Modeling Score Distributions for Meta Search

R. Manmatha, T. Rath and F. Feng,

Center for Intelligent Information Retrieval,

Computer Science Department

University of Massachusetts, Amherst, MA 01003

(manmatha,trath,feng)@cs.umass.edu

Abstract

In this paper the score distributions of a number of text search engines are modeled. It is shown empirically that the score distributions on a per query basis may be modeled using an exponential distribution for the set of non-relevant documents and a normal distribution for the set of relevant documents. Experiments show that this model fits TREC-3 and TREC-4 data for a wide variety of different search engines including INQUERY a probabilistic search engine, SMART a vector space engine, and search engines based on latent semantic indexing and language modeling. The model also works when search engines index other languages like Chinese.

It is then shown that given a query for which relevance information is not available, a mixture model consisting of an exponential and a normal distribution can be fitted to the score distribution. These distributions can be used to map the scores of a search engine to probabilities. We also discuss how the shape of the score distributions arise given certain assumptions about word distributions in documents. We hypothesize that all 'good' text search engines operating on any language have similar characteristics.

This model has many possible applications. For example, the outputs of different search

engines can be combined by averaging the probabilities (optimal if the search engines are independent) or by using the probabilities to select the best engine for each query. Results show that the technique performs as well as the best current combination techniques.

1 Introduction

In this paper we model score distributions of text search engines using a novel approach. We first show that the score distributions for a given query may be modeled using an exponential distribution for the set of non-relevant documents and a normal distribution for the set of relevant documents. When relevance information is not available, these distributions may be recovered by fitting a mixture model with a Gaussian and an exponential component to the output scores of search engines on a per query basis. These distributions are then used to map the scores to probabilities using Bayes' Rule. This approach requires no training and makes no assumption on the kind of search engine used. The model has been shown to work for a large number of search engines based on different principles. Among the engines which have been tested using data from TREC-3 and TREC-4 include INQUERY, a probabilistic search engine, SMART, a vector space search engine, Bellcore's search engine based on latent semantic indexing and a search engine based on language modeling. This model has also been shown to work for score distributions of TREC-6 INQUERY and SMART data on Chinese. To our knowledge, this is the first attempt at recovering the relevant and non-relevant distributions when no relevance information is available.

The probabilities of relevance obtained from this model have many possible applications. For example thresholds for filtering may be selected using this approach or the probabilities may be used to combine the search from many distributed databases or multi-lingual or multi-modal databases. Here, we will focus on using them to combine the outputs of different search engines (the meta-search problem).

Most combination methods proposed in the literature are ad hoc in nature and often involve the linear combination of scores [10]. This is unsatisfactory as scores from different search engines

can be very different since they are often the result of computing some metric (or non-metric) distance over sets of features. Both the distance and the features may vary from engine to engine. In fact, even the distributions of scores of different search engines could vary widely - for example, the scores of relevant documents may be clumped together for one engine while those of a second engine may be distributed in a much more uniform manner. A linear combination of results in such cases could produce misleading results. The problem is more acute when search engines operating on different media have to be combined for then the scores really mean different things.

The approach proposed here allows us to combine the outputs of search engines using the probabilities derived from the model of score distributions. In this paper we examine two approaches to combination. The first involves averaging the probabilities which is optimal in the sense of minimizing the Bayes' error if the search engines are treated as independent classifiers [24]. The second approach involves using the probabilities to discard "bad" engines while keeping the "good" ones. We show that the combination approaches proposed using these techniques do as well as the best combination techniques proposed in the literature. In addition, our technique is less ad-hoc and easier to justify. The technique can also be extended to multi-lingual and multi-modal combination.

The rest of the paper is divided as follows. Section 2 discusses prior work in modeling score distributions as well as in the area of combination. This is followed by Section 3 which discusses the modeling of score distributions of relevant and non-relevant documents and how these distributions may be recovered in the absence of relevance information by using a mixture model. Solving for the mixture model using Expectation-Maximization (EM) is also discussed. Finally, Bayes' Rule is used to map the scores to probabilities of relevance. Section 4 discusses the theoretical intuition behind using such models. Section 5 discusses how the model and the probabilities derived from it can be used for evidence combination. Finally, Section 6 concludes the paper.

2 Prior Work

Note that it is not obvious that the non-relevant data should be modeled using an exponential and the relevant data with a Gaussian. A number of researchers in the 60's and 70's starting with Swets [23] proposed fitting both the relevant and non-relevant scores using normal distributions and then using statistical decision theory to find a threshold for deciding what was relevant. Bookstein [7] pointed out that Swets implicitly relied on an equal variance assumption. Bookstein also raised the issue of whether it might be more appropriate to model the score distributions using Poissons. This modeling does not appear to have been done. van Rijsbergen [25] commented that for search engines like SMART there was no evidence that the distributions were similarly distributed let alone normally. We observe here that the empirical data for a large number of search engines clearly shows that the two distributions are not similar. Vogt and Cottrell [27] showed that in many cases (but not always) the scores for non-relevant documents *averaged over all 50 queries* were exponentially distributed. We note that the work in this paper focuses on score distributions on a per query basis. The results are not likely to be true when averaged over multiple queries.

Baumgarten [5] discussed fitting the scores of all (roughly the top 1000) documents (not just relevant or non-relevant) of one particular search engine using a Gamma distribution in the context of distributed retrieval. We note that when the number of relevant documents is small, a good approximation to the combined distribution is an exponential distribution. In such a situation the Gamma distribution, which is a generalization of an exponential distribution, will model the score distribution of all documents. However, there are a number of disadvantages of this model. First, the relevant and non-relevant distributions are not estimated separately hence limiting its usefulness. For example a posterior probability can no longer be computed from the complete distribution. Second, the approximation breaks down when the number of relevant distributions is large. Finally, the Gamma distribution (in this case) tends to give a lot of weight to the scores near zero. However, the estimates of the distribution of the scores near zero are likely to be misleading since the data provided only consists of a small fraction of all documents.

Independent of the work described in this paper, Arampatzis and his collaborators [2, 1] recently discussed how the relevant scores could be modeled using a Gaussian distribution and the non-relevant scores using an exponential distribution for a filtering application. Based apparently on modeling one filtering engine, they argue that the Gaussian only arises when massive query expansion is undertaken ($\approx$ 250 terms). Our results indicate that the Gaussian can arise when the number of query terms is much smaller. Zhang and Callan [29] also discuss using the model described in [19, 2] for filtering.

All previous researchers (including Arampatzis et al [2, 1]) assume that relevance information for some or all of the set is available. To our knowledge, there is no literature on recovering the relevant and non-relevant distributions when no relevance information is available and ours is the first attempt to do this.

A recent and extensive survey of evidence combination in information retrieval is provided by Croft [10]. Tumer and Ghosh [24] discuss past work in a related area - the combination of classifiers. They show that for statistically independent classifiers, the optimal combination is obtained by averaging the probabilities. They define optimality as equivalent to minimizing the Bayes' error.

Fox and Shaw [12] proposed a number of combination techniques including operators like the MIN and the MAX. Other techniques included one that involved setting the score of each document in the combination to the sum of the scores obtained by the individual search engines (COMBSUM), while in another the score of each document was obtained by multiplying this sum by the number of engines which had non-zero scores (COMBMNZ). Note that summing (COMBSUM) is equivalent to averaging while COMBMNZ is equivalent to weighted averaging. Lee [17, 18] studied this further with six different engines. His contribution was to normalize each engine on a per query basis improving results substantially. Lee showed that COMBMNZ worked best, followed by COMBSUM while operators like MIN and MAX were the worst. Lee also observed that the best combinations were obtained when systems retrieved similar sets of relevant documents and dissimilar sets of non-relevant documents. Vogt and Cottrell [26] also

verified this observation by looking at pairwise combinations of systems. A probabilistic approach using ranks rather than scores was proposed last year by Aslam and Montague [4, 3]. This involved extensive training across about 25 queries to obtain the probability of a rank given a query. Their results for TREC-3 were close to but slightly worse than Lee's COMBMNZ technique [1]. Aslam and Montague were able to demonstrate that rank information alone can be used to produce good combination results. The main difficulty with this technique seems to be the extensive training required of every engine on a substantial number of queries.

