Simulating Signal and Noise Queries for Score Normalization in Distributed IR

Avi Arampatzis[1] Jaap Kamps[2,3]

[1] Electrical and Computer Engineering, Democritus University of Thrace, Greece
[2] Archives and Information Studies, University of Amsterdam
[3] ISLA, Informatics Institute, University of Amsterdam
avi@ee.duth.gr kamps@uva.nl

ABSTRACT

Score normalization is indispensable in distributed retrieval and fusion or meta-search where merging of result-lists is required. Distributional approaches to score normalization with reference to relevance, such as binary mixture models like the normal-exponential, suffer from lack of universality and troublesome parameter estimation especially under sparse relevance. We develop a new approach which tackles both problems by using aggregate score distributions without reference to relevance, and is suitable for uncooperative engines. The method is based on the assumption that scores produced by engines consist of a signal and a noise component which can both be approximated by submitting well-defined sets of artificial queries to each engine. We evaluate in a standard distributed retrieval testbed and show that the signal-to-noise approach yields better results than other distributional methods.

1. INTRODUCTION

Modern best-match retrieval models calculate some kind of score per collection item which serves as a measure of the degree of relevance to an input request. Scores are used in ranking retrieved items. Their range and distribution varies wildly across different models making them incomparable across different engines [4], even across different requests on the same engine if they are influenced by non-semantic query characteristics, e.g. length. Even most probabilistic models do not calculate the probability of relevance of items directly, but some order-preserving (monotone or isotone) function of it.

The main aim of this paper is to analyse and further develop score distributional approaches to score normalization. Our underlying assumption is that normalization methods that take the shape of the SD into account will be more effective than methods that ignore it. We want to make no assumptions on the search engines generating the scores to be normalized other than that they produce ranked lists sorted by decreasing score. Thus, we treat each engine as a 'black-box' and are interested in approaches based only on observing their input-output characteristics: the queries and resulting score distributions.

2. SINGLE DISTRIBUTION METHODS

Z-score A standard method for score normalization that takes the SD into account is the Z-SCORE. Scores are normalized, per topic and engine, to the number of standard deviations that they are higher (or lower) than the mean score:

$$\text{Z-SCORE:} \quad s' = \frac{s-\mu}{\delta}$$

where μ is the mean score and δ the standard deviation. Z-SCORE assumes a normal distribution of scores, where the mean would be a meaningful 'neutral' score. As it is well-known, actual SDs are highly skewed.

Aggregate Historical CDF Simplified A recent attempt models aggregate SDs of many requests, on per-engine basis, with single distributions [3] using the historical CDF:[1]

$$\text{HIS:} \quad s' = P(S_{\text{HIS}} \leq s)$$

where $P(S_{\text{HIS}} \leq s)$ is the *cumulative density function* (CDF) of the probability distribution of all scores, and HIS refers to the fact that historical queries are used for aggregating the SD that the random variable S_{HIS} follows. HIS normalizes input scores s to the probability of a historical query scoring at or below s.

3. SIGNAL-TO-NOISE METHODS

We investigate the use of dual aggregate SDs. Assuming that scores produced by an engine consist of two components, signal and noise, the score random variable S can be decomposed as:

$$S = S_{\text{SIGNAL}} + S_{\text{NOISE}}$$

The probability densities of the components are given respectively by p_{SIGNAL} and p_{NOISE} defined across the engine's output score range.

Furthermore, we assume 'stable' system characteristics for the engine in the sense that the signal and noise levels at a score depend only on the score. We can define a function which normalizes input scores s into the fraction of the signal at s:

$$\text{S/N:} \quad s' = \frac{p_{\text{SIGNAL}}(s)}{p_{\text{SIGNAL}}(s) + p_{\text{NOISE}}(s)} \qquad (1)$$

Since engines are expected to produce increasing signal-to-noise ratios as score increases, this may be an interesting normalization.

However, the magnitude of the original score is not taken into account. An obvious improvement would be to multiply S/N with a calibrated score s, for which we could use the HIS normalization:

$$\text{S/N*HIS:} \quad s' = \frac{p_{\text{SIGNAL}}(s)}{p_{\text{SIGNAL}}(s) + p_{\text{NOISE}}(s)}\, P(S_{\text{HIS}} \leq s) \qquad (2)$$

[*] This is an extended abstract of: A. Arampatzis and J. Kamps. A signal-to-noise approach to score normalization. In *Proceedings CIKM 2009*, pages 797–806. ACM Press, New York USA, 2009.

DIR-2010 January 25, 2010, Nijmegen, the Netherlands.

[1] We simplify their proposal by removing the quantile function that only gives a constant transformation which doesn't impact DIR.

The resulting scores would be comparable across engines, however, the distribution of the variable S_{HIS} depends on the availability of historical queries. Using historical queries, although very feasible and no cooperation is required, may lead to instabilities and biases. To deal with this, we can instead use the variable S_{SIGNAL}:

$$\text{S/N*SIG:} \quad s' = \frac{p_{\text{SIGNAL}}(s)}{p_{\text{SIGNAL}}(s) + p_{\text{NOISE}}(s)} P(S_{\text{SIGNAL}} \leq s) \qquad (3)$$

This calibrates s to the probability of having signal at or below s.

The question is how to approximate p_{SIGNAL} and p_{NOISE} per engine. Seeing engines as black-boxes similarly to the historical CDF approach, we can feed each one with queries of appropriate types and generate the needed functions based on the statistical properties of the observed output scores.

4. QUERY MODELS

We develop two models for generating artificial queries given a document collection. The resulting query sets produce aggregate SDs approximating S_{NOISE} (monkey query model) and S_{SIGNAL} (human query model).

