Unsupervised Linear Score Normalization Revisited

Ilya Markov
University of Lugano
Lugano, Switzerland
ilya.markov@usi.ch

Avi Arampatzis
Democritus University of Thrace
Xanthi, Greece
avi@ee.duth.gr

Fabio Crestani
University of Lugano
Lugano, Switzerland
fabio.crestani@usi.ch

ABSTRACT

We give a fresh look into score normalization for merging result-lists, isolating the problem from other components. We focus on three of the simplest, practical, and widely-used linear methods which do not require any training data, i.e. MinMax, Sum, and Z-Score. We provide theoretical arguments on why and when the methods work, and evaluate them experimentally. We find that MinMax is the most robust under many circumstances, and that Sum is—in contrast to previous literature—the worst. Based on the insights gained, we propose another three simple methods which work as good or better than the baselines.

Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval

Keywords

Score Normalization, Distributed Retrieval

1. INTRODUCTION

Merging ranked-lists produced by several engines requires two steps: score normalization and combination. In distributed setups with disjoint collections, the combination step becomes trivial since each document is assigned a single score from one of the participating engines. This makes such setups ideal in isolating the normalization problem. Distributed setups usually include another step before normalization, i.e. resource selection (RS). While useful in improving efficiency, RS has been also shown to improve effectiveness significantly. We argue that this is due to the far-from-perfect quality of state-of-the-art normalization methods: in an ideal normalization, e.g. scores are normalized to probabilities of relevance, any kind of RS will hurt effectiveness by excluding sources with relevant documents; in such an ideal situation, the more systems one combines, the better the effectiveness. Consequently, the theoretical ceiling of effectiveness can only be achieved with an ideal normalization without RS, and distributed setups with disjoint collections are best for experimenting with normalization [1].

Previously proposed normalizations vary from linear to non-linear functions which may require or not training data for their estimation. While there is a rich literature on the subject, most experiments reported do not isolate the problem as we described above [3, 4, 5]. We give a fresh look into the simplest normalization methods: linear functions which do not require any training or search engine cooperation. Being the most practical, they are also the fastest and generally have been proved to be robust and effective. Beyond an empirical comparison, we also give theoretical justifications on their implicit assumptions. To our knowledge, these assumptions have never been made explicit or they are 'lost' in the large volume of the related literature. Based on the gained insights, we propose three alternative linear normalization methods.

In the experiments we use the gov2.1000 and gov2.30 splits of the TREC GOV2 dataset [2]. In gov2.1000 the largest 1000 hosts of the GOV2 dataset are treated as 1000 sources, and the number of sources that contain relevant documents is usually much less than 1000. In gov2.30 the collections of gov2.1000 are clustered into 30 sources; here, relevant documents appear in most of the sources. We use ten retrieval functions implemented by the Terrier toolkit [1], namely, BM25, tf-idf (Terrier and Lemur versions), language modeling (original and with Dirichlet smoothing), and a number of DFR-based functions (BM25, BB2, IFB2, InL2 and PL2). Retrieval functions are randomly assigned to sources. Topics 701-850 are used as queries. Operationally, engines return truncated rankings for efficiency reasons, thus, we only consider either the top 10 or 1000 results. Our results are summarized in Tables 1 and 2, which we will refer to throughout the paper. Statistical significance is measured with a paired t-test. † shows the significantly best baseline (MinMax, Sum or Z-Score) and ‡ marks the modification that is significantly better than any baseline, both at the 0.05 level.

2. LINEAR SCORE NORMALIZATION

The MinMax method [3] scales the output score range per engine to $[0, 1]$. We argue that the most important assumption behind MinMax is that each source contains at least 1 relevant document, and that this document will most-likely get ranked 1st. By assigning the same highest score to all 1st documents, they are ranked before any other in the merged list. Since it is also assumed that these documents are most-likely relevant, a high early precision is achieved which pushes higher other evaluation measures sensitive to early rank positions, e.g. MAP. Thus, MinMax owes its success to getting right the early ranks in the merged list. However, due to fact that the 1st document of each engine is assigned the same highest score, MinMax produces a round-robin effect in merging which may impact effectiveness neg-

[1] `http://terrier.org`

SIGIR'12, August 12–16, 2012, Portland, Oregon, USA.
ACM 978-1-4503-1472-5/12/08.

atively as the number of engines increases. Indeed, MinMax shows high performance when run on 30 sources that contain sufficient relevant documents, i.e. the above assumption is satisfied. On the contrary, when run on 1000 sources, where the assumption is weakened by the sparseness of relevant documents and additionally the round-robin effect is more prominent, MinMax performance degrades considerably.

The Sum method [5] is similar to MinMax, but without using a fixed highest score eliminating the undesirable round-robin effect: the minimum score is shifted to 0 while the sum of all scores per ranked-list is scaled to 1: $s' = s - min$, $s_{norm} = s' / \sum_i s'_i$. The intuition given in [4] is: under the assumption of exponentially-distributed scores, the normalization is equivalent to setting the means of score distributions of sources to be equal to each other. However, many studies on modeling scores show that the aforementioned assumption of exponentially-distributed scores does not hold in practice, especially for top-ranked documents. Our experimental results show that the Sum method almost always has the worst performance. This contradicts the results of the original work [5], however, the authors in the last-mentioned study used a meta-search setup and evaluated normalization and combination methods together—a setup that does not isolate the normalization problem.