A number of people have also looked at the problem of combining outputs of systems which search overlapping or disjoint databases. Voorhees et al [28] experimented with combination using a set of learned weights. Callan [8] gave a value to each database. He showed that weighting the scores by this value was substantially better than interleaving ranks. Some researchers have also investigated the notion of combining search engines over multiple media. QBIC [11] combined scores from different image techniques using linear combination.

3 Modeling Score Distributions of Search Engines

In this section we describe how the outputs of different search engines were modeled using data from the text retrieval conferences (TREC). TREC data provides the scores and relevance information for the top 1000 documents for different search engines. For the experiments here data from the ad hoc track of the TREC-3 and TREC-4 for a number of different search engines was used. We will show examples of the modeling on queries from INQUERY and SMART from TREC-3. INQUERY is a probabilistic search engine from the University of Massachusetts, Amherst while SMART is a vector space engine from Cornell University.

Our modeling begins with TREC-3 data for INQUERY. There are 50 queries available with document scores and relevance information for each query. We examine the relevant and non-relevant data separately. The data are first normalized so that the minimum and maximum score

[1] The graph for Lee's technique in [4] is incorrect.

for a query are 0 and 1 respectively.

Figure 1 shows a histogram of scores for query 151 from TREC-3 for a set of non-relevant documents. The histogram clearly shows the rapid fall in the number of non-relevant documents with increasing score [2]. A maximum-likelihood fit of an exponential curve to this data is also shown. For the purposes of fitting the exponential, the origin is shifted to the document with the lowest score. It can be shown that the maximum-likelihood for an exponential is obtained by setting the mean of the exponential to the sample mean of the data [6].

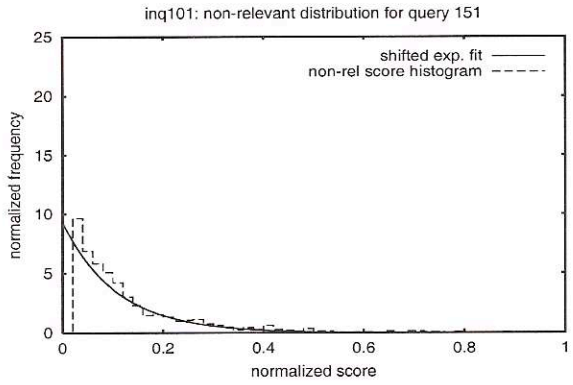

Figure 1: Histogram and shifted exponential fit to non-relevant data for query 151 INQUERY (inq101)

Figure 2 shows a histogram of scores for the set of relevant documents for the same query. The histogram approximates a normal distribution. The plot also shows a maximum-likelihood fit using a Gaussian with mean 0.466 and variance 0.042. The maximum-likelihood fit involves setting the mean and variance of the Gaussian to the sample mean and sample variance respectively of the data [6]. The exponential fit obtained for the non-relevant documents is also plotted in the figure.

The same process was repeated for all 50 queries in this track and in most cases it was found possible to approximate the non-relevant data with exponentials and the relevant data using Gaussians. The relevant data can be fitted with a Gaussian reasonably well when there is a sufficient number of relevant documents. Usually more than about 30 relevant documents were needed.

[2]Since we truncate at the top 1000 documents, the counts in the first bin (near the zero end) of the histogram will be erroneous. Thus, the first bin of the histogram is deleted for the experiments here.

When the number of relevant documents was small, the fit was bad. However, we believe this is not because the Gaussian was a bad fit but because we don't have enough relevant documents to compute the statistics in these cases. The exponential was also a good fit to the non-relevant data.

Our modeling has focused on using the top 1000 documents. This is because TREC only provides the results for the top 1000 documents. However, we note that our results will not change in any practical way if the top 2000 or the top 5000 documents were used (this has been experimentally verified for some search engines for which we have data). If anything, the Gaussian distributions are estimated better since the number of relevant documents will usually increase.

When relevance information is known, one can fit the scores using either a non-parametric distribution like a histogram or a parametric distribution as done here. When running a new query, however, relevance information is not available. Clearly, it would be useful to fit the score distributions of such data. A natural way to do this is to fit a mixture model of a shifted exponential and a Gaussian to the combined score distribution and this is discussed in the next section. We note that it is not practical to use a non-parametric technique like a histogram when relevance information is not available.

3.1 Mixture Model Fit

Consider the situation where a query is run using a search engine. The search engine returns scores but there is no relevance information available. We show below that in this situation, a mixture model consisting of an exponential and a Gaussian may be fitted to the score distributions. We can then identify the Gaussian with the distribution of the relevant information in the mixture and the exponential with the distribution of the non-relevant information in the mixture. Essentially this allows us to find the parameters of the relevant and non-relevant distributions without knowing relevance information apriori.

The density of a mixture model $p(x)$ can be written in terms of the densities of the individual

components $p(x|j)$ as follows: [6, 20]

$$p(x) = \sum_j P(j)p(x|j) \tag{1}$$

where j identifies the individual component, the $P(j)$ are known as mixing parameters and satisfy

$$\sum_{j=1}^{2} P(j) = 1, 0 \leq P(j) \leq 1. \tag{2}$$

We will denote densities with a lower case p(x) and probabilities with an uppercase P(x). In the present case, there are two components, an exponential density with mean λ

$$p(x|1) = \lambda \exp(-\lambda x) \tag{3}$$

and a Gaussian density with mean μ and variance σ^2

$$p(x|2) = \frac{1}{\sqrt{2\pi}\sigma} \exp(-\frac{(x-\mu)^2}{2\sigma^2}) \tag{4}$$

A standard approach to finding the maximum likelihood solution to the mixture model (that is, obtaining the values of the mixing parameters and the parameters of the component densities) is to use Expectation Maximization (EM) [6, 20]. This is an iterative procedure where the Expectation and Maximization steps are alternated. The reader is referred to [6, 20] for a good introduction to EM.

In the E-step the current value (denoted by the superscript old) and new values of the parameters (denoted by the superscript new) are used to compute the following expectation (where E^{comp} is known as the expectation of the complete data likelihood):

$$E[E^{comp}] = -\sum_{n=1}^{N}\sum_{j=1}^{M} P^{old}(j|x^n) ln\{P^{new}(j)p^{new}(x^n|j)\} \tag{5}$$

where

$$P(j|x) = \frac{p(x|j)P(j)}{p(x)}$$

and $P(j|x)$ is the posterior probability of j given x. x^n is a sample point, N is the number of samples and M is the number of mixture components.

The M-step involves maximizing the expectation after using the expressions for the densities of the two components (in this case exponential and Gaussian). The maximization is done by differentiating the expectation with respect to the parameters and setting the derivatives to zero. This maximization leads to a set of update procedures for the parameters of the densities and the mixing parameters. In the Appendix we derive the update equations obtained using EM for the parameters. The update equations are:

$$\mu^{new} = \frac{\sum_n P^{old}(2|x^n)x^n}{\sum_n P^{old}(2|x^n)} \tag{6}$$

$$(\sigma^{new})^2 = \frac{\sum_n P^{old}(2|x^n)||x^n - \mu^{new}||^2}{\sum_n P^{old}(2|x^n)} \tag{7}$$

$$\lambda^{new} = \frac{\sum_n P^{old}(1|x^n)}{\sum_n P^{old}(1|x^n)x^n} \tag{8}$$

$$P(1)^{new} = \frac{1}{N}\sum_n P^{old}(1|x) \tag{9}$$

Note that the updated Gaussian parameters are given by μ^{new} and σ^{new} while the updated exponential parameter is given by λ^{new}. $P(1)^{new}$ is the current estimate of the mixing parameter for the exponential ($P(2) = 1 - P(1)$).

The procedure needs an initial estimate of the component densities and mixing parameters. The EM procedure is terminated when the change in the value of value of the parameters between successive iterations is less than some predetermined threshold. Using EM to fit the data gives the mixture fit shown in Figure 3. The figure plots the mixture density as well as the component densities for the exponential and Gaussian fits. For comparison, Figure 2 shows the exponential and Gaussian fits to the non-relevant and relevant data. Comparing the two figures, it appears that the strategy of interpreting the Gaussian component of the mixture with the relevant distribution

and the exponential component of the mixture with the non-relevant distribution is a reasonable one. We should note that the correspondence between the mixture components and the fits to the relevant/non-relevant data is not always as good as that shown here but in general there is reasonable agreement.

