Towards a better understanding of language model information retrieval

M. van der Heijden
Radboud University Nijmegen
m.vanderheijden@gmail.com

I.G. Sprinkhuizen-Kuyper
Radboud University Nijmegen
Donders Institute for Brain Cognition and Behavior

Th.P. van der Weide
Radboud University Nijmegen
Institute for Computing and Information Science

Abstract

Language models form a class of successful probabilistic models in information retrieval. However, knowledge of why some methods perform better than others in a particular situation remains limited. In this study we analyze what language model factors influence information retrieval performance. Starting from popular smoothing methods we review what data features have been used. Document length and a measure of document word distribution turned out to be the important factors, in addition to a distinction in estimating the probability of seen and unseen words. We propose a class of parameter-free smoothing methods, of which multiple specific instances are possible. Instead of parameter tuning however, an analysis of data features should be used to decide upon a specific method. Finally, we discuss some initial experiments.

1. INTRODUCTION

Since the end of the last decade language models have caught on as a successful technique for information retrieval. Language models are a specific form of probabilistic models. These models have shown performance similar to vector space models [8], but with more theoretical background compared to the heuristics of vector space [12]. By using explicit models of the query and the documents the representation can be made more precise. This should lead to better performance than a 'bag of words' approach.

In language modeling the most straightforward approach, query likelihood, estimates the relevance of a document by computing the probability of generating the query from the document (e.g. [8]). An alternative and popular approach [4, 10] assumes that the query and documents are each generated from a probability distribution. Similarity is computed by comparing the query and document distributions using relative entropy. Smoothing the document distribution with collection data decreases the influence of data sparseness and query imperfections (see Section 3).

Simple unigram language models assume word independence, although taking preceding words into account improves performance [8]. However, word position is not necessarily a good indication of word dependency. More sophisticated context analysis like non-adjacent word dependency [3] and word relations extracted from external sources [1], has therefore proven interesting. In addition other extensions to language models have been developed, especially probabilistic query expansion techniques [4, 5]. Also document smoothing with a cluster of similar documents has been proposed [6, 9]. These clusters decrease data sparseness because they contain more alternative representations of concepts (e.g. synonyms).

The goal of this study is to understand the mechanics of language modeling and the interactions between various techniques. Although some methods usually perform better than others, for specific situations it is not always clear what method would give the best performance. This results in ad hoc parameter tuning or brute force trial and error of alternative techniques. It is desirable to have better insight in what properties of data and language modeling techniques influence retrieval performance and how these properties interact.

In this study we hope to shed some light on the conditions for optimal performance. In order to do so we will first give an overview in unified notation of language modeling methods from literature. Then in Section 3 we will look specifically at smoothing techniques and the factors that influence parameter settings. New parameter-free smoothing methods will be introduced in Section 4. These are based on the analysis of existing techniques, in an attempt to prevent ad-hoc parameter tuning. We have done some preliminary testing of our work, which can be found in

Sections 5 and 6. From these analyses and experiments we will try to derive future directions for a better theoretical understanding in order to increase language model retrieval performance.

2. A LANGUAGE MODELING FRAMEWORK

In order to find relevant documents for a query, model similarity will be used. The query is represented by a probabilistic model θ_{qorg} and each document by a document model θ_{dorg}. To calculate the similarity between these probabilistic models a similarity measure is needed. A popular measure is relative entropy - also known as Kullback-Leibler divergence. In general, for two (discrete) probability distributions P and Q, relative entropy is written as:

$$D(P,Q) = \sum_x P(x) \log \frac{P(x)}{Q(x)}$$

Because this measure is not symmetric there are two possible choices for P and Q. We use the query model as P and the document model as Q. The query and document models can be expanded to overcome data sparseness or to add information not present in the original models. These additions are modeled as a smoothing model θ_{qs} for the query and a smoothing model θ_{ds} for a document. Computing the similarity between the models requires interpolating the original and smoothing models, for which we introduce a query interpolation-factor λ_q and a document interpolation-factor λ_d. The similarity between a query and document can now be written as the divergence between the smoothed query and document models:

$$D(\theta_q, \theta_d) = D((1-\lambda_q)\theta_{qorg} + \lambda_q\theta_{qs}, (1-\lambda_d)\theta_{dorg} + \lambda_d\theta_{ds}) \tag{2.1}$$

