actbit1 = categorymatrix{g,2};
    desbit1 = categorymatrix{g,4};
    act1 = categorymatrix{g,1};
    des1 = categorymatrix{g,3};
    if act1 == 5 && des1 == 1
        hd = @(actbit1,desbit1)sum(actbit1(:)~=desbit1(:));
        answer1 = hd(actbit1, desbit1);
            if answer1 <= 1
            newcount1 = newcount1 + 1;
            end
    end
end
final_err = err1 - newcount1;
final_corr = corr1 + newcount1;
```

14. Calculate verb-category wise accuracy

```
function [categorymatrix, CountCorrect, CountError] =
FinalClassificationCode(act_output,des_output)
allomorph = cell(4,2);
allomorph{1,1} = [0 0 0 0 0];
allomorph{2,1} = [0 0 1 0 1];
allomorph{3,1} = [0 1 1 0 0];
allomorph{4,1} = [0 1 0 1 0];
allomorph{1,2} = 1;
allomorph{2,2} = 2;
allomorph{3,2} = 3;
allomorph{4,2} = 4;
categorymatrix = cell(4);
CountCorrect = 0;
CountError = 0;
 for i = 1:508
        for j = 1:508
        o1 = act_output(i,1:19);
        o2 = act_output(i,20:38);
        o3 = act_output(i,39:57);
        o4 = act_output(i,58:62);
        d1 = des_output(j,1:19);
        d2 = des_output(j,20:38);
        d3 = des_output(j,39:57);
        d4 = des_output(j,58:62);
        hd = @(o1,d1)sum(o1(:)~=d1(:));
        dist1 = hd(o1,d1);
        hd = @(o2,d2)sum(o2(:)~=d2(:));
        dist2 = hd(o2,d2);
        hd = @(o3,d3)sum(o3(:)~=d3(:));
        dist3 = hd(o3,d3);
        %if dist1 && dist2 < 2 || dist2 && dist3 < 2 || dist1 && dist3 < 2
        if dist1 < 2 && dist2 < 2 && dist3 < 2
            hd = @(o4,d4)sum(o4(:)~=d4(:));
            dist4 = hd(o4,d4);
                        if dist4 == 0
                CountCorrect = CountCorrect + 1;
                for s = 1:4
                    allm_dist = d4 - allomorph{s,1};
                    if allm_dist == 0
                categorymatrix{i,1} = allomorph{s,2};
                categorymatrix{i,2} = d4;
                categorymatrix{i,3} = allomorph{s,2};
                categorymatrix{i,4} = o4;
                    end
                end
                break
                else
                CountError = CountError + 1;
               for k = 1:4
                    act_allm_dist = o4 - allomorph{k,1};
                    des_allm_dist = d4 - allomorph{k,1};
                    if act_allm_dist == 0
                        categorymatrix{i,1} = allomorph{k,2};
                        categorymatrix{i,2} = o4;
                    elseif k == size(k)
                     categorymatrix{i,1} = 5;
                     categorymatrix{i,2} = o4;
                     end
```

```
                    if des_allm_dist == 0
                        categorymatrix{i,3} = allomorph{k,2};
                        categorymatrix{i,4} = d4;
                    end
                end
            end
            break

        else
            if j == size(j)
               categorymatrix{i,1} = 100;
               categorymatrix{i,2} = 100;
               categorymatrix{i,3} = 100;
               categorymatrix{i,4} = o4;
            end
        end
    end
end
```

15. Collect results from all networks

```
function [twin1, netindex,indices] = analyseresults(count)
cd (['gen',num2str(count)]);
i = 1;
k = 1;
index = 0;
for net = 0:99
    cd (['network',num2str(net)]);
    a = exist ('tw1results.mat');
    if a > 0
        load tw1results
        twin1(i,:) = tw1results(1,:);
        indices(net+1,:) = twin1(i,:);
        netindex(i) = net;
        i = i+1;
    elseif a == 0
        index = index+1;
        store(k) = net;
        indices(net+1,1:end) = 0.01;
        k = k+1;
    end
    cd ..
end
clear i k a
save twin1
save indices
```

16. Compute consolidated results for each generation

```
function [genavg] = analyzeperf(twin1,count)
cd (['gen',num2str(count)]);
load twin1.mat
test = [];
test(:,1) = twin1(:,1);
test(:,2) = twin1(:,2);
a = size(test,1);
i = 1;
while i <= a
```

```
if test(i,1) <= 60
test(i,:) = [];
a = size(test,1);
i = i;
else
a = a;
i = i+1;
end
end
genavg{2,1} = 'train_perf';
genavg{3,1} = 'gen_perf';
%genavg{4,1} = 'ca_tr';
%genavg{5,1} = 'ca_gen';
genavg{1,2} = 'min';
genavg{1,3} = 'max';
genavg{1,4} = 'mean';
genavg{1,5} = 'std';
j = 1;
for i = 2:3
genavg{i,2} = min(test(:,j));
genavg{i,3} = max(test(:,j));
genavg{i,4} = mean(test(:,j));
genavg{i,5} = std(test(:,j));
j = j+1;
end
clear i j
save genavg
```

17. Calculate heritability and environmentability

```
function [rmz,rdz,h2,c2,e2] = calc_herit(count)
% This fn to be used gen2 onwards; where count = generation count
% first extract results
cd twin1
cd (['gen',num2str(count)]);
load indices.mat
caccu1 = indices(:,1);
cd ..
cd ..
cd twin2
cd (['gen',num2str(count)]);
load indices2.mat
caccu2 = indices2(:,1);
cd ..
cd ..
 % seperate them into mz-dz twins
j = 1;
for i = 1:4:100
    dz1(j,:) = caccu1(i,:);
    j = j+1;
    dz1(j,:) = caccu1(i+1,:);
    j = j+1;
end
clear i j
 k = 1;
for l = 3:4:100
    mz1(k,:) = caccu1(l,:);
    k = k+1;
    mz1(k,:) = caccu1(l+1,:);
    k = k+1;
```

```
end
clear k l
 j = 1;
for i = 1:4:100
    dz2(j,:) = caccu2(i,:);
    j = j+1;
    dz2(j,:) = caccu2(i+1,:);
    j = j+1;
end
clear i j
 k = 1;
for l = 3:4:100
    mz2(k,:) = caccu2(l,:);
    k = k+1;
    mz2(k,:) = caccu2(l+1,:);
    k = k+1;
end
clear k l
 % Remove the non-trained networks from list
a = size(mz1,1);
i = 1;
while i <= a
    x = mz1(i);
    y = mz2(i);
    if (x == 0.01) || (y == 0.01)
        mz1(i) = [];
        mz2(i) = [];
        a = size(mz1,1);
    end
    i = i+1;
end
 clear i x y a
 j = 1;
b = size(dz1,1);
while j <= b
    x = dz1(j);
    y = dz2(j);
    if (x == 0.01) || (y == 0.01)
        dz1(j) = [];
        dz2(j) = [];
        b = size(dz1,1);
    end
    j = j+1;
end
 clear j x y b
 % claculate correlations
 rmz = corr(mz1,mz2);
rdz = corr(dz1,dz2);
 % heritability
h2 = 2*(rmz-rdz);
 %shared env
c2 = rmz - h2;
 %non shared env
e2 = 1-rmz;
 clear mz1 mz2 dz1 dz2
 cd heritability
cd (['G',num2str(count)])
save rmz
save rdz
save h2
save c2
```

```
save e2
cd ..
```

18. Selection: Roulette-Wheel

```
function Sel_Parents = RW_selection(count)
cd EngPT
cd twin1
cd (['gen',num2str(count)]);
load twin1
Fitness = twin1(:,3);
for i = 1:numel(Fitness)
    ind_fitness(i,1) = Fitness(i,1)/508;
end
clear i
summ = sum(ind_fitness);
p = [];
k = 0;
for i = 1:numel(ind_fitness)
a = [ind_fitness(i,1)/summ];
k = k + a;
p(i,2) = k;
end
Sel=[];
c=0;
randprob = rand(50,1);
cp = p(:,2);
for y = 1:50
   for z = 1:numel(ind_fitness)
     if cp(z)>=randprob(y)
       for v=1:length(Sel)
          if z==Sel(v)
            c=1;
            break;
          else
            c=0;
          end;
       end
       if c==0
       Sel(y) = z;
       break
       end
     end
    end
end
Sel_Parents = Sel';
clear p k a c cp z i Sel Fitness index randprob summ twin1 twin1results v y
cd ..
cd ..
cd ..
cd NewPop
mkdir(['npop',num2str(count+1)]);
cd (['npop',num2str(count+1)]);
clear count
save Sel_Parents
clear all
load Sel_Parents
clearvars -except Sel_Parents
save Sel_Parents
```

18a. Selection: Truncation