The same technique was used to model the result of query 151 for the Cornell SMART vector space engine. Similar results were obtained as shown in Figures 4 and 5.

This model has been fitted to a large number of search engines on TREC-3 and TREC-4 data including probabilistic engines like INQUERY and CITY and a vector space engine (SMART) as well as Bellcore's LSI engine. The fit appears to be better for "good" search engines (engines with a higher average precision in TREC-3) and worse for those with a lower average precision. The model was also tested successfully on an engine which uses language models - specifically the language model approach of Lavrenko and Croft. [16] This approach has also been used successfully to model the document scores for searches on INQUERY and SMART indexing a Chinese database (TREC-6 data).

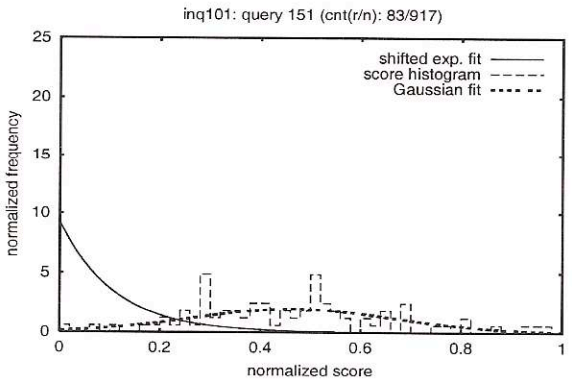

Figure 2: Exponential fit to non-relevant data and Gaussian fit to relevant data for query 151 INQUERY (inq101)

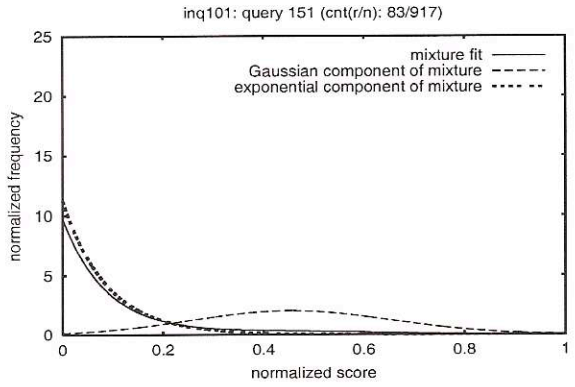

Figure 3: Mixture model fit showing exponential component, Gaussian component and the combined mixture for query 151 INQUERY (inq101). Compare with Figure2

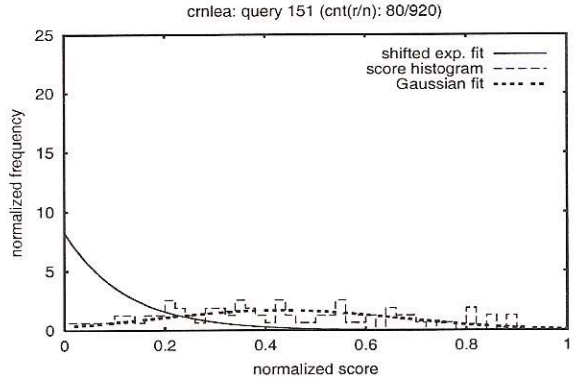

Figure 4: Exponential fit to non-relevant data and Gaussian fit to relevant data for query 151 SMART (crnlea).

3.2 Computing Posterior Probabilities

Using Bayes' Rule one can compute the probability of relevance given the score as

$$P(rel|score) = \frac{p(score|rel)P(rel)}{p(score|rel)P(rel) + p(score|nonrel)P(nonrel)}$$

where $P(rel|score)$ is the posterior probability of relevance of the document given its score, $p(score|rel)$ and $p(score|nonrel)$ are the probabilities of the score given that the document is relevant and the score given that the document is non-relevant respectively. P(rel) and P(nonrel) are the prior probabilities of relevance and non-relevance.

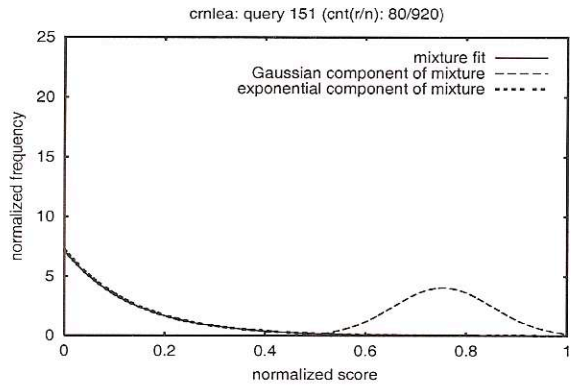

Figure 5: Mixture model fit showing exponential component, Gaussian component and the combined mixture for query 151 SMART (crnlea). Compare with Figure 4

In our model, $P(score|rel)$ is given by the Gaussian component of the mixture while $P(score|nonrel)$ is given by the exponential part of the mixture. P(rel) and P(nonrel) may be obtained by using the mixing parameters. Thus, $P(rel|score)$ can be computed in a simple manner.

Figure 6 shows the posterior probability obtained for SMART for query 164 using the previously determined mixture fits. The figure shows the posterior probabilities obtained from the separate Gaussian and exponential fits when relevance information is available and also the posterior probabilities obtained from the Gaussian and exponential part of the mixture. P(rel) and P(nonrel) are taken to be the mixing parameters in this case. Note that the differences in the two curves reflect fitting errors both for the mixture fit as well as the separate Gaussian and exponential fits obtained when relevance information is available.

In general, we expect the posterior probabilities to be a monotonic function of the score. In other words, as the score increases so should the posterior probability. In some cases we may have the situation depicted in Figure 7 where the posterior seems to decrease with increasing scores. The figure depicts the posterior probabilities for INQUERY for query 154 using TREC-3 data. This situation arises because the Gaussian density falls much more rapidly than the exponential and hence the two densities intersect twice. Note that in this case the posterior probabilities obtained both from the mixture fit (no relevance information available) as well as that obtained using relevance data show this problem. One solution would be not to use a Gaussian density but to use

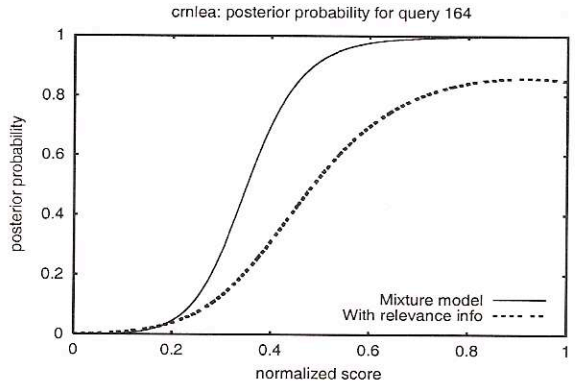

Figure 6: Posterior probability for query 164 for the SMART engine for TREC-3 data. The dotted line is obtained from the separate Gaussian and exponential fits computed using relevance information. The solid line is obtained from the mixture fits.

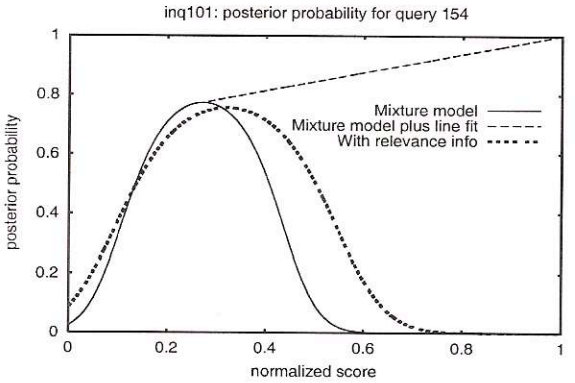

Figure 7: Posterior probability for query 154 for the INQUERY engine for TREC-3 data. The bold dotted line is obtained from the separate Gaussian and exponential fits computed using relevance information. The solid line is obtained from the mixture fits. The dotted line joins the maximum point of the mixture to the point(1,1). The final posterior mapping follows the solid line up to the maximum point and then the straight line curve thus preserving monotonicity

another function which has the same form (like a Gamma distribution) but decreases less rapidly. However, as we discuss below, an exponential-Gamma mixture does not perform as well. Instead we force the posterior probability to be monotonic by drawing a straight line from the point where the posterior is maximum to the point (1,1). The final posterior probability curve is given by the portion of the posterior probability computed using Bayes' rule up to the maximum portion of the curve and the straight line thereafter.