In order to incorporate the possibility of word dependence in the models we introduce a link model L. Simple language models that assume word independence will have $L = \emptyset$. The query and document models are probabilistic models, so the relative entropy can be expanded to probabilities. The model divergence can be written in terms of the probability of a word w given word dependence L and the query model θ_q or document model θ_d:

$$D(\theta_q, \theta_d) = -\sum_{w \in V} P(w|L,\theta_q)P(L|\theta_q)P(\theta_q) \log \left(P(w|L,\theta_d)P(L|\theta_d)P(\theta_d)\right) \tag{2.2}$$

The query term in the log has been dropped since it is constant for all documents and thus does nog influence ranking. Usually the priors are also disregarded because they depend on external factors (e.g. user, author). These are difficult to take into account, especially in a lab setting.

Assuming word independence is unrealistic (e.g. 'White House'), therefore links between words are considered, represented by our link model L. The simple unigram model, also called query-likelihood model, can be recovered from relative entropy when $L = \emptyset$. An estimate of the query model θ_q using only the original query (the interpolation-factor λ_q = 0) then leads to the probability:

$$P(w|\emptyset,\theta_q) = \frac{1}{|q|}\sum_{t \in q} \delta(w,t)$$

where $\delta(w,t)$ is the indicator function which is 1 when $w = t$ and 0 otherwise. Substituting in Equation (2.2) (dropping priors and with $L = \emptyset$) we regain the negative log-likelihood of the simple unigram model [4].

Other N-gram models do take links between words into account and thus have a nonempty L. The link between words is static however (e.g. bigram models take only the previous word as link) and as a consequence the probability $P(L|\theta_q)$ equals 1. $P(w|L,\theta_q)$ can then be written as $P(w|w_{-1}\cdots w_{-(N-1)},\theta_q)$. More complex links with non-static probabilities are possible, but they are not further considered here.

Query expansion can be considered a form of query smoothing. Information is added to the query in an attempt to overcome query incompleteness. The factor λ_q is a combining weight for the original query and the smoothing model. Whether a word w is a good candidate for expansion can be estimated by calculating the probability of observing that word given the query q and a set of documents D (assumed) relevant to the query [4, 5]:

$$P(w|q) = P(w)\prod_{t \in q}\sum_{d \in D(q)} P(t|d)P(d|w) \tag{2.3}$$

For each query term t the evidence of a relation between t and w is estimated by multiplying the probability of encountering a document $d \in D(q)$ given w and the probability of the query term t given the document d. By summing over all documents this probability indicates whether the candidate word fits the query and can be used for query expansion. The query smoothing model can now be constructed by taking the top ranked words, resulting from computing this expansion probability for all words in D. However, the expansion model can also be constructed by retrieving synonyms, hypernyms and/or hyponyms from for instance Wordnet.

Another approach to query modeling is the so called relevance model [5]. Instead of using only the query words, a model of all relevant words would perform better. When no training data is available however, the query is still the best estimate of relevance. The difference with the aforementioned query expansion method is that the smoothing model is the probability distribution of $P(w|q)$ (Equation (2.3)) over the whole vocabulary of D, instead of a limited set of expansion words.

3. SMOOTHING ESTIMATES

One of the factors influencing language model performance is the smoothing strategy used to cope with data sparseness. When estimating the probability of encountering a query term in a document, using normalized frequency will lead to zero probabilities for words not present in the document. This is clearly undesirable, as the absence of one term does not mean the document is completely irrelevant. Smoothing the document estimate with some background distribution can effectively solve the data sparseness problem. The smoothed probability estimate of a word w can be written as an interpolation of the document probability and the background probability:

$$(1 - \lambda)P(w|d) + \lambda P(w|C) \tag{3.1}$$

where λ is an interpolation factor (λ_d in Equation (2.1)), d is the document and C the collection. For small λ this means the estimate is dominated by the count of the word in the document, while for larger λ the estimate is closer to the collection probability, so smoothing increases with increasing λ.

A myriad of methods has been proposed over the years and for every specific situation the optimal technique can be sought. Chen and Goodman [2] compared a number of methods in the context of speech recognition, resulting in a somewhat more formal understanding of various method's merits. These results do not necessarily carry over to language models in information retrieval, because they are used differently. An overview of often used smoothing techniques in IR is given by Zhai and Lafferty [11], based on the work of [2]. The study compares performance of Jelinek-Mercer, Dirichlet and absolute discount smoothing on various corpora and assesses the influence of the smoothing parameters.