Monkeys on Modified Typewriters In parallel to the popular thought experiment of a monkey hitting keys at random on a typewriter, let us imagine a keyboard with the terms of a query language on its keys plus "enter". The keys are considered equally accessible and of equal size, except "enter" which has a different size and thus different probability to be hit if keys are hit at random. The monkey, not understanding the grammar and semantics of the query language, will select terms uniformly. Moreover, terms will be independent. If p is the probability of hitting "enter", then the probability that the monkey will type k terms before hitting "enter" is given by (the discrete analogue of the *exponential distribution* called) the *geometric distribution*:

$$P(K_1 = k) = (1-p)^k p, \quad k = 0, 1, 2, \ldots$$

Note that a p fraction of the total queries will be of zero-length. The mean query length will be $1/p$.

Assuming r monkeys using identical keyboards (characterized by the same p) are typing independently, the random variable $K = \sum_{m=1}^{r} K_m$, where K_m is the geometrically distributed variable associated with the mth monkey, follows a *negative binomial distribution*:

$$g(k; r, p) = \binom{k+r-1}{k} p^r (1-p)^k, \quad k = 0, 1, 2, \ldots$$

Under an alternative parameterization, $\lim_{r\to\infty} g(k; r, p)$ converges to the *Poisson distribution* with a rate $\lambda = r(1/p - 1)$:

$$\text{Poisson}(k; \lambda) = \frac{\lambda^k e^{-\lambda}}{k!}$$

Humans on Search Engines Query terms occur, in general, in a dependent way (i.e. the occurrence of one makes the chances of occurrence of some others better than random) due to all of them pointing at the same topic. For natural language queries, there exists also serial dependence, imposed by grammar and semantics. When incorporating dependencies, retrieval models are becoming practically intractable, which led in the past to the infamous *term independence assumption*. Instead of trying to model term probabilities of occurrence and dependencies, we can rather tackle both features at once by picking real text fragments out of a corpus. The remaining question is how long those fragments should be.

Table 1: **Distributed retrieval results for TREC-123 and TREC-4 over all 100 engines. Significant-tested with a bootstrap test, one-tailed, at significance levels 0.05 (°), 0.01 (◉), 0.001 (•).**

	TREC-123			TREC-4		
run	P10	P20	P30	P10	P20	P30
ROUNDROBIN	0.1835	0.1835	0.1835	0.0584	0.0584	0.0584
Z-SCORE	0.2320◉	0.2285◉	0.2167◉	0.1300•	0.1130•	0.0940◉
HIS	0.2340°	0.2120$^-$	0.2017$^-$	0.1920•	0.1540•	0.1487•

Table 2: **Distributed retrieval results for TREC-123 and TREC-4 over all 100 engines.**

	TREC-123			TREC-4		
run	P10	P20	P30	P10	P20	P30
HIS	0.2400	0.2165	0.2047	0.1920	0.1540	0.1487
S/N	0.2630$^-$	0.2495°	0.2290$^-$	0.1980$^-$	0.1740$^-$	0.1560$^-$
S/N∗HIS	0.3020•	0.2770•	0.2537•	0.2380◉	0.1920◉	0.1740°
S/N∗SIG	0.3380•	0.3095•	0.2790•	0.2400°	0.2090•	0.1793°

Arampatzis and Kamps [1] arrive at a truncated Poisson/Power-law model of query length. The bulk of queries concentrates at short lengths where a power-law does not fit at all given the current query languages, therefore it makes practical sense to use a truncated mix of Poisson-Zipf to generate query lengths. In such a practical model, the lengths are Poisson-distributed for $k < k_0$ while they are Zipf-distributed for $k \geq k_0$. The choice of k_0 depends on the specific domain (i.e., a combination of features of the document collection, query/indexing language, and pattern of use of the system). As a rule of thumb, k_0 seems to be just above the mean observed query length.

5. EVALUATION: DIR TESTBEDS

Standard score normalization methods like the MinMax ignore the score distribution: $s' = \frac{s - min}{max - min}$, with min (max) the minimal (maximal) score per query and engine. That is, MinMax forces all scores in [0,1], resulting in a maximal score per topic and engine of 1. In DIR, we will be doing effectively a ROUNDROBIN picking the top result of each engine. We calculate also the Z-SCORE over the top 1,000 results, which is much more effective than ROUNDROBIN (see Table 1). The historical CDF approach HIS is also significantly better than ROUNDROBIN, and at least as good as Z-SCORE. We compare HIS against the new signal-to-noise methods S/N, S/N∗HIS, and S/N∗SIG. Table 2 presents the distributed retrieval results without resource selection. Overall, the S/N∗HIS and S/N∗SIG runs show significant improvements over the strong baseline of HIS, while the consistent improvements in S/N are mostly non-significant.

Acknowledgments This research was supported by the Netherlands Organization for Scientific Research (NWO), CATCH programme, under project number 640.001.501.

REFERENCES

[1] A. Arampatzis and J. Kamps. A study of query length. In *Proceedings SIGIR'08*, pages 811–812. ACM, 2008.

[2] A. Arampatzis and J. Kamps. A signal-to-noise approach to score normalization. In *Proceedings CIKM 2009*, pages 797–806. ACM Press, New York USA, 2009.

[3] M. Fernández, D. Vallet, and P. Castells. Using historical data to enhance rank aggregation. In *Proceedings SIGIR'06*, pages 643–644. ACM, 2006.

[4] S. Robertson. On score distributions and relevance. In *Proceedings of 29th European Conference on IR Research (ECIR'07)*, pages 40–51. Springer, 2007.