How to Count Thumb-Ups and Thumb-Downs?

An Information Retrieval Approach to User-Rating based Ranking of Items

Dell Zhang
Birkbeck, University of London
Malet Street
London WC1E 7HX, UK
dell.z@ieee.org

Robert Mao
Microsoft Research
1 Microsoft Way
Redmond, WA 98052, USA
robmao@microsoft.com

Haitao Li
Microsoft Corporation
1 Microsoft Way
Redmond, WA 98052, USA
lht1999@gmail.com

Joanne Mao
Hughes Network Systems
11717 Exploration Lane
Germantown, MD 20876, USA
zhijuan@gmail.com

ABSTRACT

It is a common practice among Web 2.0 services to allow users to rate items on their sites. In this paper, we first point out the flaws of the popular methods for user-rating based ranking of items, and then argue that two well-known Information Retrieval (IR) techniques, namely the Probability Ranking Principle and Statistical Language Modelling, provide a simple but effective solution to this problem.

Categories and Subject Descriptors:
H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval — *retrieval models*

General Terms:
Human Factors, Measurement, Theory.

Keywords:
Web 2.0, Information Retrieval, Probability Ranking Principle, Statistical Language Modelling, Smoothing.

1. PROBLEM

Suppose that you are building a Web 2.0 service which allows users to rate items (such as commercial-products, photos, videos, songs, news-reports, and answers-to-questions) on your site, you probably want to sort items according to their user-ratings so that stuff "liked" by users would be ranked higher than those "disliked". How should you do that? What is the best way to count such thumb-ups and thumb-downs?

Let's focus on binary rating systems first and then generalise to graded rating systems later. Given an item i, let $n_\uparrow(i)$ denote the number of thumb-ups and $n_\downarrow(i)$ denote the number of thumb-downs. To sort the relevant items based on user-ratings, a score function $s(n_\uparrow, n_\downarrow) \in \mathbb{R}$ would need to be calculated for each of them.

2. POPULAR METHODS

There are currently three popular methods widely used in practice for this problem, each of which has some flaws.

The first method is to use the *difference* between the number of thumb-ups and the number of thumb-downs as the score function, i.e.,

$$s(n_\uparrow, n_\downarrow) = n_\uparrow - n_\downarrow .$$

For example, Urban Dictionary, a web-based dictionary of slang words and phrases, uses this method. Assume that item i has 200 thumb-ups and 100 thumb-downs, while item j has 1,200 thumb-ups and 1,000 thumb-downs, this method would rank item i (whose score is 100) lower than item j (whose score is 200). However, this does not sound right, because item i has twice thumb-ups than thumb-downs, while item j has only slightly more thumb-ups than thumb-downs.

The second method is to use the *proportion* of thumb-ups in all user-ratings as the score function, i.e.,

$$s(n_\uparrow, n_\downarrow) = \frac{n_\uparrow}{n_\uparrow + n_\downarrow} .$$

For example, Amazon, the largest online retailer company in the United States, uses this method. Assume that item i has 200 thumb-ups and 1 thumb-down, while item j has 2 thumb-ups and 0 thumb-down, this method would rank item i (whose score is 0.995) lower than item j (whose score is 1.000). However, this does not sound right, because although both item i and item j have almost none thumb-down, item i has hundreds of thumb-ups, while item j has only a couple of thumb-ups.

The third method was advocated by Evan Miller's online article[1] on this topic to avoid the flaws of the above two simple methods. The idea is to treat the existing set of user-ratings as a statistical sampling of a hypothetical set of user-ratings from all users, and then use the *lower bound* of *Wilson score confidence interval* [3] for the proportion of thumb-ups as the score function, i.e.,

$$s(n_\uparrow, n_\downarrow) = \frac{\hat{p} + \frac{z^2_{1-\alpha/2}}{2n} - \sqrt{\frac{z^2_{1-\alpha/2}}{n}\left[\hat{p}(1-\hat{p}) + \frac{z^2_{1-\alpha/2}}{4n}\right]}}{1 + \frac{z^2_{1-\alpha/2}}{n}},$$

where $n = n_\uparrow + n_\downarrow$ is the total number of user-ratings, $\hat{p} = n_\uparrow / n$ is the observed proportion of thumb-ups, and

[1] http://www.evanmiller.org/how-not-to-sort-by-average-rating.html

SIGIR'11, July 24–28, 2011, Beijing, China.
ACM 978-1-4503-0757-4/11/07.

$z_{1-\alpha/2}$ is the $(1-\alpha/2)$ quantile of the standard normal distribution. With the default parameter value $\alpha = 0.10$, the above score function estimates what the "real" proportion of thumb-ups at least is at 95% chance, therefore it balances the proportion of thumb-ups with the uncertainty due to a small number of observations. This method is considered as the current state of the art and thus adopted by many sites. For example, Reddit, a famous social news site, has mentioned in its official blog post[2] that this method is used for their ranking of comments. Nevertheless, this method is not well justified either. First, the above formula cannot be applied to calculate scores for the items that have not received any user-rating yet: the prevailing implementation assigns score 0 to such items, which is wrong since this implies that "no user-rating yet" is roughly same as "zero thumb-up vs. one billion thumb-downs". Second, as the lower bound is biased towards one side only, it always underestimates the "real" proportion of thumb-ups. Third, it is not clear how tight the lower bound is, i.e., how far it deviates away from the "real" proportion of thumb-ups. Fourth, the difference between the lower bound and the "real" proportion of thumb-ups are inconsistent for items with different number of user-ratings. Assume that item i has 1 thumb-up and 2 thumb-downs, while item j has 100 thumb-ups and 200 thumb-downs, this method would rank item i (whose score is 0.386) lower than item j (whose score is 0.575). However, this does not sound right, because while we are not really sure whether item i is good or bad, we have a lot of evidence that item j is bad, so we should rank item i higher than item j. For another example, using this method, we have $s(500, 501) > s(5, 1)$, which does not make much sense.