We have assumed that the priors P(rel) and P(nonrel) may be estimated using the mixing parameters. When there are few relevant documents the mixing parameters provide a poorer estimate of the priors. In a normal retrieval the number of relevant documents is small and hence estimates of the mixing parameters are less accurate. Extensive experiments have shown that P(nonrel) is best estimated using the following procedure. Let P(1) be the mixing parameter corresponding to the exponential. Then

$$P(nonrel) = \begin{cases} P(1) & if\ P(1) \leq 0.8 \\ 0.8 & otherwise \end{cases}$$

and P(rel) = 1 - P(nonrel). This approach to estimating the priors improves the average precision results slightly when we combine results.

3.2.1 Other Mixture Models

There is a large family of densities which could possibly have fit the data. For example, the Poisson and Gamma distributions approximate the Gaussian for appropriate parameter choices. However, using a Poisson/Poisson (non-relevant/relevant) or an exponential/ Poisson combination did not fit the data well. We also tried a mixture model with an exponential distribution for the scores of the non-relevant document and a Gamma distribution for the scores of the relevant documents. The rationale for doing this was that the Gamma distribution would fall off more slowly than a Gaussian on the right hand side and might, therefore, lead to better posterior distributions.

The Gamma distribution is given by:

$$p(x) = \frac{1}{\theta^k \Gamma(k)} x^{k-1} \exp^{-x/\theta}$$

The update equations for a solution using EM of the exponential-Gamma mixture are derived

in Appendix 2. The exponential and mixing parameters are updated as before:

$$\lambda^{new} = \frac{\sum_n P^{old}(1|x^n)}{\sum_n P^{old}(1|x^n)x^n} \tag{10}$$

$$P(1)^{new} = \frac{1}{N}\sum_n P^{old}(1|x^n) \tag{11}$$

The Gamma update parameters are more complicated to derive (because of the Gamma function in the denominator). From Appendix 2 k^{new} is given by:

$$k^{new} = \begin{cases} \frac{0.5000876+0.1648852\eta-0.0544274\eta^2}{\eta} & 0 \leq \eta \leq 0.5772 \\ \frac{8.898919+9.059950\eta+0.9775373\eta^2}{\eta(17.79728+11.968477\eta+\eta^2)} & 0.5772 \leq \eta \leq 17.0 \\ \frac{1}{\eta} & 17 \leq \eta \end{cases}$$

where

$$\eta = \frac{\sum_n P^{old}(2|x^n) ln \frac{\sum_s P^{old}(2|x^s)x^s}{x^n \sum_s P^{old}(2|x^s)}}{\sum_n P^{old}(2|x^n)}$$

The update equation for θ^{new} is then given by:

$$\theta^{new} = \frac{\sum_n P^{old}(2|x^n)x^n}{k^{new}\sum_n P^{old}(2|x^n)}$$

While numerically fitting the exponential-Gamma distribution one needs to be careful since it tends to "blow-up" because of the Gamma function in the denominator of the Gamma distribution. The results for the exponential-Gamma distribution were not as good as the ones obtained using the exponential-Gaussian distribution. Thus our choice of distributions - exponential for the non-relevant and Gaussian for the relevant - is dictated by two considerations; that the functions fit the data well and that they can be recovered using a mixture model when relevance information is not available.

3.2.2 Initial Conditions

Like any non-linear equation solver, EM may find solutions arising from local maxima and is sometimes sensitive to initial conditions [20]. Arbitrary initial conditions were tried and some sensitivity to initial conditions was noticed but usually for the poorer search engines (search engines much lower down in a TREC-3 ranking by average precision).

4 Shape of Distributions

We will now attempt to give some insight into the shape of the score distributions.

Given a term (or word) assume that the distribution of this term in the set of relevant documents is given by a Poisson distribution with parameter λ_r. That is,

$$P_r(x) = \frac{\lambda_r^x \exp(-\lambda_r)}{x!}$$

where x is the number of times that the term occurs in a particular document and $P_r(x)$ is the probability of x occurrences of the term in the set of relevant documents. Also assume that its distribution in the set of non-relevant documents is given by another Poisson distribution with the parameter λ_n. That is,

$$P_n(x) = \frac{\lambda_n^x \exp(-\lambda_n)}{x!}$$

where $P_n(x)$ is the probability of x occurrences of the term in the set of non-relevant documents. In general, λ_n will be much smaller than λ_r.

Numerous attempts have been made to model word distributions in the past. Harter [15] used a mixture of 2 Poissons to model the distributions of words in a document. Our model in this section is closely related to his model. It has been argued by some researchers that the 2 Poisson model is not a good approximation and that other distributions like the negative binomial are better models of the distributions of words in documents [21]. A mixture of a large number of Poissons has also been used to fit the data [9]. Since we would like to fit a distribution to the relevant

and another to the non-relevant, it is much more convenient for us to assume the 2-Poisson model here. Additionally, the main purpose of this section is to provide some insight and not a rigorous derivation.

Given a query consisting of 1 term and assuming that the score given to a document is proportional to the number of matching words in the document, the distribution of the scores of relevant documents is then given by the Poisson distribution:

$$P_r(x) = \frac{\lambda_r^x \exp(-\lambda_r)}{x!}$$

and the distribution of the scores of non-relevant documents is given by the Poisson distribution:

$$P_n(x) = \frac{\lambda_n^x \exp(-\lambda_n)}{x!}$$

The actual scores for many search engines is weighted by some function of the term frequency and the inverse document frequency. However, empirical evidence [14] shows that the score may be reasonably approximated as being proportional to the number of matching words.

For the set of relevant documents, λ_r will usually be large. For large values of λ, the Poisson distribution tends to a normal distribution (see Figure 8). On the other hand for small values of λ, the Poisson distribution will tend towards a distribution which is falling rapidly (see Figure 8). The shape of these curves is consistent with the experimental modeling of scores for TREC-3 and TREC-4 data (see the previous section). The experiments showed us that the normal distribution is a good fit for the score distributions of the relevant data. For non-relevant data, the experiments show that the exponential distribution is a good fit. For small λ_n, the Poisson distribution shows a decreasing distribution. Although, this is not the same as an exponential distribution, it does have the same general shape as an exponential (rapid monotonic decrease).

It is much harder to derive the score distributions if the query consists of two or more terms. This is because the actual scores of search engines are complicated functions. However, there is

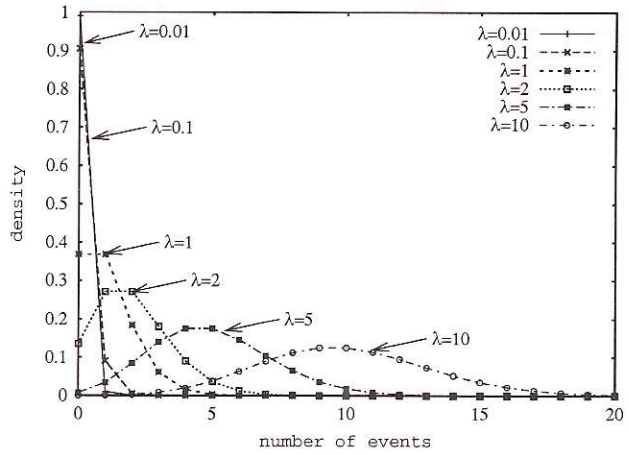

Figure 8: The Poisson distribution for different values of λ.

empirical evidence that the major contribution to the scores is provided by the number of matching terms [14]. We also note that Robertson and Walker [22] motivated a $tf - idf$ scoring function from the 2-Poisson model. We assume first that the score is proportional to the number of matching words and provide an intuitive argument for queries with two or more terms. For simplicity we will consider the case where the query has just two terms but the argument applies in general. In this case we can assume that the two terms say "oil" and "spill" can be clubbed together to create a single term - "oil spill". Then the λ_r (the average frequency of a term over relevant documents) for joint occurrences of this term "oil spill" is much lower than the λ_r for either "oil" or "spill". In other words the chances of finding the combination "oil spill" are much lower than that of finding "oil" or "spill" separately. When the query contains two terms, it is reasonable to assume that the λ_n (i.e. the average frequency over non-relevant documents) does not change much as it essentially reflects the background probabilities of the word.