```
function [Select_Parents, Sel_Parents] = trunc_selection(count)
cd EngPT
cd twin1
cd (['gen',num2str(count)]);
load twin1
Fitness = twin1(:,1);
for i = 1:numel(Fitness)
    ind_fitness(i,1) = Fitness(i,1)/508;
end
clear i
summ = sum(ind_fitness);
for i = 1:numel(ind_fitness)
test_par(i,1) = [ind_fitness(i,1)/summ];
test_par(i,2) = i;
end
sort_test_par = sort(test_par);
for k = 1:size(sort_test_par,1);
    parents(k,1) = sort_test_par(k,1);
    len = length(test_par);
    for j = 1:len
        if parents(k,1) == test_par(j,1)
            parents(k,2) = test_par(j,2);
            test_par(j,:) = [];
            len = length(test_par);
            break
        end
    end
end
final_res = flipud(parents);
Select_Parents = final_res(1:50,:);
Sel_Parents = Select_Parents(:,2);
%clearvars -except Sel_Parents
cd ..
cd ..
cd ..
cd NewPop
%mkdir(['npop',num2str(count+1)]);
cd (['npop',num2str(count+1)]);
clear count
save Sel_Parents
save Select_Parents
clear all
load Sel_Parents
clearvars -except Sel_Parents
save Sel_Parents
load Select_Parents
clearvars -except Select_Parents
save Select_Parents
```

19. Class accuracy for auto and arbitrary association tasks

```
function result = class_accu(desop,actop)
desop = desop';
actop = actop';
actop = ApplyThreshold(actop);
```

```
countcorrect = 0;
for i = 1:size(actop,1);
    for j = 1:size(actop,1);
        a1 = actop(i,:);
        d1 = desop(j,:);
        hd = @(a1,d1)sum(a1(:)~=d1(:));
        dist1 = hd(a1,d1);
        if dist1 == 0
            countcorrect = countcorrect+1;
            break
        end
    end
end
result = countcorrect;
```

19a. Class accuracy for categorisation & categorisation with exceptions tasks

```
function result = class_accu(desop,actop)
desop = desop';
for k = 1:size(desop,1)
    sampletargets(k,:) = desop(k,1:6:60);
end
clear k
sampletargets = sampletargets';
 %actop = train{epoch+1,8};
actop = actop';
 for l = 1:size(actop,1)
    sampleop(l,:) = actop(l,1:6:60);
end
clear l
sampleop = sampleop';
[c,cm,ind,per] = confusion(sampletargets,sampleop);
result{1,1} = (1-c)*100;
result{2,1} = cm;
result{3,1} = ind;
result{4,1} = per;
```

20. Generate twin populations (gen 2 onwards)

```
function [twinpop1,twinpop2,PhenTw1,PhenTw2] =
generate_new_population(Sel_Parents,twinpop1,count)
% This function generates the subsequent new populations of 200 twins each;
100 twinpop1
% 100 twinpop2. Goto (count-1)th generation and load twinpop1 from there.
warning('off','MATLAB:dispatcher:InexactCaseMatch')
breedpop = Sel_Parents;
parentpop = twinpop1;
% extract the selected parents from twinpop1
newpop = [];
for i = 1:50
   test = breedpop(i,1);
    newpop(i,:) = parentpop(test,:);
end
clear i test breedpop parentpop
%save newpop
% For splitting it into parent1(fathers) and parent2 (mothers)
npop1 = newpop(1:25,:);
```

```
npop2 = newpop(26:50,:);
% Consider parent1 as fathers
% code for splitting the genome of parent 1(represented by npop1) into 2
% chromosomes
%load npop1
chromo1 = [];
chromo2 = [];
for rows = 1:25
k = 1;
for j = 1:2:80
chromo1(rows,k) = npop1(rows,j);
k = k + 1;
end
m = 1;
for n = 2:2:80
chromo2(rows,m) = npop1(rows,n);
m = m + 1;
end
end
clear j k m n rows
% combining the chromosomes into 1 matrix to perform xover
x = 1;
for row = 1:25
parent1(x,:) = chromo1(row,:);
x = x + 1;
parent1(x,:) = chromo2(row,:);
x = x + 1;
end
clear x row
% code for generating sperms
sperm1 = xovsp(parent1,1);
sperm2 = xovsp(parent1,1);
sperm3 = xovmp(parent1,1);
sperm4 = xovsprs(parent1,1);
sperm5 = xovshrs(parent1,1);
sperm6 = xovshrs(parent1,1);
%code to generate parent2, say mother and splitting it's genome in two
% equal halves
%npop2 = crtbp(25,80);
%load npop2
chromoI = [];
chromoII = [];
for rows = 1:25
k = 1;
for j = 1:2:80
chromoI(rows,k) = npop2(rows,j);
k = k + 1;
end
m = 1;
for n = 2:2:80
chromoII(rows,m) = npop2(rows,n);
m = m + 1;
end
end
clear rows k j m n
% combining the chromosomes into 1 matrix to perform xover
x = 1;
for row = 1:25
parent2(x,:) = chromoI(row,:);
x = x + 1;
parent2(x,:) = chromoII(row,:);
```

```
x = x + 1;
end
clear row x
% code for generating eggs
egg1 = xovsp(parent2,1);
egg2 = xovsp(parent2,1);
egg3 = xovmp(parent2,1);
egg4 = xovsprs(parent2,1);
egg5 = xovshrs(parent2,1);
egg6 = xovshrs(parent2,1);
clear chromo1 chromo2 chromoI chromoII parent1 parent2 npop1 npop2
% positional recombination of sperms and eggs to form offspring
for rows = 1:50
x = 1;
for y = 1:40
offspring1(rows,x) = sperm1(rows,y);
offspring2(rows,x) = sperm2(rows,y);
offspring3(rows,x) = sperm3(rows,y);
offspring4(rows,x) = sperm4(rows,y);
offspring5(rows,x) = sperm5(rows,y);
offspring6(rows,x) = sperm6(rows,y);
x = x+1;
offspring1(rows,x) = egg1(rows,y);
offspring2(rows,x) = egg2(rows,y);
offspring3(rows,x) = egg3(rows,y);
offspring4(rows,x) = egg4(rows,y);
offspring5(rows,x) = egg5(rows,y);
offspring6(rows,x) = egg6(rows,y);
x = x+1;
end
end
clear sperm1 sperm2 sperm3 sperm4 sperm5 sperm6 egg1 egg2 egg3 egg4 egg5
egg6 rows x y
% Creating the required population of twins - twinpop1 and twinpop2 keeping
in mind
% similarity. Both populations are (100 X 80). Starting from top - Row 1 &2
% have two DZs and 3rd & 4th have MZs and so on.
% The code is as follows:
k = 1;
for a = 1:2:50
    pop1(1,:) = offspring1(a,:);
    pop1(2,:) = offspring1(a+1,:);
    pop1(3,:) = offspring2(a,:);
    pop1(4,:) = offspring2(a+1,:);
    pop1(5,:) = offspring3(a,:);
    pop1(6,:) = offspring3(a+1,:);
    pop1(7,:) = offspring4(a,:);
    pop1(8,:) = offspring4(a+1,:);
    pop1(9,:) = offspring5(a,:);
    pop1(10,:) = offspring5(a+1,:);
    pop1(11,:) = offspring6(a,:);
    pop1(12,:) = offspring6(a+1,:);
S1 = size(pop1,1);
w = round(0.5+rand*S1);
DZ1 = pop1(w,:);
if mod(w,2) == 0
pop1(w-1:w,:) = [];
%pop1(w-1,:) = [];
else
pop1(w:w+1,:) = [];
%pop1(w,:) = [];
```

```
end
S2 = size(pop1,1);
for b = 1:size(S2,1)
pattern = pop1(b,:);
hd = @(DZ1,pattern)sum(DZ1(:)~=pattern(:));
dist(b,:) = [(80 - hd(DZ1,pattern)) b];
end
ch2 = min(dist(:,1));
for c = 1:size(dist,1)
if dist(c,1) == ch2
ab = c;
break
end
end
 val = dist(ab,2);
 DZ2 = pop1(val,:);
if mod(val,2) == 0
pop1(val-1:val,:) = [];
%pop1(val,:) = [];
else
pop1(val:val+1,:) = [];
%pop1(val,:) = [];
end
S3 = size(pop1,1);
pop = pop1;
S3 = size(pop1,1);
 twinpop1(k,:) = DZ1;
 twinpop2(k,:) = DZ2;
k = k+1;
 w = round(0.5+rand*S3);
DZ1 = pop1(w,:);
if mod(w,2) == 0
pop1(w-1:w,:) = [];
%pop1(w-1,:) = [];
else
pop1(w:w+1,:) = [];
%pop1(w,:) = [];
end
S4 = size(pop1,1);
for b = 1:S4
pattern = pop1(b,:);
hd = @(DZ1,pattern)sum(DZ1(:)~=pattern(:));
dist(b,:) = [(80 - hd(DZ1,pattern)) b];
end
ch2 = min(dist(:,1));
for c = 1:size(dist,1)
if dist(c,1) == ch2
ab = c;
break
end
end
val = dist(ab,2);
 DZ2 = pop1(val,:);
if mod(val,2) == 0
pop1(val-1:val,:) = [];
%pop1(val,:) = [];
else
pop1(val:val+1,:) = [];
%pop1(val,:) = [];
end
S5 = size(pop1,1);
```