The main question remains what factors determine performance of smoothing techniques. The logical place to start would be to look at existing methods and analyze the parameters and properties. Although finding optimal parameters is usually possible, minimizing the number of parameters should improve the model by making it simpler. Following [11] we will analyze the properties of Jelinek-Mercer, Dirichlet and the absolute discounting methods.

The Jelinek-Mercer (jm) method uses a fixed smoothing parameter λ_{jm}. Obviously, performance is sensitive to the setting of λ_{jm}, which indirectly depends on corpus characteristics and document characteristics like document length.

So, another class of smoothing methods directly uses document length ($|d|$). Smoothing decreases with document length, which is intuitively correct. Because more data is available longer documents will result in better estimates. So for two documents d_1, d_2 the following holds: $|d_1| > |d_2| \rightarrow \lambda(d_1) < \lambda(d_2)$. Dirichlet (dir) is a popular method that fits into this category:

$$\lambda_{dir} = \frac{1}{\frac{1}{\mu}|d| + 1}$$

The parameter μ is the scaling factor that dictates when a document is considered 'long'. For small μ the word probability estimate is dominated by the count in the document. For $\mu \gg |d|$ smoothing will increase with λ_{dir} going to 1. So, μ can be seen as a unity for document length and the overall effectiveness depends on the distribution of document lengths in the corpus. See [7] for effects of μ on the length of the retrieved documents.

To understand absolute discounting (ad) two separate cases should be considered. First, many words will not occur in the document and thus have a probability of zero. All probability estimates of words that do occur in the document are likely overestimated. As a consequence these probabilities should be discounted. Hence the estimate of the probability $P(w|d)$ is:

$$P(w|d) = \begin{cases} \frac{c(w,d)-\delta}{|d|-\delta|V_d|} & \text{if } w \in d \\ 0 & w \notin d \end{cases}$$

The amount of discount is given by δ, in the range $0 \leq \delta \leq 1$. V_d is the vocabulary of the document d, and $c(w, d)$ is the count function counting the number of occurrences of w in d. The smoothed estimate is constructed by redistributing the probability mass subtracted of seen words to unseen words. Because for every unique word in the document some probability has been discounted, λ_{ad} is the ratio between unique words ($|V_d|$) and the length of the document ($|d|$) scaled by δ:

$$\lambda_{ad} = \delta \frac{|V_d|}{|\,d\,|}$$

The parameter δ governs how much probability mass is redistributed to the collection model. Documents that use many different words will be influenced more by the collection whereas documents with a limited vocabulary will be smoothed less. The number of unique words is thus used as a measure for how focused the topic of the document is.

From analyzing these smoothing methods we can try to distill the relevant smoothing factors. The parameter in Jelinek-Mercer smoothing is usually fitted for a specific document collection and set of training queries thus providing only minimal insight in the relevant properties. A comparison of the sensitivity of λ_{jm} when using different collections [11], shows that the optimal value may vary considerably with the collection. Dirichlet and absolute discount smoothing incorporate document length, which is a reasonable addition because shorter documents are more likely to be affected by data sparseness. Dirichlet weights the influence of document length with μ, which is often tuned using training data. Although the optimal setting of μ is also sensitive to the collection used, it is less so than λ_{jm} [11]. Absolute discounting introduces information on the word distribution and differentiates the probability estimates for seen and unseen words.

Zhai and Lafferty note that they have not looked at more sophisticated query modeling techniques which may change the results because of the interaction between the query properties and the smoothing performance. Thus it is interesting to establish what the properties of an optimal smoothing method should be when dedicated query modeling is used. Lavrenko and Croft [5] proposed an explicit relevance model to capture the topic of the query, but Lafferty and Zhai [4] have also worked on more specific query modeling. Section 5 contains some initial work in this direction.

Document length, word distribution and the distinction between seen and unseen words are the main factors used in the smoothing methods we looked at. The next section will introduce some alternative smoothing methods, using these features as a basis in combination with some new ideas. The last part of this article will report the experiments and preliminary results.