The Poisson model for the shape of the relevant and non-relevant distributions that we have derived applies to both probabilistic engines like INQUERY and vector space engines like SMART. For vector space engines the number of matching terms is given by the dot product of two vectors - one representing the query and the other the document. Further, this model is language independent (as long as word frequencies in any language have an approximately 2-Poisson like distribution).

Thus, we predict that a mixture of exponential and Gaussian distributions will fit a much larger class of text search engines operating on different languages.

5 Combining Search Engines

The posterior probabilities obtained by using the model discussed above has many possible applications. For example the probabilities could be used to select a threshold for filtering documents or for combining the outputs in distributed retrieval. Here we discuss one possible application which involves combining the outputs of different search engines on a common database to improve results.

In general the range of search engine scores may vary quite a bit - for example, one engine may have scores ranging from 0 to 1 while another can have scores ranging from -20 to 150. Other approaches to combination have focused on normalizing the range of the scores and then fusing them by simple techniques like linear combination or by taking the minimum and maximum scores. However, range normalization does not take into account the actual distribution of the scores. Consider, for example, the model of the scores discussed previously where the scores of the relevant documents follow a normal distribution and the scores of the non-relevant follow an exponential distribution. Also consider two different search engines which have different parameter values for these distributions. A simple (linear) range normalization and combination clearly does not suffice.

There are a number of possible ways the probabilities can be used to combine the search engines. We propose two alternative schemes for combination. The first involves averaging the probabilities. This is optimal in the sense of minimizing the Bayes' error if the search engines are independent [24]. Of course the outputs of search engines are not necessarily independent. In the following figures and discussions, data are taken from TREC-3. inq101 and crnlea denote runs of the INQUERY and SMART engines, META200 denotes combination by averaging the posteriors obtained using the mixture model, while META900 denotes the combination by averaging the posterior probabilities using the Gaussian and exponential fits assuming relevance is given. Thus, any

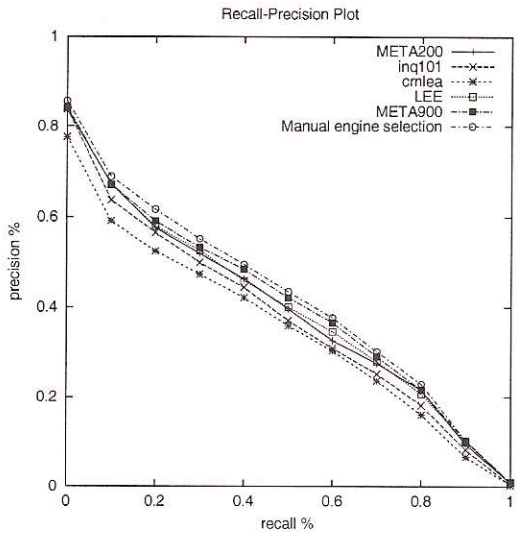

Figure 9: Recall precision graphs for combining inq101 and crnlea using different techniques (see text). Data from TREC-3

difference between META200 and META900 is caused by the errors in performing a mixture fit to the model. LEE denotes Lee's COMBMNZ technique while the manual engine selection technique involves selecting the best engine(s) and discarding the worst engine(s) on a per query basis using the average precision for that query. Manual engine selection provides an indication of the best combination result we can achieve. Note that both META900 and manual engine selection require relevance information and are only plotted to provide a baseline for understanding the limits of combination.

Figure 9 shows recall-precision plots for combining INQUERY and SMART on TREC-3 data. Precision is defined as the fraction of retrieved documents which are relevant while recall is the fraction of relevant documents which have been retrieved. The recall-precision graph is usually created by averaging over a certain number of queries - in this case 50. As the figure shows META200 performs considerably better than either INQUERY and SMART - in fact its average precision is 6% better than that of INQUERY and 13% better than that of SMART. LEE is slightly better (about 1%) than META200 although the difference is not significant. META900 has an average precision about 10% better than INQUERY and clearly performs better than either META200

or LEE's implying that if the mixture fit could be improved the technique would perform even better. Finally, the plot for manual engine selection clearly indicates that both META200 and LEE's are close to obtaining the best performance possible from combination.

Figure 10 describes combination results for the top five engines in TREC-3. The x-axis is the number of engines combined while the y-axis is the average precision. As the plot shows combination clearly improves the results. There are four graphs in the figure. The first curve is the average precision of the individual search engines. The second plot META200 shows the combination method applied to 1, 2, 3, 4 or 5 engines. As can be clearly seen there is a considerable improvement over using even the best search engine and overall the improvement seems to increase with the number of search engines combined. With the top 2 engines, META200 shows an improvement of 6% over the best single engine and using the top 3 engines, META200 shows an improvement of almost 12%. LEE's COMBMNZ technique is also shown in the same graph. Its average precision is seen to be slightly worse than META200 but the difference is not really significant. The performance obtained using META900 (i.e. combination with the posterior probabilities obtained with relevance information) is 15% better than the best single engine. Again this indicates that if the mixture fit were improved we could do even better.

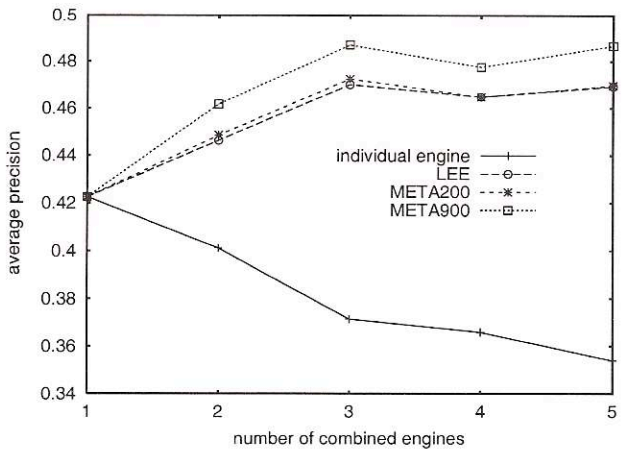

Figure 10: Recall precision graphs for combining the best five techniques from TREC-3.

Figure 11 demonstrates that this approach also works for other languages. The figure shows

the combination results for INQUERY and SMART when indexing a Chinese database. The data in this case is from TREC-6. As can be clearly seen, combination using both META200 and LEE's COMBMNZ show an improvement over either engine. However, in this particular case the improvement is much less than that for English. Also the difference between META900 and META200 is small indicating that perhaps we are close to the limit of what can be achieved.

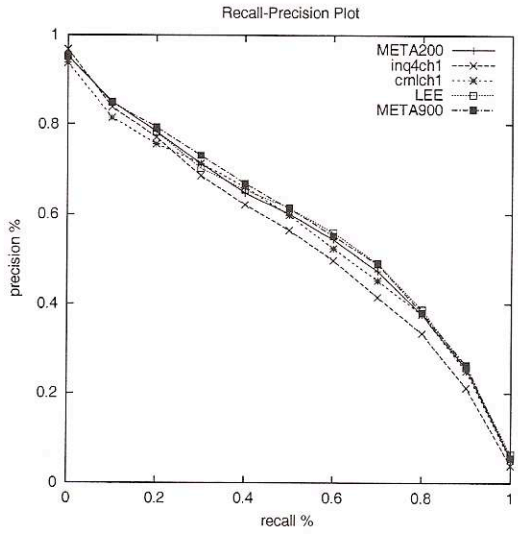

Figure 11: Recall precision graphs for combining inq4ch1 and crnlch1 with Chinese queries and Chinese databases. Data from TREC-6

Combination of "good" search engines usually improves the scores. Partly this reflects the fact that the score distribution models fit "good" search engines better than "poor" search engines. However, the combination of two search engines when the performance of one is substantially worse than the others leads to a result which can be worse than that of the best engine. This is consistent with the well known observation that combining a bad classifier with a good classifier can lead to a result which may not be better than the best individual classifier. Two search engines INQUERY (inq101) and XEROX (xerox4) were picked. On the basis of average precision, inq101 is ranked 4th while xerox4 is ranked 35th among 40 engines in TREC-3. The average precision of inq101 is more than twice that of xerox4. Figure 12 clearly shows that INQUERY performs much better than the XEROX engine. The combination META200 is much better than XEROX

but worse than INQUERY. LEE's is slightly better than META200 but still worse than INQUERY. The best option in such cases is to avoid combination.