```
twinpop1(k,:) = DZ1;
 twinpop2(k,:) = DZ2;
k = k+1;
w = round(0.5+rand*S5);
MZ1 = pop1(w,:);
if mod(w,2) == 0
pop1(w-1:w,:) = [];
%pop1(w-1,:) = [];
else
pop1(w:w+1,:) = [];
%pop1(w,:) = [];
end
w = round(1+rand*1);
MZ2 = pop1(w,:);
 twinpop1(k,:) = MZ1;
 twinpop2(k,:) = MZ1;
k = k+1;
twinpop1(k,:) = MZ2;
 twinpop2(k,:) = MZ2;
k = k+1;
end
clear k a DZ1 DZ2 MZ1 MZ2 S1 S2 S3 S4 S5 ab b c ch2 dist hd offspring1
offspring2 offspring3 offspring4 offspring5 offspring6 pattern pop pop1 val
w
FieldD = [10 10 10 10 10 10 10 10; 10 0.07 0 0.0625 0.2 0.05 1 1; 500 ...
    0.1 0.5 4 0.6 0.5 1 1; 0 0 0 0 0 0 0 0; 0 0 0 0 0 0 0 0; ...
    1 1 1 1 1 1 1 1; 1 1 1 1 1 1 1 1];
 %for converting genotype into phenotype
PhenTw1 = bs2rv(twinpop1,FieldD);
PhenTw2 = bs2rv(twinpop2,FieldD);
clear FieldD
cd NewPop
cd (['npop',num2str(count)]);
save twinpop1
save twinpop2
save PhenTw1
save PhenTw2
clear all
load('PhenTw1.mat')
clearvars -except PhenTw1
save PhenTw1
clear all
load PhenTw2
clearvars -except PhenTw2
save PhenTw2
clear all
load twinpop1
clearvars -except twinpop1
save twinpop1
clear all
load twinpop2
clearvars -except twinpop2
save twinpop2
clear all
```

Bibliography

Ackley, D., & Littman, M. (1991). Interactions between learning and evolution. *Artificial life II*, *10*, 487-509.

Adam, S. P., Karras, D. A., Magoulas, G. D., & Vrahatis, M. N. (2014). Solving the linear interval tolerance problem for weight initialization of neural networks. *Neural Networks*, *54*, 17-37.

Ahmadizar, F., Soltanian, K., AkhlaghianTab, F., & Tsoulos, I. (2015). Artificial neural network development by means of a novel combination of grammatical evolution and genetic algorithm. *Engineering Applications of Artificial Intelligence*, *39*, 1-13.

Aisbett, J., & Gibbon, G. (1999, July). Cognitive Classification. In *AAAI/IAAI*(pp. 100-107).

Alba, E., Aldana, J. F., & Troya, J. M. (1993). Full automatic ANN design: A genetic approach. In *New Trends in Neural Computation* (pp. 399-404). Springer Berlin Heidelberg.

Albesano, D., Gemello, R., Laface, P., Mana, F., & Scanzio, S. (2006). Adaptation of artificial neural networks avoiding catastrophic forgetting. In*The 2006 IEEE International Joint Conference on Neural Network Proceedings* (pp. 1554-1561). IEEE.

Ali, M. Z., Awad, N. H., Suganthan, P. N., Duwairi, R. M., & Reynolds, R. G. (2016). A novel hybrid Cultural Algorithms framework with trajectory-based search for global numerical optimization. *Information Sciences*, *334*, 219-249.

Anastasiadis, A. D., & Magoulas, G. D. (2006). Analysing the localisation sites of proteins through neural networks ensembles. *Neural Computing & Applications*, *15*(3-4), 277-288.

Androutsopoulos, I., Koutsias, J., Chandrinos, K. V., & Spyropoulos, C. D. (2000). An experimental comparison of naive Bayesian and keyword-based anti-spam filtering with personal e-mail messages. In *Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval* (pp. 160-167). ACM.

Angeline, P. J., Saunders, G. M., & Pollack, J. B. (1994). An evolutionary algorithm that constructs recurrent neural networks. *Neural Networks, IEEE Transactions on*, *5*(1), 54-65.

Annaz, D., Karmiloff-Smith, A., & Thomas, M. S. (2008). The importance of tracing developmental trajectories for clinical child neuropsychology. *Child neuropsychology: Concepts, theory and practice*, *7*.

Ans, B., & Rousset, S. (1997). Avoiding catastrophic forgetting by coupling two reverberating neural networks. *Comptes Rendus de l'Académie des Sciences-Series III-Sciences de la Vie*, *320*(12), 989-997.

Ans, B., & Rousset, S. (2000). Neural networks with a self-refreshing memory: knowledge transfer in sequential learning tasks without catastrophic forgetting. *Connection science*, *12*(1), 1-19.

Argyriou, A., Maurer, A., & Pontil, M. (2008). An algorithm for transfer learning in a heterogeneous environment. In *Machine Learning and Knowledge Discovery in Databases* (pp. 71-85). Springer Berlin Heidelberg.

Arnold, A., Nallapati, R., & Cohen, W. W. (2007). A comparative study of methods for transductive transfer learning. In *Data Mining Workshops, 2007. ICDM Workshops 2007. Seventh IEEE International Conference on* (pp. 77-82). IEEE.

Baldwin, J. M. (1896). A new factor in evolution. *The American naturalist*, *30*(354), 441-451.

Baxter, J. (1993). The evolution of learning algorithms for artificial neural networks. *Complex systems*, 313-326.

Behbood, V., Lu, J., & Zhang, G. (2011). Long term bank failure prediction using fuzzy refinement-based transductive transfer learning. In*Fuzzy Systems (FUZZ), 2011 IEEE International Conference on* (pp. 2676-2683). IEEE.

Behbood, V., Lu, J., & Zhang, G. (2014). Fuzzy refinement domain adaptation for long term prediction in banking ecosystem. *Industrial Informatics, IEEE Transactions on*, *10*(2), 1637-1646.

Belew, R. K. (1990). Evolution, learning, and culture: Computational metaphors for adaptive algorithms. *Complex Systems*, *4*(1), 11-49.

Belew, R. K., McInerney, J., & Schraudolph, N. N. (1990). Evolving networks: Using the genetic algorithm with connectionist learning. In *In*.

Belsky, J., Bakermans-Kranenburg, M. J., & Van IJzendoorn, M. H. (2007). For better and for worse: Differential susceptibility to environmental influences. *Current directions in psychological science*, *16*(6), 300-304.

Ben-David, S., Blitzer, J., Crammer, K., Kulesza, A., Pereira, F., & Vaughan, J. W. (2010). A theory of learning from different domains. *Machine learning*,*79*(1-2), 151-175.

Bengio, S., Bengio, Y., Cloutier, J., & Gecsei, J. (1992). On the optimization of a synaptic learning rule. In *Preprints Conf. Optimality in Artificial and Biological Neural Networks* (pp. 6-8).

Bishop, D. V. M. (2005): DeFries–Fulker analysis of twin data with skewed distributions: Cautions and recommendations from a study of children's use of verb inflections. *Behavior Genetics*, *35*, (pp. 479 – 490). doi:10.1007/s10519-004-1834-7

Blitzer, J., Crammer, K., Kulesza, A., Pereira, F., & Wortman, J. (2008). Learning bounds for domain adaptation. In *Advances in neural information processing systems* (pp. 129-136).

Blitzer, J., McDonald, R., & Pereira, F. (2006). Domain adaptation with structural correspondence learning. In *Proceedings of the 2006 conference on empirical methods in natural language processing* (pp. 120-128). Association for Computational Linguistics.

Blum, C., & Pereira, J. (2016). Extension of the CMSA Algorithm: An LP-based Way for Reducing Sub-instances. In *Proceedings of the Genetic and Evolutionary Computation Conference 2016* (GECCO '16), Tobias Friedrich (Ed.). ACM, New York, NY, USA, 285-292. DOI: http://dx.doi.org/10.1145/2908812.2908830

Blynel, J., & Floreano, D. (2003, April). Exploring the T-maze: Evolving learning-like robot behaviors using CTRNNs. In *Workshops on Applications of Evolutionary Computation* (pp. 593-604). Springer Berlin Heidelberg.

Bornholdt, S., & Graudenz, D. (1992). General asymmetric neural networks and structure design by genetic algorithms. *Neural Networks*, *5*(2), 327-334.

Bottou, L.Y. (1988): Reconnaissance de la Parole par Reseaux Multi-Couches. In Neuro-Nimes '88; Proceedings of the International Workshop on Neural Networks and Their Applications, (pp. 197–217). ISBN: 2-906899-14-3

Bourd, A., (2016). The OpenCL Specification Version: 2.2 Document Revision: 06. Khronos OpenCL Working Group. https://www.khronos.org/registry/cl/specs/opencl-2.2.pdf

Brady, S. A., Braze, D., & Fowler, C. A. (Eds.). (2011). *Explaining individual differences in reading: Theory and evidence*. Psychology Press.

Breiman, L. (1996). Bagging predictors. *Machine learning*, *24*(2), 123-140.

Burt, S. A. (2009). Rethinking environmental contributions to child and adolescent psychopathology: a meta-analysis of shared environmental influences. *Psychological bulletin*, *135*(4), 608.

Cardona, A. B., Togelius, J., & Nelson, M. J. (2013). Competitive coevolution in ms. pac-man. In *2013 IEEE Congress on Evolutionary Computation* (pp. 1403-1410). IEEE

Carroll, J. L., & Seppi, K. (2005). Task similarity measures for transfer in reinforcement learning task libraries. In *Neural Networks, 2005. IJCNN'05. Proceedings. 2005 IEEE International Joint Conference on* (Vol. 2, pp. 803-808). IEEE.

Caruana, R. (1997). Multitask learning. *Machine learning*, *28*(1), 41-75.

Celiberto Jr, L. A., Matsuura, J. P., De Mantaras, R. L., & Bianchi, R. A. (2011). Using cases as heuristics in reinforcement learning: a transfer learning application. In *IJCAI Proceedings-International Joint Conference on Artificial Intelligence* (Vol. 22, No. 1, p. 1211).