4. NEW SMOOTHING METHODS

A disadvantage of the smothing methods introduced above is that all need parameter tuning. Here we introduce some new smoothing methods which do not need parameter tuning. The main reason to use smoothing methods is to overcome data sparseness. As this is a property of the data, we may be able to use information from the data directly to find the right smoothing parameter. We propose a class of smoothing methods based on word probabilities. However, to prevent introducing a word dependent parameter the information will be generalized over the vocabulary. The class of methods can be described by the function:

$$\lambda(x) = \frac{1}{|V|} \sum_{w \in V} \frac{1}{1 + x(w)} \tag{4.1}$$

with x a function from words to the non-negative numbers. This equation gives a smoothing factor (in the range $(0, 1]$) that depends only on the document and collection (cf. Section 3). The rest of this chapter will give some smoothing strategies that fit within this class.

The ratio between the probability of a word in the document and in the collection indicates the typicality of the word. We will call this ratio $\alpha(w)$:

$$\alpha(w) = P(w|d)/P(w|C) \tag{4.2}$$

$\lambda(\alpha)$ gives us a measure of the centrality of d in C. When using $\lambda(\alpha)$ as smoothing method, smoothing will increase when $\alpha(w) = 0$. As documents with a small vocabulary have many words for which $\alpha(w) = 0$, this should alleviate data sparseness.

The Dirichlet method takes document length into account, but is tuned with a free parameter μ. We will attempt to utilize information in the data to replace the parameter. We can use the same concept as in Eq. (4.2) but scale the probability in the document with a document length factor:

$$\beta(w) = P(w|d)/\rho \tag{4.3}$$

where ρ is the relative document length (i.e. scaled to the range [0,1]): $\rho = \frac{\text{rank}(d)}{|C|}$. The longest document has rank 1, the shortest document rank $|C|$ (documents with equal length will get the same rank). When a word w does not occur in the document $\beta(w) = 0$, but in all other cases β is weighted with the document length. The smoothing method $\lambda(\beta)$ increases smoothing when data is sparse and takes document length into account. Long documents will thus be smoothed less than short documents, as in Dirichlet smoothing.

Smoothing with $\lambda(\beta)$ uses $P(w|d)$ as a measure of data sparseness but no longer compares this probability with the collection probability. In order to reintroduce this information α and β should be combined. This leads to a probability ratio scaled by document length:

$$\gamma(w) = P(w|d)/(\rho P(w|C)) \tag{4.4}$$

In smoothing with $\lambda(\gamma)$ the collection data will be used when $P(w|d)$ is zero, in all other cases scaling with the collection data and document length occurs. Hence, short documents with atypical word probabilities compared to the collection will be smoothed most.

The rationale of introducing the factor α was that the difference between the document and collection probabilities gives information on the amount of smoothing needed. However a consequence of choosing the ratio is that for each word not in the document smoothing increases ($\alpha(w) = 0$ thus the contribution to $\lambda(\alpha)$ for that particular word is 1). Because the collection will have many more words than any single document, this may introduce quite some noise. Instead we could look at a different measure that also indicates the discrepancy between document and collection probabilities:

$$\delta(w) = \frac{1 - |P(w|d) - P(w|C)|}{\rho} \tag{4.5}$$

with ρ again the relative document length. Because smoothing is needed when the absolute difference between the document and collection probabilities is large, we take one minus the difference.

5. EXPERIMENTS

We performed some initial experiments on smoothing and query expansion. The goal of these experiments is to get more insight into the factors influencing the retrieval results, in order to predict what we can expect when using the new smoothing methods introduced in Section 4.

The data set comprised AP88-89, WSJ87-92, ZIFF1-2 on TREC-disks 1&2. This standard data set allows us to compare our results with previous studies on smoothing, specifically [11]. The corpus consists of slightly less than 470 thousand documents with a total disk size of approximately 1.5GB. For the language model implementation we used the Lemur toolkit (www.lemurproject.org). The data was preprocessed by stemming with the Porter stemmer. The TREC topics 1 through 150 were used as test queries. Following [11] we used four different kind of queries derived from these topics: short keyword (i.e. title), long keyword (concept field), short verbose (description) and long verbose (title, description and narrative).