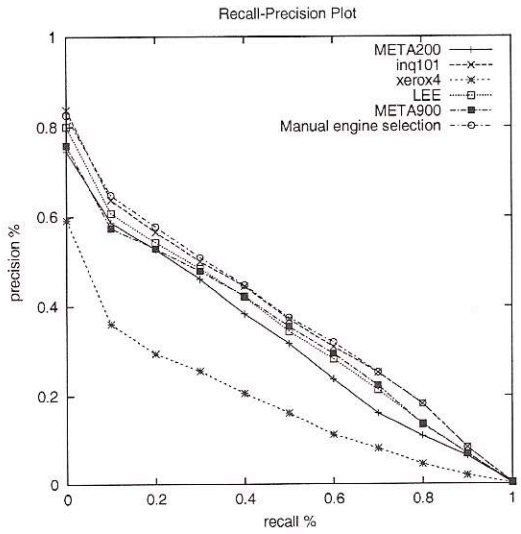

Figure 12: Recall precision graphs for combining inq101 and xerox4 using different techniques. Data from TREC-3.

5.1 Automatic Engine Selection

The previous example shows that if we could have somehow figured out that we need to pick INQUERY as the best possible engine for every query then the performance would improve considerably. The ability to model and compute the relevant and non-relevant distributions allows us to develop techniques to automatically selecting engines on a per query basis. Here, we examine two such approaches.

The first approach essentially tries to ensure that the distance between the mean of the normal distribution and the point at which the densities intersect is large (all distributions are obtained using the mixture model). The idea is that if this distance is large then it will be easier to separate the relevant and non-relevant documents. If the distance is less than a threshold, the engine is discarded for that query. The posterior probability of all engines selected (i.e. not discarded) for a particular query are averaged to obtain document probabilities as before. If all engines for

a particular query are below the threshold, then the one with the highest posterior probability is selected. The threshold is selected based on empirical data to be 0.16. The results for this technique are labelled as META206 in Figure 13.

The second approach uses the posterior probabilities obtained using Bayes' rule from the mixture components. In some situations the maximum of the posterior probability is quite small. A posterior of 0.5 indicates that the relevant and non-relevant distributions weighted by the priors are of equal magnitude. In other words a posterior of 0.5 indicates the point at which the exponential and Gaussian densities intersect after weighting by the prior. It is clear that a "good" engine should preferably have a higher posterior. Empirically if the posterior for a particular engine and a particular query was less than 0.7 then that engine was regarded as poor and discarded for that particular query. If both engines had maximum posteriors greater than 0.7 then they were averaged. If neither engine had a maximum posterior greater than 0.7 both were again averaged and combined - this is plotted as META207.

Figure 13 shows the results of combining two engines whose performance is very different. We again use inq101 and xerox4. As is clear from Figure 13, META206 and META207 perform about equally well and and both are better than META200 (straight averaging of posterior probabilities). The average precisions of META206 and META207 are essentially the same as LEE's. We have also carried out other experiments with other engines all of which demonstrate that engine selection can be done using the models of score distributions discussed here. We note that both META206 and META207 are still worse than using INQUERY alone indicating that there is further scope for improving the engine selection procedure. Of course, this also implies that when one search engine performs much worse than another it may be best not to use the "poor" search engine.

5.2 Discussion of Combination Results

The results above show that the mixture modeling performs as well as the best current techniques (Lee's) available for combination. Although there is scope for a slight improvement it is also clear that we are approaching the limits of the best performance we can achieve.

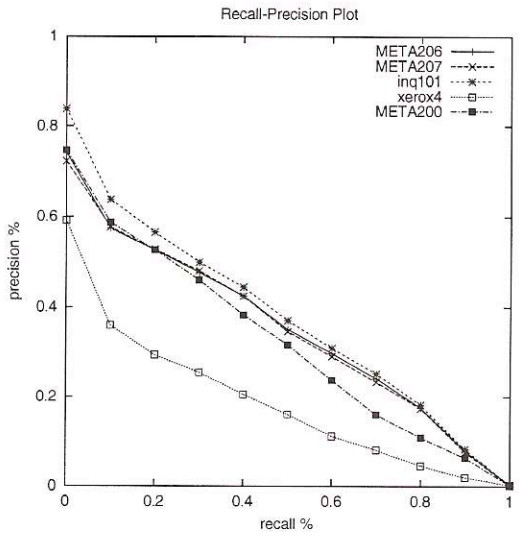

Figure 13: Recall precision graphs for combining inq101 and xerox4 by automatically selecting the best engine using probabilities and distributions.

Lee's COMBMNZ technique performs surprisingly well. In the case where INQUERY and SMART are combined we note that for many queries INQUERY and SMART have distributions which are very similar. In such a situation, their posterior distributions will also look remarkably similar and hence averaging is a good strategy. Since COMBSUM involves computing a document score by just adding the scores for all engines which find that document, it will produce the same ranking as averaging and hence it will also be good. COMBMNZ involves multiplying COMBSUM by the number of engines which found that document and hence it will also produce good results. In this particular situation COMBMNZ involves essentially combining the posterior probabilities without having to do the mixture modeling. However, in the more general case, the good performance of COMBMNZ is hard to explain.

Lee argued that COMBMNZ worked because different search engines are more likely to return the same relevant documents while the non-relevant documents returned are more likely to be random. In other words, COMBMNZ exploits the correlation between relevant documents returned by different search engines. The model for combination proposed here involving the averaging of probabilities from different search engines does not explicitly exploit the correlation across search

engines (recall that averaging probabilities is optimal when we know that the search engines are independent). Inspite of this, it still performs as well as COMBMNZ. The model of combination is also intuitively satisfying for a number of reasons. First, it combines engines in a natural way using probabilities and is therefore easier to explain. Second, it indicates where improvements can be made for better performance. Third, the same technique may be used for combining multi-lingual engines. It will also extend to multi-modal engines provided the distributions of scores behave in a similar way for search engines indexing other media.

6 Future Work and Other Applications of Score Modeling

The model for score distributions leads to a number of interesting questions and possible future applications and extensions of the technique.

The first interesting point to note is the large number of different search engines which seem to have similar score distributions. It is not obvious that this should be the case. For example, there are a large class of functions which could change the score distribution without changing the actual document rankings. This leads us to believe that the form of the score distributions is intimately related to the distributions of terms (words) in languages. We have intuitively justified the form of the score distributions from a two Poisson model for word distributions. However, a more formal derivation from either a Poisson model or a more accurate model of word distributions (say a negative binomial distribution) would be useful.

The assumption that the relevant and non-relevant distributions are given by a Gaussian and an exponential are only approximations. There are probably better models around and it would be interesting to find these. The use of any such models must take into account whether they can be recovered from a mixture when relevance information is not available.

We note that the distributions obtained from the EM based mixture models are not identical to those obtained when complete relevance information is available. Further, the results of combining search engines using the EM based mixture model are also not as accurate as those obtained when

complete relevance information is available. This suggests that there is room for improvement in the EM based mixture modeling. It also suggests the possibility of using partial relevance information (as in relevance feedback) to improve the mixture modeling. In fact, relevance feedback may fit naturally into this model.

In the case of combination, one surprising result is that Lee's technique (based on normalizing and weighted averaging of the scores) performs as well as it does. Lee's technique was proposed on heuristic grounds. We note that Lee's technique explicitly exploits correlations between different engines (that is, the fact that the same document appears across multiple engines) while in the technique proposed here this correlation is not exploited.