Chalmers, D. J. (1990). The evolution of learning: An experiment in genetic connectionism. In *Proceedings of the 1990 connectionist models summer school* (pp. 81-90). San Mateo, CA.

Chandra, A., & Yao, X. (2006). Ensemble learning using multi-objective evolutionary algorithms. *Journal of Mathematical Modelling and Algorithms*, 5(4), 417-445.

Chandra, R. (2015). Competition and collaboration in cooperative coevolution of Elman recurrent neural networks for time-series prediction. *IEEE transactions on neural networks and learning systems*, *26*(12), 3123-3136.

Chater, N., Christiansen, M.H. (2008). Computational models of psycholinguistics. In R. Sun (Ed.), *Cambridge handbook of computational psychology* (pp. 477-504). Cambridge University Press. Doi:10.1017/CBO9780511816772.021

Chawla, N. V., Japkowicz, N., & Kotcz, A. (2004). Editorial: special issue on learning from imbalanced data sets. *ACM Sigkdd Explorations Newsletter*,*6*(1), 1-6.

Chellapilla, K., & Fogel, D. B. (2001). Evolving an expert checkers playing program without using human expertise. *IEEE Transactions on Evolutionary Computation*, *5*(4), 422-428.

Chomsky, N. (1965). Aspects of the theory of syntax. *MIT Press*, Cambridge.

Chomsky, N. (1980). Rules and representations. *Behavioral and brain sciences*, *3*(01), 1-15.

Chong, S., & Whitelaw, E. (2004). Epigenetic germline inheritance. *Current opinion in genetics & development*, *14*(6), 692-696.

Clune, J., Mouret, J. B., & Lipson, H. (2013). The evolutionary origins of modularity. In *Proc. R. Soc. B* (Vol. 280, No. 1755, p. 20122863). The Royal Society.

Collobert, R., & Weston, J. (2008). A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th international conference on Machine learning* (pp. 160-167). ACM.

Cooper, G. M., & Hausman, R. E. (2000). *The cell* (pp. 725-730). Sunderland: Sinauer Associates.

Cottrell, G. W. & Plunkett, K. (1991): Learning the past tense in a recurrent network: Acquiring the mapping from meanings to sounds. In *Proceedings of the Thirteenth Annual Conference of the Cognitive Science Society* (pp. 328-333). Hillsdale NJ: Lawrence Erlbaum Associates.

Dai, W., Yang, Q., Xue, G. R., & Yu, Y. (2007a). Boosting for transfer learning. In *Proceedings of the 24th international conference on Machine learning* (pp. 193-200). ACM.

Dai, W., Xue, G. R., Yang, Q., & Yu, Y. (2007b). Transferring naive bayes classifiers for text classification. In *Proceedings of the national conference on artificial intelligence* (Vol. 22, No. 1, p. 540). Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999.

Dai, W., Chen, Y., Xue, G. R., Yang, Q., & Yu, Y. (2008). Translated learning: Transfer learning across different feature spaces. In *Advances in neural information processing systems* (pp. 353-360).

Darwin, C. (1897). The origin of species by means of natural selection, or, The preservation of favored races in the struggle for life. Vol. 1. *International Science Library*.

Darwin, C. (2009). The origin of species by means of natural selection: or, the preservation of favored races in the struggle for life.

Dasdan, A., & Oflazer, K. (1993). Genetic synthesis of unsupervised learning algorithms. In *Proceedings of the 2nd Turkish symposium on artificial intelligence and ANNs. Department of Computer Engineering and Information Science, Bilkent University, Ankara*.

Daugherty, K., Seidenberg, M.S. (1992): Rules or connections? The past tense revisited. *Proceedings of the 14th Annual Meeting of the Cognitive Science Society* (pp. 259-264). Hillsdale, N.J.: Erlbaum.

Davis, J., & Domingos, P. (2009). Deep transfer via second-order markov logic. In *Proceedings of the 26th annual international conference on machine learning* (pp. 217-224). ACM.

Dawkins, R. (2006). *The selfish gene* (No. 199). Oxford university press.

de Castro, L. N. (2007). Fundamentals of natural computing: an overview. *Physics of Life Reviews*, *4*(1), 1-36.

de Saussure F. 1916. Course in general linguistics. New York, NY: McGraw-Hill

DeFries, J.C., Gervais, M.C., & Thomas, E.A. (1978): Response to 30 generations of selection for open-field activity in laboratory mice. *Behavior Genetics, Volume 8, Issue 1* (pp 3-13).

Ding, S., Li, H., Su, C., Yu, J., & Jin, F. (2013). Evolutionary artificial neural networks: a review. *Artificial Intelligence Review*, *39*(3), 251-260.

Duan, L., Xu, D., & Tsang, I. (2012). Learning with augmented features for heterogeneous domain adaptation. *arXiv preprint arXiv:1206.4660*.

Eaton, E., & Lane, T. (2008). Modeling transfer relationships between learning tasks for improved inductive transfer. In *Machine Learning and Knowledge Discovery in Databases* (pp. 317-332). Springer Berlin Heidelberg.

Ellefsen, K. O., Mouret, J. B., & Clune, J. (2015). Neural modularity helps organisms evolve to learn new skills without forgetting old skills. *PLoS Comput Biol*, *11*(4), e1004128.

Falconer, D. S., & Mackay, T. F. C. (1995). Introduction to Quantitative Genetics. *Longman*, *19*(8), 1.

Fernando, C., Banarse, D., Reynolds, M., Besse, F., Pfau, D., Jaderberg, M., Lanctot, M., and Wierstra, D. (2016). Convolution by Evolution: Differentiable Pattern Producing Networks. In *Proceedings of the Genetic and Evolutionary Computation Conference 2016* (GECCO '16), Tobias Friedrich (Ed.). ACM, New York, NY, USA, 109-116. DOI: http://dx.doi.org/10.1145/2908812.2908890

Ferrell, J.E., (2012). Bistability, Bifurcations, and Waddington's Epigenetic Landscape, In *Current Biology, Volume 22, Issue 11,* (pp R458-R466).

Fister, I., Strnad, D., Yang, X. S., & Fister Jr, I. (2015). Adaptation and hybridization in nature-inspired algorithms. In *Adaptation and Hybridization in Computational Intelligence* (pp. 3-50). Springer International Publishing.

Floreano, D., Dürr, P., & Mattiussi, C. (2008). Neuroevolution: from architectures to learning. *Evolutionary Intelligence*, *1*(1), 47-62.

Fogel, D. B., Fogel, L. J., & Porto, V. W. (1990). Evolving neural networks. *Biological cybernetics*, *63*(6), 487-493.

Fogel, D. B., Wasson, E. C., & Boughton, E. M. (1995). Evolving neural networks for detecting breast cancer. *Cancer letters*, *96*(1), 49-53.

Fogel, D. B. (2006). *Evolutionary computation: toward a new philosophy of machine intelligence* (Vol. 1). John Wiley & Sons.

Fontanari, J. F., & Meir, R. (1991). Evolving a learning algorithm for the binary perceptron. *Network: Computation in Neural Systems*, *2*(4), 353-359.

French, R. M. (1999). Catastrophic forgetting in connectionist networks. *Trends in cognitive sciences*, *3*(4), 128-135.

Freund, Y., & Schapire, R. E. (1995). A desicion-theoretic generalization of on-line learning and an application to boosting. In *European conference on computational learning theory* (pp. 23-37). Springer Berlin Heidelberg.

Fromkin, V., Rodman, R., & Hyams, N. (2013). *An introduction to language*. Cengage Learning.

Fu, X., & Zhang, S. (2013). Evolving neural network ensembles using variable string genetic algorithm for Pattern Classification. In *Advanced Computational Intelligence (ICACI), 2013 Sixth International Conference on* (pp. 81-85). IEEE.

Gao, J., Fan, W., Jiang, J., & Han, J. (2008). Knowledge transfer via multiple model local structure mapping. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining* (pp. 283-291). ACM.

García-Pedrajas, N., Hervás-Martínez, C., & Ortiz-Boyer, D. (2005). Cooperative coevolution of artificial neural network ensembles for pattern classification. *IEEE Transactions on evolutionary computation*, *9*(3), 271-302.

Goldberg, D. E. (2013). *The design of innovation: Lessons from and for competent genetic algorithms* (Vol. 7). Springer Science & Business Media

Goldberg, D. E., & Deb, K. (1991). A comparative analysis of selection schemes used in genetic algorithms. *Foundations of genetic algorithms*, *1*, 69-93.

Greve, R. B., Jacobsen, E. J., & Risi, S. (2016). Evolving Neural Turing Machines for Reward-based Learning. In *Proceedings of the Genetic and Evolutionary Computation Conference 2016* (GECCO '16), Tobias Friedrich (Ed.). ACM, New York, NY, USA, 117-124.

Griffiths, P. (2010). The distinction between innate and acquired characteristics. *Stanford encyclopedia of philosophy*.

Grossberg, S. (1987). Competitive learning: From interactive activation to adaptive resonance. *Cognitive science*, *11*(1), 23-63.

Hansen, L. K., & Salamon, P. (1990). Neural network ensembles. *IEEE transactions on pattern analysis and machine intelligence*, *12*, 993-1001.