3. PROPOSED APPROACH

In this paper, we propose to address the problem of user-rating based ranking of items by formulating it as an extremely simple Information Retrieval (IR) [1] system: each user-rating — thumb-up or thumb-down — is considered as a *term*; each item is considered as a *document* that consists of a number of those two terms. Since users would like to find good items from the collection, the ranking of the items could be regarded as searching the collection with a virtual *query* of one term — thumb-up ($q = \uparrow$). The better ratings an item has received from users, the more *relevant* it is to the query thumb-up. According to the Probability Ranking Principle [2], we should rank documents by their probabilities of being relevant to the query, in our case, $\Pr[R = 1|i, \uparrow]$. This has been strictly proved to be the optimal retrieval strategy, in the sense that it minimises the expected loss (a.k.a. the Bayes risk) under 1/0 loss (i.e., you lose a point for either returning a non-relevant document or failing to return a relevant document). Making use of the Statistical Language Modelling [4] technique for retrieval, we treat each item i as a bag of user-ratings and construct a *unigram* model $M(i)$ for it, then the probability of an item being good (i.e., relevant to the query thumb-up) $\Pr[R = 1|i, \uparrow]$ can be calculated as the probability of the query being generated from its corresponding unigram model: $\Pr[\uparrow |M(i)]$. So the problem becomes how we can accurately estimate the probability $\Pr[\uparrow |M(i)]$ for each item i. Given only a small number of observed user-ratings, the maximum likelihood estimator using the proportion of thumb-ups (i.e., the second method mentioned in Section 2) does not work due to the limitation of its frequentist view of probabilities, which is a well-known fact in the Information Retrieval community. The solution is to move from frequentist inference to Bayesian inference. Before we see any user-rating for item i, we have a prior belief about the probability for it to get thumb-ups, e.g., $\Pr[\uparrow |M(i)] = 0.5$ if we do not have any reason to favour thumb-ups or thumb-downs. After we see a user-rating for item i, we should revise or update our belief accordingly, i.e., increase $\Pr[\uparrow |M(i)]$ when we see a thumb-up and decrease it otherwise. The above theoretical reasoning leads to the so-called *smoothing* technique in Statistical Language Modelling [4]. One of the simplest smoothing techniques is Laplace smoothing (a.k.a. Laplace's rule of succession), which assumes that every item "by default" has 1 thumb-up and 1 thumb-down (known as pseudo-counts). Although it avoids most flaws of those popular methods (such as getting zero-probability for unseen user-ratings), it probably puts too much weight on the prior. A better choice is its more generalised form, Lidstone smoothing, which assumes that every item "by default" has α thumb-ups and α thumb-downs, where $\alpha > 0$ is a parameter. Previous research studies have shown that the performance of Lidstone smoothing with $\alpha < 1$ is usually superior to $\alpha = 1$ (i.e., Laplace smoothing). From the Bayesian point of view, this essentially corresponds to the expected value of the posterior distribution of thumb-up probability, using beta distribution with parameter α as the prior and binomial distribution as the likelihood. To further account for the situation where users are risk-averse (e.g., in online shopping) or risk-loving, we could make it more general to use a non-uniform prior, i.e., assume that every item "by default" has α thumb-ups and β thumb-downs. In summary, we propose to use the following score function:

$$s(n_\uparrow, n_\downarrow) = \frac{n_\uparrow + \alpha}{(n_\uparrow + \alpha) + (n_\downarrow + \beta)},$$

where $0 < \alpha, \beta < 1$ are two parameters both of which have default values 0.5. It can be generalised to graded rating systems straightforwardly by taking each graded rating as multiple thumb-ups and thump-downs. For example, a 3-star rating in the 5-star scale system can be regarded as 3 thumb-ups and 5-3=2 thumb-downs. It can also be easily extended to take the ageing of user-ratings into account through Time-Sensitive Language Modelling techniques [5].

4. REFERENCES

[1] C. D. Manning, P. Raghavan, and H. Schütze. *Introduction to Information Retrieval*. Cambridge University Press, 2008.

[2] S. E. Robertson. *Readings in Information Retrieval*, chapter The Probability Ranking Principle in IR, pages 281–286. Morgan Kaufmann, 1997.

[3] E. B. Wilson. Probable inference, the law of succession, and statistical inference. *Journal of the American Statistical Association*, 22:209–212, 1927.

[4] C. Zhai. *Statistical Language Models for Information Retrieval*. Morgan and Claypool, 2008.

[5] D. Zhang, J. Lu, R. Mao, and J.-Y. Nie. Time-sensitive language modelling for online term recurrence prediction. In *In Proceedings of the 2nd International Conference on the Theory of Information Retrieval (ICTIR)*, pages 128–138, Cambridge, UK, 2009.

[2] http://blog.reddit.com/2009/10/reddits-new-comment-sorting-system.html