The score modeling work can be applied to a number of different areas besides the meta-searching of a single database proposed here. An obvious extension is to the problem of combining results from multiple search engines which may operate on different databases. It is not completely obvious that the results will improve in this case. For example, one of the explanations provided for the efficacy of Lee's technique is that it tends to promote those documents which are returned by multiple search engines. When the databases are different, the same document is unlikely to be present in multiple databases and hence the efficacy of the proposed combination technique in such cases is unknown.

Other extensions could include the application to one or more databases in different languages. Multi-lingual retrieval often involves using the same query in multiple languages to search databases in different languages. A common problem in such situations is how the ranks output for search engines operating on different languages should be combined and there does not seem to be solution which is satisfactory. Score modeling could allow us to convert the scores of each search engine to a probability and then combine the probabilities. The probabilities can be combined in a more natural way We hope to do some experiments in the near future to test this hypothesis.

An interesting situation arises in the case of multiple databases. In some cases, some databases may have few or no relevant documents. In such cases a relevant distribution cannot be estimated and hence there may be problems in applying the mixture model. Note that this is unlikely to

occur in the situations where all engines operate on a single database - in fact for engines indexing a single database, we have never seen this happen.

Score modeling has also been applied to the filtering task. Arampatzis et al [2, 1] independently approximated the relevant documents using a Gaussian and the non-relevant documents using an exponential. They applied the model to finding a threshold for the information filtering task. They did not use a mixture model. Since the filtering task provides relevance information on a subset of the data, this relevance information was used to estimate the Gaussian and exponential distributions separately. Zhang and Callan [29] also used score modeling for filtering. They computed an error function and solved it using gradient descent.

Another interesting question that comes to mind is whether these models also fit search engines operating on other media. This needs to be tested. If true, it could lead to one way of combining multimedia engines for multi-modal retrieval.

7 Conclusion

We have demonstrated how to model the score distributions of a number of text search engines. Specifically, it was shown empirically that the score distributions on a per query basis may be fitted using an exponential distribution for the set of non-relevant documents and a normal distribution for the set of relevant documents.

It was then shown that given a query for which relevance information is not available, a mixture model consisting of an exponential and a normal distribution may be fitted to the score distribution. These distributions were used to map the scores of a search engine to probabilities.

The model of score distributions was used to combine the results from different search engines to produce a meta-search engine. The results were substantially better than either search engine provided no "search engine" performed really poorly. Different combination techniques were proposed including averaging the posterior probabilities of the different engines as well as using the probabilities and distributions to selectively discard some engines on a per query basis.

Future work will include attempts to further improve the modeling for better performance. Other possible applications of modeling score distributions like choosing thresholds for topic detection and tracking will also be examined. Finally we will also examine the possibility that search engines indexing other media like images can also be modeled in the same way.

8 Acknowledgements

The origins of this work can be traced back to a discussion at the University of Glasgow. The first author would like to thank Bruce Croft for extensive discussions and also for pointing out relevant literature especially previous literature on score modeling in information retrieval.

This material is based on work supported in part by the CIIR, in part by the National Science Foundation under grant numbers IRI-9619117 and IIS-9909073, and in part by SPAWARSYSCEN-SD grant number N66001-99-1-8912. Any opinions, findings and conclusions or recommendations expressed in this material are the author(s) and do not necessarily reflect those of the sponsor(s).

A Appendix I: Derivation of the Maximum Likelihood Solution to the Exponential-Gaussian Mixture using Expectation Maximization

We derive here the maximum likelihood solution to the exponential-Gaussian mixture using the Expectation Maximization algorithm. We will base our technique on the one used in [6] to solve for a mixture of Gaussians. The expectation of the negative log-likelihood for a mixture model is given by equation 5 which we reproduce below:

$$E[E^{comp}] = -\sum_{n=1}^{N}\sum_{j=1}^{M} P^{old}(j|x^n) ln\{P^{new}(j)p^{new}(x^n|j)\} \tag{12}$$

where

$$P(j|x) = \frac{p(x|j)P(j)}{p(x)}$$

Here, current values are denoted by the superscript old and the new values of the parameters are denoted by the superscript new. We will now derive the update equations for the Expectation Maximization procedure.

Expanding equation 12 by substituting the expressions for the exponential density (equation 3) and the Gaussian density (equation 4) for $p^{new}(x^n|j)$ we obtain the following equation,

$$E[E^{comp}] = -\sum_{n=1}^{N}[P^{old}(1|x^n)\{ln[P^{new}(1)] + ln\lambda^{new} - \lambda^{new}x^n)\} \quad (13)$$

$$+ \; P^{old}(2|x^n)\{ln[P^{new}(2)] - ln[\sqrt{2\pi}\sigma^{new}] - \frac{(x^n - \mu^{new})^2}{2(\sigma^{new})^2}\}] \quad (14)$$

The M-step involves maximizing the above expression. The maximization is done by differentiating the expectation with respect to the parameters and setting the derivatives to zero. Thus, differentiating with respect to λ^{new} and setting the result to zero we have:

$$\frac{\partial E[E^{comp}]}{\partial \lambda^{new}} = -\sum_{n=1}^{N}\left[P^{old}(1|x^n)(\frac{1}{\lambda^{new}} - x^n)\right] = 0$$

The solution to this equation is clearly

$$\lambda^{new} = \frac{\sum_n P^{old}(1|x^n)}{\sum_n P^{old}(1|x^n)x^n}$$

In a similar fashion by differentiating with respect to the mean of the Gaussian μ^{new} and setting the result to zero we obtain:

$$\frac{\partial E[E^{comp}]}{\partial \mu^{new}} = -\sum_{n=1}^{N}\left[P^{old}(2|x^n)\frac{x^n - \mu^{new}}{(\sigma^{new})^2}\right] = 0$$

which gives the following update equation for μ^{new}

$$\mu^{new} = \frac{\sum_n P^{old}(2|x^n)x^n}{\sum_n P^{old}(2|x^n)}$$

Differentiating with respect to σ^{new} and again setting the result to zero we obtain:

$$\frac{\partial E[E^{comp}]}{\partial \sigma^{new}} = -\sum_{n=1}^{N}\left[P^{old}(2|x^n)\{-\frac{1}{\sigma^{new}} + \frac{(x^n - \mu^{new})^2}{(\sigma^{new})^3}\}\right] = 0$$

$$(\sigma^{new})^2 = \frac{\sum_n P^{old}(2|x^n)(x^n - \mu^{new})^2}{\sum_n P^{old}(2|x^n)}$$

Finally, to obtain the mixing parameters we need to enforce the constraint that the mixing parameters must sum to 1. That is, $\sum_j P^{new}(j) = 1$. This can be done using a Lagrange multiplier k and minimizing the function

$$E[E^{comp}] + k\left(\sum_j P^{new}(j) = 1\right)$$

Setting the derivative of the above function with respect to $P^{new}(j)$ to zero we obtain the solution for the mixing parameter as:

$$P^{new}(j) = \frac{1}{N}\sum_n P^{old}(j|x)$$

In other words the mixing parameter is estimated by averaging the posterior function for that component over all points.

B Appendix II: Derivation of the Maximum Likelihood Solution to the Exponential-Gamma Mixture using Expectation Maximization

We follow the same steps as were used to derive the solution for the exponential-Gaussian mixture in Appendix I. Expanding equation 12 by substituting the expressions for the exponential density (equation 3) and the Gamma density (equation 3.2.1) for $p^{new}(x^n|j)$ we obtain the following equation,

$$\begin{aligned} E[E^{comp}] &= -\sum_{n=1}^{N}[P^{old}(1|x^n)\{ln[P^{new}(1)] + ln\lambda^{new} - \lambda^{new}x^n)\} \\ &+ P^{old}(2|x^n)\{ln[P^{new}(2)] - kln(\theta^{new}) - ln[\Gamma(k^{new})] + (k-1)ln(x^n) - x^n + \theta^{new}\}] \end{aligned}$$

The M-step involves maximizing the above expression. The solution for the exponential parameter is the same as in Appendix I. By differentiating with respect to the Gamma parameter θ^{new} we have:

$$\frac{\partial E[E^{comp}]}{\partial \theta^{new}} = -\sum_{n=1}^{N} P^{old}(2|x^n)\{\frac{-k}{\theta} + \frac{x^n}{\theta^2}\} = 0$$

which gives us the following update equation for θ^{new}

$$\theta^{new} = \frac{\sum_n P^{old}(1|x^n)x^n}{k^{new}\sum_n P^{old}(1|x^n)} \tag{15}$$

Note that this depends on k^{new}. This implies that k^{new} must be computed before θ^{new}.