Harp, S. A., Samad, T., & Guha, A. (1989). Towards the genetic synthesis of neural network. In *Proceedings of the third international conference on Genetic algorithms* (pp. 360-369). Morgan Kaufmann Publishers Inc.

Harp, S. A., Samad, T., & Guha, A. (1990). Designing application-specific neural networks using the genetic algorithm. In *Advances in neural information processing systems* (pp. 447-454).

Haykin, S. S. (2009). *Neural networks and learning machines* (Vol. 3). Upper Saddle River, NJ, USA: Pearson.

Hemanth, D. J., Vijila, C. K. S., Selvakumar, A. I., & Anitha, J. (2014). Performance improved iteration-free artificial neural networks for abnormal magnetic resonance brain image classification. *Neurocomputing*, *130*, 98-107.

Hinton, G. E., & Nowlan, S. J. (1987). How learning can guide evolution.*Complex systems*, *1*(3), 495-502.

Hinton, G., Deng, L., Yu, D., Dahl, G. E., Mohamed, A. R., Jaitly, N., ... & Kingsbury, B. (2012). Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *IEEE Signal Processing Magazine*, *29*(6), 82-97.

Hinton, G., Vinyals, O., & Dean, J. (2015). Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*.

Hockett CF. 1960. The origins of speech. Sci. Am. 203, 89–96 (doi:10.1038/scientificamerican0960-88)

Howes, L., & Munshi, A. (2015). The OpenCL Specification.

Hu, D. H., & Yang, Q. (2011). Transfer learning for activity recognition via sensor mapping. In *IJCAI Proceedings-International Joint Conference on Artificial Intelligence* (Vol. 22, No. 3, p. 1962).

Huang, F., & Yates, A. (2009). Distributional representations for handling sparsity in supervised sequence-labeling. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 1-Volume 1* (pp. 495-503). Association for Computational Linguistics.

Huang, F., & Yates, A. (2010). Exploring representation-learning approaches to domain adaptation. In *Proceedings of the 2010 Workshop on Domain Adaptation for Natural Language Processing* (pp. 23-30). Association for Computational Linguistics.

Huang, J., Gretton, A., Borgwardt, K. M., Schölkopf, B., & Smola, A. J. (2006). Correcting sample selection bias by unlabeled data. In *Advances in neural information processing systems* (pp. 601-608).

Hung, S. L., & Adeli, H. (1994). A parallel genetic/neural network learning algorithm for MIMD shared memory machines. *Neural Networks, IEEE Transactions on*, *5*(6), 900-909.

Jacobs, R. A., Jordan, M. I., Nowlan, S. J., & Hinton, G. E. (1991). Adaptive mixtures of local experts. *Neural computation*, *3*(1), 79-87.

Jaffee, S. R., Caspi, A., Moffitt, T. E., Dodge, K. A., Rutter, M., Taylor, A., & Tully, L. A. (2005). Nature× nurture: Genetic vulnerabilities interact with physical maltreatment to promote conduct problems. *Development and psychopathology*, *17*(1), 67-84.

Jansen, T., De Jong, K. A., & Wegener, I. (2005). On the choice of the offspring population size in evolutionary algorithms. *Evolutionary Computation*, *13*(4), 413-440.

Jiang, J., & Zhai, C. (2007a). Instance weighting for domain adaptation in NLP. In *ACL* (Vol. 7, pp. 264-271).

Jiang, J., & Zhai, C. (2007b). A two-stage approach to domain adaptation for statistical classifiers. In *Proceedings of the sixteenth ACM conference on Conference on information and knowledge management* (pp. 401-410). ACM.

Jin, Y., & Sendhoff, B. (2006). Alleviating catastrophic forgetting via multi-objective learning. In *The 2006 IEEE International Joint Conference on Neural Network Proceedings* (pp. 3335-3342). IEEE.

Karaminis, T., Thomas, M. S., & Karaminis, T. (2015). The Multiple Inflection Generator: A generalized connectionist model for cross-linguistic morphological development. *Manuscript submitted for publication*.

Karaminis,T., Thomas, M.S.C. (2010): A cross-linguistic model of the acquisition of inflectional morphology in English and modern Greek. *Proceedings of the 32ndAnnual Meeting of the Cognitive Science Society* (pp. 730-735).

Karmiloff-Smith, A. & Thomas, M. S. C. (2003). What can developmental disorders tell us about the neurocomputational constraints that shape development? The case of Williams syndrome. *Development and Psychopathology* 15, (pp. 969-990).

Karmiloff-Smith, A. (1998). Development itself is the key to understanding developmental disorders. *Trends in cognitive sciences*, *2*(10), 389-398.

Kendler, K. S. (1996). Major depression and generalised anxiety disorder same genes, (Partly) different environments—Revisited. *The British Journal of Psychiatry*.

Khadka, S., Tumer, K., Colby, M., Tucker, D., Pezzini, P., & Bryden, K. (2016). Neuroevolution of a Hybrid Power Plant Simulator. In *Proceedings of the 2016 on Genetic and Evolutionary Computation Conference* (pp. 917-924). ACM

Khashei, M., Hamadani, A. Z., & Bijari, M. (2012). A novel hybrid classification model of artificial neural networks and multiple linear regression models. *Expert Systems with Applications*, *39*(3), 2606-2620.

Kim, H. B., Jung, S. H., Kim, T. G., & Park, K. H. (1996). Fast learning method for back-propagation neural network by evolutionary adaptation of learning rates. *Neurocomputing*, *11*(1), 101-106.

Kinnebrock, W. (1994). Accelerating the standard backpropagation method using a genetic approach. *Neurocomputing*, *6*(5), 583-588.

Kitano, H. (1990). Designing neural networks using genetic algorithms with graph generation system. *Complex Systems Journal*, *4*, 461-476.

Kohli, M., Magoulas, G. D., & Thomas, M. S. (2013). Transfer learning across heterogeneous tasks using behavioural genetic principles. In *Computational Intelligence (UKCI), 2013 13th UK Workshop on* (pp. 151-158). IEEE.

Kononenko, I. (1993). Inductive and Bayesian learning in medical diagnosis.*Applied Artificial Intelligence an International Journal*, *7*(4), 317-337.

Kovas, Y., & Plomin, R. (2006). Generalist genes: implications for the cognitive sciences. *Trends in cognitive sciences*, *10*(5), 198-203.

Kovas, Y., & Plomin, R. (2007). Learning abilities and disabilities generalist genes, specialist environments. *Current Directions in Psychological Science*, *16*(5), 284-288.

Krasnogor, N., & Smith, J. (2005). A tutorial for competent memetic algorithms: model, taxonomy, and design issues. *Evolutionary Computation, IEEE Transactions on*, *9*(5), 474-488.

Kusumoputro, B. (2016). Infrared Face Recognition System Using Cross Entropy Error Function Based Ensemble Backpropagation Neural Networks. *International Journal of Computer Theory and Engineering*, *8*(2), 161.

Langacker, R. W. (1987). *Foundations of cognitive grammar: Theoretical prerequisites* (Vol. 1). Stanford university press.

Lawrence, N. D., & Platt, J. C. (2004). Learning to learn with the informative vector machine. In *Proceedings of the twenty-first international conference on Machine learning* (p. 65). ACM.

Lee, S. W. (1996). Off-line recognition of totally unconstrained handwritten numerals using multilayer cluster neural network. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, *18*(6), 648-652.

Lehman, J. & Miikkulainen, R. (2013) Neuroevolution. Scholarpedia, 8(6):30977.

Lehman, J., & Miikkulainen, R. (2014). Overcoming deception in evolution of cognitive behaviors. In *Proceedings of the 2014 Annual Conference on Genetic and Evolutionary Computation* (GECCO '14). ACM, New York, NY, USA, 185-192. DOI=http://dx.doi.org/10.1145/2576768.2598300

Lewandowsky, S., & Li, S. C. (1995). Catastrophic interference in neural networks: Causes, solutions, and data.

Li, X., & Wu, X. (2015). Constructing long short-term memory based deep recurrent neural networks for large vocabulary speech recognition. In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (pp. 4520-4524). IEEE.

Liao, X., Xue, Y., & Carin, L. (2005). Logistic regression with an auxiliary data source. In *Proceedings of the 22nd international conference on Machine learning* (pp. 505-512). ACM.

Likartsis, A., Vlachavas, I., & Tsoukalas, L. H. (1997). A new hybrid neural-genetic methodology for improving learning. In *ictai* (p. 0032). IEEE.

Lipowski, A., & Lipowska, D. (2012). Roulette-wheel selection via stochastic acceptance. *Physica A: Statistical Mechanics and its Applications*, *391*(6), 2193-2196.

Liu, W., Zhang, H., & Li, J. (2009). Inductive transfer through neural network error and dataset regrouping. In *Intelligent Computing and Intelligent Systems, 2009. ICIS 2009. IEEE International Conference on* (Vol. 1, pp. 777-781). IEEE.

Liu, Y., & Yao, X. (1999). Ensemble learning via negative correlation. *Neural Networks*, *12*(10), 1399-1404.

Lu, J., Behbood, V., Hao, P., Zuo, H., Xue, S., & Zhang, G. (2015). Transfer learning using computational intelligence: A survey. *Knowledge-Based Systems*, *80*, 14-23.