A first and rather trivial test is comparing precision/recall (or a related measure, e.g. Mean Average Precision) for smoothed and non-smoothed language model based retrieval. A more interesting test will be gauging the interaction between smoothing and various query expansion techniques. Results obtained using query expansion can be compared to the results without expansion for the mainstream smoothing methods as reported by [11]. This will enable us to test to what extend query expansion is either complementary or comparable to smoothing.

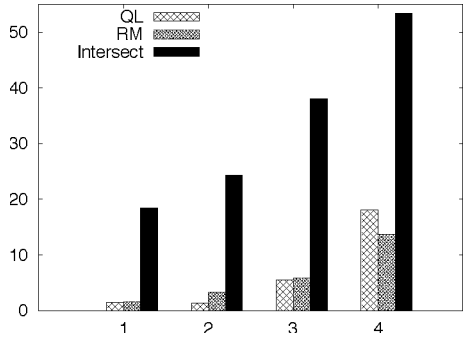

FIGURE 1: Histogram showing the number of relevant documents retrieved (y-axis) only by query likelihood (QL), relevance model (RM) and retrieved by both (Intersect) split on document length parts (x-axis, part 1 shortest documents, part 4 longest documents).

Preliminary testing has shown that smoothing methods retrieve different sets of relevant documents, indicating it may be possible to improve retrieval performance by combining methods. The same applies for retrieval with or without query expansion, indicating that it can indeed be seen as a special case of smoothing. Testing will have to prove whether these differences are significant, and we will thus compare the results of various smoothing methods with or without query expansion on the four types of queries.

We hypothesize that we can gain from using a specific smoothing method or a combination of smoothing and query expansion techniques for each type of collection and query. In particular we expect that collections with shorter documents will benefit more from aggressive smoothing and query expansion; longer queries will require less query expansion, but possibly more smoothing to decrease the influence of uninformative words in the query.

In order to find the determining factors for smoothing we divided the collection in parts based on document length. Although document length is a simple measure to divide the collection, it is only a very coarse grained one. Document length does probably not convey useful information on the document content, which is ultimately what we are interested in. Analogous reasoning can be applied to query length.

Instead of looking at division over length it would be more informative to use similarity between the words in queries or documents. We propose a measure to assess word similarity by intersecting the result sets of both words when used as single word queries, summing over word combinations. This results in a score for the cohesion (σ) within the query or document:

$$\sigma = \frac{2}{N^2 - N} \sum_{i<j}^{N} \text{sim}(w_i, w_j) \quad \text{with} \quad \text{sim}(w_i, w_j) = \frac{|R(w_i) \cap R(w_j)|}{|R(w_i) \cup R(w_j)|} \tag{5.1}$$

and $R(w)$ the result set for a query w.

We can use this measure to calculate cohesion within queries and test our smoothing methods against subsets of queries with similar cohesion. The same can be done with documents, but this becomes computationally very expensive because documents are generally longer than queries and the complexity of the cohesion measure is $\mathcal{O}(N^2)$ expensive retrieval operations. Furthermore, comparing large amounts of words will most likely result in summing over coincidental similarities, thus decreasing the information this measure provides.

6. RESULTS

A first trivial test is showing that smoothing indeed improves performance. As can be seen in Table 1, retrieval with a simple language model without smoothing (Dirichlet $\mu = 0$) has a Mean Average Precision of around zero. The retrieval has been split over four equally sized parts of the document collection ordered by document length, where part 1 contains the shortest documents and part 4 the longest. Longer documents consistently perform better than shorter documents, which could be a result of longer documents using more words to describe a concept thus providing more possible matches. Furthermore the results indicate that performance is not very sensitive to the setting of the Dirichlet smoothing parameter as performance varies only slightly for parameter settings between 500 and 2500.

μ	parts	1	2	3	4
0	map	.0012	.0019	.0044	.0021
	P5	.0134	.0067	.0121	.0053
500	map	.0243	.0325	.0574	.0938
	P5	.2604	.3141	.3705	.4200
1000	map	.0248	.0329	.0590	.0978
	P5	.2671	.3114	.3919	.4293
2500	map	.0247	.0329	.0604	.1013
	P5	.2523	.3047	.4000	.4400

TABLE 1: Mean average precision (map) and precision after 5 documents retrieved (P5) for collection parts (part 1 containing the shortest documents, part 4 the longest documents).