We now compute the update equations for k^{new}. Minimizing with respect to k by differentiating with respect to k^{new} and setting the result to zero we have:

$$\frac{\partial E[E^{comp}]}{\partial k^{new}} = -\sum_{n=1}^{N} P^{old}(2|x^n)\{-ln(\theta^{new}) - \frac{1}{\Gamma(k)}\frac{d\Gamma(k)}{dk} + ln(x^n)\} = 0$$

We now substitute for θ^{new} from equation 15 and also replace the expression $\frac{1}{\Gamma(k)}\frac{d\Gamma(k)}{dk}$ by $\psi(k)$.

This gives us the following equation:

$$-\sum_{n=1}^{N} P^{old}(2|x^n)\{-ln[\sum_s P^{old}(2|x^s)x^s] + ln[k^{new}\sum_s P^{old}(2|x^s)] - \psi(k) + ln(x^n)\} = 0$$

The above equation may be simplified and rewritten as

$$lnk - \psi(k) = \frac{\sum_n P^{old}(2|x^n) ln \frac{\sum_s P^{old}(2|x^s)x^s}{x^n \sum_s P^{old}(2|x^s)}}{\sum_n P^{old}(2|x^n)}$$

For convenience we will write $\eta = lnk - \psi(k)$. To determine k, we use the approximation in Greenwood and Durand [13] to obtain

$$k^{new} = \begin{cases} \frac{0.5000876+0.1648852\eta-0.0544274\eta^2}{\eta} & 0 \leq \eta \leq 0.5772 \\ \frac{8.898919+9.059950\eta+0.9775373\eta^2}{\eta(17.79728+11.968477\eta+\eta^2)} & 0.5772 \leq \eta \leq 17.0 \\ \frac{1}{\eta} & 17 \leq \eta \end{cases}$$

Given k^{new} the update equation for θ^{new} can be obtained from equation 15.

References

[1] A. Arampatzis, J. Beney, C. H. A. Koster, and T. P. van der Weide. Incrementality, half-life and threshold optimization for adaptive document filtering. In *Proc. of the 9th Text Retrieval Conference (TREC-9)*, pages 589–600, Oct. 2001.

[2] A. Arampatzis and A. van Hameren. Maximum likelihood estimation for filtering thresholds. In *In the Proc. of the 24th ACM SIGIR conf. on Research and Developement in Information Retrieval*, pages 285–293, Sept 2001.

[3] J. Aslam and M. Montague. Models for metasearch. In *In the Proc. of the 24th ACM SIGIR conf. on Research and Developement in Information Retrieval*, pages 276–284, Sept 2001.

[4] J. A. Aslam, , and M. Montague. Bayes optimal metasearch: A probabilistic model for combining the results of multiple retrieval systems. In *the Proc. of the 23rd ACM SIGIR conf. on Research and Developement in Information Retrieval*, pages 379–381, 2000.

[5] C. Baumgarten. A probabilistic solution to the selection and fusion problem in distributed information retrieval. In *In the Proc. of the 22nd ACM SIGIR conf. on Research and Developement in Information Retrieval*, pages 246–253, Aug 1999.

[6] C. M. Bishop. *Neural Networks for Pattern Recognition*. Oxford University Press, 1995.

[7] A. Bookstein. When the most Pertinent document should not be retrieved - an analysis of the Swets model. *Information Processing and Management*, 13:377–383, 1977.

[8] J. Callan, Z. Lu, and W. B. Croft. TREC and TIPSTER experiments with INQUERY. In *the Proc. of the 18th ACM SIGIR conf. on Research and Developement in Information Retrieval*, pages 21–28, 1995.

[9] K. W. Church and W. A. Gale. Poisson mixtures. *Natural Language Engineering*, 1(2):163–190, 1995.

[10] W. B. Croft. Combining approaches to information retrieval. In W. B. Croft, editor, *Advances in Information Retrieval*, pages 1–36. Kluwer Academic Publishers, 2000.

[11] M. Flickner, H. S. Sawhney, J. Ashley, Q. Huang, B. Dom, M. Gorkani, J. Hafner, D. Lee, D. Petkovic, D. Steele, and P. Yanker. Query by image and video content: The QBIC system. *IEEE Computer Magazine*, 28(9):23–30, Sept. 1995.

[12] E. Fox and J. Shaw. Combination of multiple searches. In *the Proc. of the 2nd Text Retrieval Conference (TREC-2)*, pages 243–252. National Institute of Standards and Technology Special Publications 500-215, 1994.

[13] J. A. Greenwood and D. Durand. Aids for fitting the gamma distribution by maximum likelihood. *Technometrics*, 2(1):55–65, 1960.

[14] W. Greiff. The use of exploratory data analysis in information retrieval research. In W. B. Croft, editor, *Advances in Information Retrieval*, pages 37–72. Kluwer Academic Publishers, 2000.

[15] S. P. Harter. A probabilistic approach to automatic keyword indexing. *Journal of the American Society for Information Science*, 20:197–206, 1975.

[16] V. Lavrenko and W. B. Croft. Relevance based language models. In *In the Proc. of the 24th ACM SIGIR conf. on Research and Developement in Information Retrieval*, pages 120–127, Sept 2001.

[17] J. H. Lee. Combining multiple evidence form different properties of weighting schemes. In *the Proc. of the 18th Intl. Conf. on Research and Development in Information Retrieval (SIGIR'95)*, pages 180–188, 1995.

[18] J. H. Lee. Analyses of multiple evidence combination. In *the Proc. of the 20th Intl. Conf. on Research and Development in Information Retrieval (SIGIR'97)*, pages 267–276, 1997.

[19] R. Manmatha, T. Rath, and F. Feng. Modeling score distributions for combining the outputs of search engines. In *the Proc. of the 24th ACM SIGIR conf. on Research and Developement in Information Retrieval*, pages 267–275, Sept 2001.

[20] G. McLachlan and D. Peel. *Finite Mixture Models*. John Wiley, 2000.

[21] F. Mosteller and D. Wallace. *Inference and Disputed Authorship: The Federalist*. Addison Weseley, 1964.

[22] S. E. Robertson and S. Walker. Some simple effective approximations to the 2-poisson model for probabilistic weighted retrieval. In *the Proc. of the 17th ACM SIGIR conf. on Research and Developement in Information Retrieval*, pages 232–241, 1994.

[23] J. A. Swets. Information retrieval systems. *Science*, 141:245–250, 1963.

[24] K. Tumer and J. Ghosh. Linear and order statistics combiners for pattern clasification. In A. Sharkey, editor, *Combining Artificial Neural Networks*, pages 127–162. Springer-Verlag, 1999.

[25] C. J. van Rijsbergen. *Information Retrieval*. Butterworths, 1979.

[26] C. Vogt and G. Cottrell. Predicting the performance of linearly combined IR systems. In *the Proc. of the 21st ACM SIGIR conf. on Research and Developement in Information Retrieval*, pages 190–196, 1998.

[27] C. Vogt and G. Cottrell. Fusion via a linear combination of scores. In *Information Retrieval*, pages 1–22, 1999.

[28] E. Voorhees, N. Gupta, and B. Johnson-Laird. Learning collection fusion strategies. In *the Proc. of the 18th ACM SIGIR conf. on Research and Developement in Information Retrieval*, pages 172–179, 1995.

[29] Y. Zhang and J. Callan. Maximum likelihood estimation for filtering thresholds. In *In the Proc. of the 24th ACM SIGIR conf. on Research and Developement in Information Retrieval*, pages 294–302, Sept 2001.