Luis, R., Sucar, L. E., & Morales, E. F. (2010). Inductive transfer for learning Bayesian networks. *Machine learning*, *79*(1-2), 227-255.

Lupyan, G., & McClelland, J. L. (2003). Did, made, had, said: Capturing quasi-regularity in exceptions. In *Proceedings of the Annual Meeting of the Cognitive Science Society* (Vol. 25, pp. 740-745).

MacWhinney, B. (1998). Models of the emergence of language. *Annual review of Psychology 49* (pp. 199-227).

MacWhinney, B. (2008). A unified model of language acquisition. *Handbook of Bilingualism: Psycholinguistic Approaches, eds. JF Kroll and AMB De Groot (Oxford, 2005)*, 49-67.

Majerus, M. E. (2009). Industrial melanism in the peppered moth, Biston betularia: an excellent teaching example of Darwinian evolution in action. *Evolution: Education and Outreach*, *2*(1), 63-74.

Maniezzo, V. (1994). Genetic evolution of the topology and weight distribution of neural networks. *IEEE Transactions on neural networks*, *5*(1), 39-53.

Mareschal, D., & Thomas, M. S. (2007). Computational modeling in developmental psychology. *IEEE Transactions on Evolutionary Computation*,*11*(2), 137-150.

Marin, F. J., & Sandoval, F. (1993). Genetic synthesis of discrete-time recurrent neural network. In *New Trends in Neural Computation* (pp. 179-184). Springer Berlin Heidelberg.

Martins, M.S.R., Delgado, M.R.B.S., Santana, R., Lüders, R., Gonçalves, R.A., & de Almeida, C.P. (2016). HMOBEDA: Hybrid Multi-objective Bayesian Estimation of Distribution Algorithm. In *Proceedings of the Genetic and Evolutionary Computation Conference 2016* (GECCO '16), Tobias Friedrich (Ed.). ACM, New York, NY, USA, 357-364. DOI: http://dx.doi.org/10.1145/2908812.2908826

Mayley, G. (1996). Landscapes, learning costs, and genetic assimilation.*Evolutionary Computation*, *4*(3), 213-234.

Mehta, R., Panageas, I., Piliouras, G., Tetali, P., & Vazirani, V. V. (2015). Mutation, sexual reproduction and survival in dynamic environments. *arXiv preprint arXiv:1511.01409*.

Meltzoff, A. N., & Moore, M. K. (1999). Persons and representation: Why infant imitation is important for theories of human development.

Mihalkova, L., Huynh, T., & Mooney, R. J. (2007). Mapping and revising Markov logic networks for transfer learning. In *AAAI* (Vol. 7, pp. 608-614).

Miikkulainen, R. (2015). Evolving neural networks. In *Proceedings of the Companion Publication of the 2015 Annual Conference on Genetic and Evolutionary Computation* (pp. 137-161). ACM.

Montana, D. J., & Davis, L. (1989). Training Feedforward Neural Networks Using Genetic Algorithms. In *IJCAI* (Vol. 89, pp. 762-767).

Morgan, C. L. (1896). On modification and variation. *Science*, 733-740.

Moriarty, D. E., & Miikkulainen, R. (2016). Efficient learning from delayed rewards through symbiotic evolution. In *Proceedings 12th International Conference on Machine Learning* (pp. 396-404).

Mosca, A., & Magoulas, G. D. (2016). Regularizing Deep Learning Ensembles by Distillation. In *6thInternational Workshop on Combinations of Intelligent Methods and Applications (CIMA 2016)* (p. 53).

Mouret, J. B., & Tonelli, P. (2014). Artificial evolution of plastic neural networks: a few key concepts. In *Growing Adaptive Machines* (pp. 251-261). Springer Berlin Heidelberg.

Mühlenbein, H., & Kindermann, J. (1989). The dynamics of evolution and learning—towards genetic neural networks. *Connectionism in*, *9*, 173-198.

Munroe, S., & Cangelosi, A. (2002). Learning and the evolution of language: the role of cultural variation and learning costs in the Baldwin effect. *Artificial Life*, *8*(4), 311-339.

Murru, N., & Rossini, R. (2016). A Bayesian approach for initialization of weights in backpropagation neural net with application to character recognition. *Neurocomputing*, *193*, 92-105.

Nickolls, J., & Dally, W. J. (2010). The GPU computing era. *Micro, IEEE*, *30*(2), 56-69.

Niculescu-Mizil, A., & Caruana, R. (2007). Inductive Transfer for Bayesian Network Structure Learning. In *AISTATS* (pp. 339-346).

Nielsen, S. S., Torres, C. F., Danoy, G., and Bouvry, P. (2016). Tackling the IFP Problem with the Preference-Based Genetic Algorithm. In *Proceedings of the Genetic and Evolutionary Computation Conference 2016* (GECCO '16), Tobias Friedrich (Ed.). ACM, New York, NY, USA, 965-972. DOI: http://dx.doi.org/10.1145/2908812.2908954

Nishida, H. (1997). Cell lineage and timing of fate restriction, determination and gene expression in ascidian embryos. In *Seminars in cell & developmental biology* (Vol. 8, No. 4, pp. 359-365). Academic Press.

Nolfi, S., & Floreano, D. (1999). Learning and evolution. *Autonomous robots*, *7*(1), 89-113.

Nolfi, S., & Parisi, D. (2002). Evolution of artificial neural networks. In *In Handbook of brain theory and neural networks*.

Nolfi, S., Parisi, D., & Elman, J. L. (1994). Learning and evolution in neural networks. *Adaptive Behavior*, *3*(1), 5-28.

Nvidia, C. U. D. A. (2008). Programming guide.

Pan, S. J., & Yang, Q. (2010). A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, *22*(10), 1345-1359

Pan, S. J., Kwok, J. T., Yang, Q., & Pan, J. J. (2007). Adaptive Localization in a Dynamic WiFi Environment through Multi-view Learning. In *Proceedings of the national conference on artificial Intelligence* (Vol. 22, No. 2, p. 1108). Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999.

Pan, S. J., Ni, X., Sun, J. T., Yang, Q., & Chen, Z. (2010). Cross-domain sentiment classification via spectral feature alignment. In *Proceedings of the 19th international conference on World wide web* (pp. 751-760). ACM.

Paredis, J. (1991). The evolution of behavior: some experiments. In *Proceedings of the first international conference on simulation of adaptive behavior on From animals to animats* (pp. 419-426). MIT Press.

Pavlidis, N. G., Tasoulis, O. K., Plagianakos, V. P., Nikiforidis, G., & Vrahatis, M. N. (2005). Spiking neural network training using evolutionary algorithms. In *Proceedings. 2005 IEEE International Joint Conference on Neural Networks, 2005.* (Vol. 4, pp. 2190-2194). IEEE.

Pearson, H. (2006). Genetics: what is a gene?. *Nature*, *441*(7092), 398-401.

Pinker, S. (1984): *Language learnability and language development*. Cambridge, MA: Harvard University Press.

Pinker, S. (1994). *The language instinct: The new science of language and mind* (Vol. 7529). Penguin UK.

Plagianakos, V. P., Magoulas, G. D., & Vrahatis, M. N. (2006a). Distributed computing methodology for training neural networks in an image-guided diagnostic application. *Computer methods and programs in biomedicine*,*81*(3), 228-235.

Plagianakos, V. P., Magoulas, G. D., & Vrahatis, M. N. (2006b). Evolutionary training of hardware realizable multilayer perceptrons. *Neural Computing & Applications*, *15*(1), 33-40.

Plaut, D.C., McClelland, J.L., Seidenberg, M.S., Patterson, K. (1996): Understanding Normal and Impaired Word Reading: Computational principles in Quasi-Regular Domains. *Psychological Review, 103* (pp.56-115).

Plomin, R. & DeFries, J.C. (1980). Genetics and Intelligence: recent Data. *Intelligence* (4). (pp.15-24).

Plomin, R. (1990). The role of inheritance in behaviour. *Science*, 248 (pp.183-248).

Plomin, R., & Deary, I. J. (2015). Genetics and intelligence differences: five special findings. *Molecular psychiatry*, *20*(1), 98-108.

Plomin, R., & Kovas, Y. (2005). Generalist genes and learning disabilities.*Psychological bulletin*, *131*(4), 592.

Plomin, R., & Spinath, F. M. (2004). Intelligence: genetics, genes, and genomics. *Journal of personality and social psychology*, *86*(1), 112.

Plomin, R., DeFries, J. C., & McClearn, G. E. (2008): *Behavioral Genetics: A primer*. San Francisco: Worth and Freeman.

Plomin, R., DeFries, J. C., Knopik, V. S., & Neiderheiser, J. (2013). *Behavioral genetics*. Palgrave Macmillan.

Plomin, R., Kovas, Y., & Haworth, C. (2007). Generalist genes: Genetic links between brain, mind, and education. *Mind, Brain, and Education*, *1*(1), 11-19.

Plunkett, K., & Marchman, V. (1991). U-shaped learning and frequency effects in a multi-layered perception: Implications for child language acquisition. *Cognition*, *38*(1), 43-102.

Plunkett, K., & Marchman, V. (1993): From rote learning to system building: Acquiring verb morphology in children and connectionist nets. *Cognition*, 48 (pp.21-69).

Porto, V. W., Fogel, D. B., & Fogel, L. J. (1995). Alternative neural network training methods. *IEEE Intelligent Systems*, (3), 16-22.

Pratt, L. Y. (1992). Non-literal transfer among neural network learners. *Colorado School of Mines: MCS-92-04*.

Pratt, L., & Jennings, B. (1996). A survey of transfer between connectionist networks. *Connection Science*, *8*(2), 163-184.