In [5] also the Z-Score method is proposed, which normalizes each score to its number of standard deviations that it is away from the mean score. However, as previously noted in [1], this method rather assumes a bell-like distribution of document scores (e.g. a Normal), where the mean would be a meaningful 'neutral' score. However, in practice score distributions are highly skewed and clearly violate this assumption. Still if only the top documents are considered from each result list, they are likely to be relevant, and therefore the distribution of their scores may be close to that of relevant documents, i.e. likely a Normal. Indeed, our results show that when only the top-10 documents are retrieved from each source, Z-Score usually shows higher performance than when applied on the top-1000 documents.

So far we have seen that methods that assume particular score distribution shapes, such as Sum and Z-Score, are worse, and that MinMax which does not make such assumptions is better. In other words, the sum or the mean of scores do not seem to be a good statistics for normalization, and that we should be looking into the direction of MinMax for developing a better normalization. We also know that using a fixed highest score should be preferably avoided so as to eliminate the round-robin effect, as well as, to be able to down-weigh sources of no relevance (haunting the main assumption of MinMax). Another problem rarely mentioned that all three methods above have is that they are greatly affected by the minimum score seen, or else, the chosen truncation point of the rankings (e.g. the minimum score of top-10 documents is usually very different from the minimum score of top-1000).

We will first try to deal with the minimum-score problem. The lowest theoretical score of many scoring functions is 0. Thus, making the assumption that the most non-relevant documents usually score at 0, the minimum in MinMax function should also be set to 0. This way we obtain the Max method: $s_{norm} = s/max$. Our results show that Max, although simpler, almost always has the same or higher performance than that of the MinMax method.

Let us now try to deal with non-relevant sources which violate the main assumption of MinMax. Scoring functions aim at assigning scores in a way that relevant documents have very different scores from non-relevant documents. Therefore, if the standard deviation σ of scores in a ranked-list is high, the list is likely to contain both relevant and non-relevant documents. If σ is low, the list is likely to contain either only relevant or only non-relevant documents. On one hand, long ranked-lists (e.g. 1000 documents) are likely to contain many non-relevant documents and, therefore, we prefer those that have high standard deviation (i.e. they also contain many relevant documents). In this case, the following linear modification makes sense: $s_{norm} = \sigma \frac{s-min}{max-min}$. We call it MM-Stdv. On the other hand, short ranked lists (e.g. 10 documents) may contain only relevant documents and, therefore, we might prefer the lists with low standard deviation. Unit-variance linear modification (UV) is similar to Z-Score and scale-invariant but does not shift the mean to 0: $s_{norm} = score/\sigma$. Our results support both formulas depending on the situation.

	gov2.30			gov2.1000		
	MAP	p@10	p@100	MAP	p@10	p@100
MinMax	0.0700†	**0.1953**‡	0.1674†	0.0027†	**0.0161**‡	**0.0140**‡
Sum	0.0171	0.0134	0.0292	0.0004	0.0013	0.0019
Z-Score	0.0367	0.0819	0.0935	0.0014	0.0013	0.0054
Max	**0.0722**	**0.1953**	**0.1734**‡	0.0023	**0.0161**	**0.0140**
MMStdv	0.0632	0.1738	0.1348	**0.0038**	0.0114	0.0088
UV	0.0182	0.1248	0.0676	0.0006	0.0013	0.0009

Table 1: Top 1000 documents are retrieved.

	gov2.30			gov2.1000		
	MAP	p@10	p@100	MAP	p@10	p@100
MinMax	0.0372†	0.1966	0.1437†	0.0027	0.0161†	0.0140
Sum	0.0355	0.1785	0.1401	0.0029	0.0013	0.0140
Z-Score	0.0353	0.1839	0.1413	0.0031	0.0007	0.0162
Max	**0.0397**‡	0.1953	**0.1656**‡	0.0028	0.0161	0.0140
MMStdv	0.0366	0.1604	0.1432	0.0025	0.0054	0.0139
UV	0.0381	**0.1980**	0.1498	**0.0049**‡	**0.0228**	**0.0225**

Table 2: Top 10 documents are retrieved.

3. CONCLUSIONS

We isolated the normalization problem and found that MinMax is the most robust method under many circumstances, Z-Score and Sum may perform well when some conditions are met, and Sum is worse than previous literature suggested. Furthermore, we gave theoretical insights on why and when these methods work or fail, and proposed three new methods that work as good or better than the baselines.

4. REFERENCES

[1] A. Arampatzis and J. Kamps. A signal-to-noise approach to score normalization. In *Proceeding of the ACM CIKM*, pages 797–806, 2009.

[2] J. Arguello, J. Callan, and F. Diaz. Classification-based resource selection. In *Proceedings of the ACM CIKM*, pages 1277–1286. ACM, 2009.

[3] J. H. Lee. Analyses of multiple evidence combination. In *Proceedings of the ACM SIGIR*, pages 267–276. ACM, 1997.

[4] R. Manmatha and H. Sever. A formal approach to score normalization for meta-search. In *Proceedings of the HLT*, pages 98–103. MKP Inc., 2002.

[5] M. Montague and J. A. Aslam. Relevance score normalization for metasearch. In *Proceedings of the ACM CIKM*, pages 427–433. ACM, 2001.