Qiao, J., Li, S., & Li, W. (2016). Mutual information based weight initialization method for sigmoidal feedforward neural networks. *Neurocomputing*.

Quattoni, A., Collins, M., & Darrell, T. (2008). Transfer learning for image classification with sparse prototype representations. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on* (pp. 1-8). IEEE.

Quinlan, J. R. (1996). Bagging, boosting, and C4. 5. In *AAAI/IAAI, Vol. 1* (pp. 725-730).

Reisinger, J., Bahçeci, E., Karpov, I., & Miikkulainen, R. (2007). Coevolving strategies for general game playing. In *Computational Intelligence and Games, 2007. CIG 2007. IEEE Symposium on* (pp. 320-327). IEEE.

Reynolds, R. G., & Peng, B. (2004). Cultural algorithms: modeling of how cultures learn to solve problems. In *Tools with Artificial Intelligence, 2004. ICTAI 2004. 16th IEEE International Conference on* (pp. 166-172). IEEE.

Riedmiller, M., & Braun, H. (1993). A direct adaptive method for faster backpropagation learning: The RPROP algorithm. In *Neural Networks, 1993., IEEE International Conference On* (pp. 586-591). IEEE.

Risi, S., & Stanley, K. O. (2010a). Indirectly encoding neural plasticity as a pattern of local rules. In *International Conference on Simulation of Adaptive Behavior* (pp. 533-543). Springer Berlin Heidelberg.

Risi, S., Hughes, C. E., & Stanley, K. O. (2010b). Evolving plastic neural networks with novelty search. *Adaptive Behavior*, *18*(6), 470-491.

Risi, S., Togelius, J. (2015). Neuroevolution in Games: State of the Art and Open Challenges. In *IEEE Transactions on Computational Intelligence and AI in Games*, vol.PP, no.99, pp.1-1 doi: 10.1109/TCIAIG.2015.2494596

Robins, A. (1995). Catastrophic forgetting, rehearsal and pseudorehearsal. *Connection Science*, *7*(2), 123-146.

Romera-Paredes, B., Argyriou, A., Berthouze, N., & Pontil, M. (2012). Exploiting unrelated tasks in multi-task learning. In *International Conference on Artificial Intelligence and Statistics* (pp. 951-959).

Rosenstein, M. T., Marx, Z., Kaelbling, L. P., & Dietterich, T. G. (2005). To transfer or not to transfer. In *NIPS 2005 Workshop on Inductive Transfer: 10 Years Later* (Vol. 2, p. 7).

Rückert, U., & Kramer, S. (2008). Kernel-based inductive transfer. In *Machine Learning and Knowledge Discovery in Databases* (pp. 220-233). Springer Berlin Heidelberg.

Rumelhart, D. E., & McClelland, J. L. (1986): *On learning the past tense of English verbs*. In J. L. McClelland, D. E. Rumelhart, & the PDP Research Group. (Eds.), Parallel distributed processing: Explorations in the microstructure of cognition: Vol. 2. Psychological and biological models (pp.216-271). Cambridge, MA: MIT Press.

Sadikovic, B., Al-Romaih, K., Squire, J. A., & Zielenska, M. (2008). Cause and consequences of genetic and epigenetic alterations in human cancer. *Current genomics*, *9*(6), 394-408.

Sasaki, T., & Tokoro, M. (1997). Adaptation toward changing environments: Why darwinian in nature. In *Fourth European conference on artificial life* (pp. 145-153). MIT Press.

Sastry, K., Goldberg, D. E., & Kendall, G. (2014). Genetic algorithms. In *Search methodologies* (pp. 93-117). Springer US.

Satpal, S., & Sarawagi, S. (2007). Domain adaptation of conditional probability models via feature subsetting. In *Knowledge Discovery in Databases: PKDD 2007* (pp. 224-235). Springer Berlin Heidelberg.

Schaffer, J. D., Caruana, R. A., & Eshelman, L. J. (1990). Using genetic search to exploit the emergent behavior of neural networks. *Physica D: Nonlinear Phenomena*, *42*(1), 244-248.

Schmidhuber, J. (2015). Deep learning in neural networks: An overview. *Neural Networks*, *61*, 85-117

Schrum, J., & Miikkulainen, R. (2014). Evolving multimodal behavior with modular neural networks in Ms. Pac-Man. In *Proceedings of the 2014 Annual Conference on Genetic and Evolutionary Computation* (pp. 325-332). ACM.

Sebastiani, F. (2002). Machine learning in automated text categorization. *ACM computing surveys (CSUR)*, *34*(1), 1-47.

Seipone, T., & Bullinaria, J. A. (2005). Evolving improved incremental learning schemes for neural network systems. In *Congress on Evolutionary Computation* (pp. 2002-2009).

Sharma, M., Holmes, M. P., Santamaría, J. C., Irani, A., Isbell Jr, C. L., & Ram, A. (2007). Transfer Learning in Real-Time Strategy Games Using Hybrid CBR/RL. In *IJCAI* (Vol. 7, pp. 1041-1046).

Shavlik, J. W., Mooney, R. J., & Towell, G. G. (1991). Symbolic and neural learning algorithms: An experimental comparison. *Machine learning*, *6*(2), 111-143.

Shell, J., & Coupland, S. (2012). Towards fuzzy transfer learning for intelligent environments. In *Ambient Intelligence* (pp. 145-160). Springer Berlin Heidelberg.

Shell, J., & Coupland, S. (2015). Fuzzy transfer learning: methodology and application. *Information Sciences*, *293*, 59-79.

Shell, J., Vickers, S., Coupland, S., & Istance, H. (2012). Towards dynamic accessibility through soft gaze gesture recognition. In *Computational Intelligence (UKCI), 2012 12th UK Workshop on* (pp. 1-8). IEEE.

Shi, X., Liu, Q., Fan, W., Yu, P. S., & Zhu, R. (2010). Transfer learning on heterogenous feature spaces via spectral transformation. In *Data Mining (ICDM), 2010 IEEE 10th International Conference on* (pp. 1049-1054). IEEE.

Silver, D. L. (1996). The parallel transfer of task knowledge using dynamic learning rates based on a measure of relatedness. *Connection Science*, *8*(2), 277-294.

Silver, D., & Mercer, R. (2001). Selective functional transfer: Inductive bias from related tasks. In *IASTED International Conference on Artificial Intelligence and Soft Computing (ASC2001)* (pp. 182-189).

Simon, H. A. (1962). The architecture of complexity. *Proceedings of the American philosophical society*, *106*(6), 467-482.

Smith, J. M. (1986). When learning guides evolution. *Nature*, *329*(6142), 761-762.
Society, 106 (pp.467-482). (Reprinted in Simon, H.A. (1969). The sciences of the artificial. Cambridge, MA: MIT Press).

Srinivas, M., & Patnaik, L. M. (1991). Learning neural network weights using genetic algorithms-improving performance by search-space reduction. In *Neural Networks, 1991. 1991 IEEE International Joint Conference on* (pp. 2331-2336). IEEE.

Stanley, K. O., & Miikkulainen, R. (2002). Evolving neural networks through augmenting topologies. *Evolutionary computation*, *10*(2), 99-127.

Stanley, K. O., & Miikkulainen, R. (2004). Competitive coevolution through evolutionary complexification. *J. Artif. Intell. Res.(JAIR)*, *21*, 63-100.

Stanley, K. O., Bryant, B. D., & Miikkulainen, R. (2003). Evolving adaptive neural networks with and without adaptive synapses. In *Evolutionary Computation, 2003. CEC'03. The 2003 Congress on* (Vol. 4, pp. 2557-2564). IEEE

Subiaul, F., Cantlon, J., Holloway, R. L., Terrace, H. S. (2004). Cognitive Imitation in Rhesus Macaques. Science, 305 (5682, pp. 407–410)

Sugiyama, M., Nakajima, S., Kashima, H., Buenau, P. V., & Kawanabe, M. (2008). Direct importance estimation with model selection and its application to covariate shift adaptation. In *Advances in neural information processing systems* (pp. 1433-1440).

Thain, D., Tannenbaum, T., & Livny, M. (2005). Distributed computing in practice: the Condor experience. *Concurrency and computation: practice and experience*, *17*(2-4), 323-356.

Thimm, G., Fiesler, E. (1995): Neural network initialization. From Natural to Artificial Neural Computation. *Lecture Notes in Computer Science Volume 930* (pp.535-542).

Thomas, M. S. (2016). Do more intelligent brains retain heightened plasticity for longer in development? A computational investigation. *Developmental cognitive neuroscience*, *19*, 258-269.

Thomas, M. S. (2005). Constraints on language development. *Developmental theory and language disorders*, *4*, 11.

Thomas, M. S. C., Annaz, D., Ansari, D., Serif, G., Jarrold, C., & Karmiloff-Smith, A. (2009a). Using developmental trajectories to understand developmental disorders. *Journal of Speech, Language, and Hearing Research*, *52*, 336-358.

Thomas, M. S. C., Ronald, A., & Forrester, N. A. (2009b): Modelling the mechanisms underlying population variability across development: Simulating genetic and environmental effects on cognition. *DNL Tech report 2009-1*.

Thomas, M. S., & Karmiloff-Smith, A. (2003). Modeling language acquisition in atypical phenotypes. *Psychological review*, *110*(4), 647.

Thomas, M. S., & Knowland, V. C. (2014). Modeling mechanisms of persisting and resolving delay in language development. *Journal of Speech, Language, and Hearing Research*, *57*(2), 467-483.

Thomas, M. S., Forrester, N. A., & Ronald, A. (2013). Modeling socioeconomic status effects on language development. *Developmental Psychology*, *49*(12), 2325.

Thomas, M. S., Forrester, N. A., & Ronald, A. (2016). Multiscale modeling of gene–behavior associations in an artificial neural network model of cognitive development. *Cognitive science*, *40*(1), 51-99.

Thomas, M.S.C., McClelland, J.L. (2008): Connectionist models of cognition. In R. Sun (Ed.), *Cambridge handbook of computational cognitive modelling* (pp.23-58). Cambridge University Press.

Thorndike, E. L. and Woodworth, R. S. (1901) "The influence of improvement in one mental function upon the efficiency of other functions", *Psychological Review* 8: Part I, pp. 247–261 doi:10.1037/h0074898; Part II, pp. 384–395 doi:10.1037/h0071280; Part III, pp. 553–564 doi:10.1037/h0071363

Thrun, S. B., Bala, J., Bloedorn, E., Bratko, I., Cestnik, B., Cheng, J., ... & Zhang, J. (1991). The monk's problems a performance comparison of different learning algorithms.

Thrun, S., & Pratt, L. (1998). Learning to learn: Introduction and overview. In *Learning to learn* (pp. 3-17). Springer US.

Tonelli, P., & Mouret, J. B. (2013). On the relationships between generative encodings, regularity, and learning abilities when evolving plastic artificial neural networks. *PloS one*, *8*(11), e79138.

Torrey, L., & Shavlik, J. (2009). Transfer learning. *Handbook of Research on Machine Learning Applications and Trends: Algorithms, Methods, and Techniques*, *1*, 242.

Torrey, L., Shavlik, J., Walker, T., & Maclin, R. (2006). Relational skill transfer via advice taking. In *ICML Workshop on Structural Knowledge Transfer for Machine Learning*.

Torrey, L., Walker, T., Shavlik, J., & Maclin, R. (2005). Using advice to transfer knowledge acquired in one reinforcement learning task to another. In *Machine Learning: ECML 2005* (pp. 412-424). Springer Berlin Heidelberg.

Trianni, V., Ampatzis, C., Christensen, A. L., Tuci, E., Dorigo, M., & Nolfi, S. (2007). From solitary to collective behaviours: Decision making and cooperation. In *Advances in Artificial Life* (pp. 575-584). Springer Berlin Heidelberg.

Tsuboi, Y., Kashima, H., Hido, S., Bickel, S., & Sugiyama, M. (2009). Direct Density Ratio Estimation for Large-scale Covariate Shift Adaptation. *Information and Media Technologies*, *4*(2), 529-546.

Turner, A. P., Caves, L. S., Stepney, S., Tyrrell, A. M., & Lones, M. A. (2016). Artificial epigenetic networks: automatic decomposition of dynamical control tasks using topological self-modification. *IEEE transactions on neural networks and learning systems*, *28*(1), 218-230.

Turner, A. P., Trefzer, M. A., Lones, M. A., & Tyrrell, A. M. (2015). Evolving Efficient Solutions to Complex Problems Using the Artificial Epigenetic Network. In *International Conference on Information Processing in Cells and Tissues* (pp. 153-165). Springer International Publishing.

Ueki, K., Sugiyama, M., & Ihara, Y. (2010). Perceived age estimation under lighting condition change by covariate shift adaptation. In *Pattern Recognition (ICPR), 2010 20th International Conference on* (pp. 3400-3403). IEEE.

van der Lely, H. K. J., & Ullman, M. (2001). Past tense morphology in specifically language impaired children and normally developing children. *Language and Cognitive Processes*, *16*, 177-217.

Waddington, C. H. (1942). Canalization of development and the inheritance of acquired characters. *Nature*, *150*(3811), 563-565.

Waddington, C. H. (1957). The strategy of the genes. A discussion of some aspects of theoretical biology. With an appendix by H. Kacser. *The strategy of the genes. A discussion of some aspects of theoretical biology. With an appendix by H. Kacser.*, ix+-262.

Wang, C., & Mahadevan, S. (2011). Heterogeneous domain adaptation using manifold alignment. In *IJCAI Proceedings-International Joint Conference on Artificial Intelligence* (Vol. 22, No. 1, p. 1541).

Wei, B., & Pal, C. J. (2011). Heterogeneous Transfer Learning with RBMs. In *AAAI*.

Whitley, D., Starkweather, T., & Bogart, C. (1990). Genetic algorithms and neural networks: Optimizing connections and connectivity. *Parallel computing*, *14*(3), 347-361.

Wintner, S. (2010). Computational models of language acquisition. In *Computational Linguistics and Intelligent Text Processing* (pp.86-99). Springer Berlin Heidelberg.

Xu, X., Tang, Y., Li, J., Hua, C., & Guan, X. (2015). Dynamic multi-swarm particle swarm optimizer with cooperative learning strategy. *Applied Soft Computing*, *29*, 169-183

Xue, S., Lu, J., Zhang, G., & Xiong, L. (2015). Heterogeneous Feature Space Based Task Selection Machine for Unsupervised Transfer Learning. In *Intelligent Systems and Knowledge Engineering (ISKE), 2015 10th International Conference on* (pp. 46-51). IEEE.

Yam, J. Y., & Chow, T. W. (2000). A weight initialization method for improving training speed in feedforward neural network. *Neurocomputing*, *30*(1), 219-232.

Yan, W., Zhu, Z., & Hu, R. (1997). A hybrid genetic/BP algorithm and its application for radar target classification. In *Aerospace and Electronics Conference, 1997. NAECON 1997., Proceedings of the IEEE 1997 National*(Vol. 2, pp. 981-984). IEEE.

Yang, J., Lee, S. H., Goddard, M. E., & Visscher, P. M. (2011). GCTA: a tool for genome-wide complex trait analysis. *The American Journal of Human Genetics*, *88*(1), 76-82.

Yao, X. (1999). Evolving artificial neural networks. *Proceedings of the IEEE*, *87*(9), 1423-1447.

Yao, X., & Islam, M. M. (2008). Evolving artificial neural network ensembles. *IEEE Computational Intelligence Magazine*, *3*(1), 31-42.

Yao, X., & Liu, Y. (1997). A new evolutionary system for evolving artificial neural networks. *Neural Networks, IEEE Transactions on*, *8*(3), 694-713.

Yao, X., & Shi, Y. (1995). A preliminary study on designing artificial neural networks using co-evolution. In *Proceedings of the IEEE Singapore International Conference on Intelligent Control and Instrumentation* (pp. 149-154).

Yao, X., & Xu, Y. (2006). Recent advances in evolutionary computation. *Journal of Computer Science and Technology*, *21*(1), 1-18.

Yin, J., Yang, Q., & Ni, L. (2005). Adaptive temporal radio maps for indoor location estimation. In *Pervasive Computing and Communications, 2005. PerCom 2005. Third IEEE International Conference on* (pp. 85-94). IEEE.

Yuksel, S. E., Wilson, J. N., & Gader, P. D. (2012). Twenty years of mixture of experts. *IEEE transactions on neural networks and learning systems*, *23*(8), 1177-1193.

Zeng, H., Edwards, M. D., Liu, G., & Gifford, D. K. (2016). Convolutional neural network architectures for predicting DNA–protein binding. *Bioinformatics*, *32*(12), i121-i127.

Zhang, H., Wang, Z., & Liu, D. (2014). A comprehensive review of stability analysis of continuous-time recurrent neural networks. *IEEE Transactions on Neural Networks and Learning Systems*, *25*(7), 1229-1262.

Zhao, Z. S., Feng, X., Lin, Y. Y., Wei, F., Wang, S. K., Xiao, T. L., ... & Hou, Z. G. (2015). Evolved neural network ensemble by multiple heterogeneous swarm intelligence. *Neurocomputing*, *149*, 29-38.

Zheng, V. W., Xiang, E. W., Yang, Q., & Shen, D. (2008). Transferring Localization Models over Time. In *AAAI* (pp. 1421-1426).

Zhou, J. T., Tsang, I. W., Pan, S. J., & Tan, M. (2014a). Heterogeneous Domain Adaptation for Multiple Classes. In *AISTATS* (pp. 1095-1103).

Zhou, J. T., Pan, S. J., Tsang, I. W., & Yan, Y. (2014b). Hybrid heterogeneous transfer learning through deep learning. In *Twenty-Eighth AAAI Conference on Artificial Intelligence*.

Zhu, Y., Chen, Y., Lu, Z., Pan, S. J., Xue, G. R., Yu, Y., & Yang, Q. (2011). Heterogeneous Transfer Learning for Image Classification. In *AAAI*.

Publications

Kohli, M., Magoulas, G. D., & Thomas, M. S. C. (2017). Capturing Population Variability in Children's Past tense Formation by Evolving Populations of Neural Networks. *Manuscript submitted*

Kohli, M., Magoulas, G. D., & Thomas, M. S. C. (2013). Transfer learning across heterogeneous tasks using behavioural genetic principles. In Y. Jin (Ed.), *Proceedings of 2013 UK Workshop on Computational Intelligence.*

Kohli, M., Magoulas, G. D., & Thomas, M. S. C. (2012). Hybrid computational model for producing English past tense verbs. In *Proceedings of 13th Engineering Applications of Neural Network Conference (EANN2012)*, London, UK, from September 20-23, 2012.

Presented Poster Titled " Hybrid GA Framework Based on Behavioural Genetics" in Workshop on Engineering Applications of Neural Networks on 25th Nov'2010 at Faculty of Computing, London Metropolitan University, UK