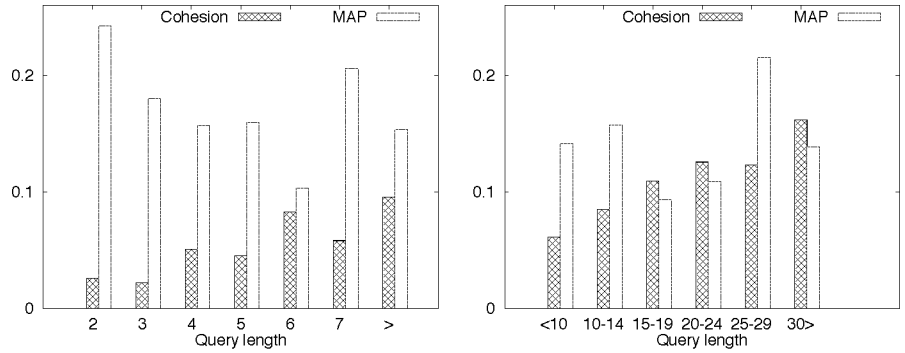

FIGURE 2: Cohesion scores and mean average precision (map), for title (left) and description queries (right), binned by query length (x-axis).

We also tested performance for relevance models compared to query likelihood, on different length documents. Figure 1 shows a histogram of relevant documents retrieved by only the query likelihood method (QL), the relevance model (RM) and the overlap between the two. Both methods are able to retrieve documents the other method did not retrieve. An initial attempt to make a rank based combination of results seemed to perform slightly better but this result was not significant, so further testing remains to be done.

Lavrenko and Croft [5] stated that explicitly modeling relevance can address notions of synonymy and polysemy, but apparently it also introduces some noise. Although retrieval with a relevance model on average performs slightly better than without, the differences are small. For short documents performance of QL and RM is very similar, but retrieval without RM produces better results for the longest documents. A possible explanation would be that the RM indeed introduces synonyms that are useful to match short documents with relatively small vocabularies. For longer documents however, it is more likely that the author already used some synonyms enabling better direct matching. The RM would for these longer documents introduce more noise and concept drift decreasing any performance gain.

We were also interested in testing whether smoothing interacted with using a relevance model. Relevance models add words to the query, therefore smoothing may lead to performance loss due to assigning too much probability to less important words. A comparison of a number of smoothing methods against each other and their respective versions with relevance model can be found in Table 2. Lemur-toolkit defaults have been used for the smoothing parameters (i.e. $\lambda_{jm} = .5$, $\mu_{dir} = 1000$, $\delta_{ad} = .7$). Results are furthermore split for query type as described earlier. For the three smoothing methods tested here, mean average precision (map) increases when using a relevance model. Precision after 5 retrieved documents (P5) shows mixed results however. These results indicate that query smoothing by means of a relevance model and document smoothing do not interact. Additionally we can see that differences between smoothing methods are relatively small, but that Dirichlet slightly outperforms the other methods.

In the previous section we introduced a measure for query cohesion/redundancy, in Figure 2 mean cohesion scores are shown binned on query length. Title and description queries tend to have more cohesion when queries are longer. Longer queries with redundancy should show performance similar to shorter queries, how much performance is lost with increasing redundancy still needs further analysis. This could shed some light on whether the lower retrieval scores of description queries (see Table 2) are caused by a length effect (due to redundancy), or a

Query type		Title		Description		Long keyword		Verbose	
Method		QL	RM	QL	RM	QL	RM	QL	RM
JM	map	.1571	.2082	.1195	.1598	.1991	.2231	.2071	.2168
	P5	.3693	.4360	.3280	.3573	.4765	.4591	.4773	.4707
Dirichlet	map	.1839	.2078	.1365	.1678	.2127	.2317	.2158	.2325
	P5	.4360	.4480	.3947	.3800	.5221	.5168	.5160	.5053
AbsDiscount	map	.1677	.2046	.1164	.1408	.1986	.2236	.1985	.2019
	P5	.4187	.4640	.3547	.3653	.4926	.5034	.4800	.5107

TABLE 2: Comparison of performance for Query likelihood (QL) and Relevance model (RM) retrieval with the smoothing methods as described in Section 3.

consequence of more noise in the query due to sentence filler words that are not present in keyword queries.

7. CONCLUSION

In this study we have analyzed what factors influence language model information retrieval performance. Starting from popular smoothing methods we established what data features were used. Document length and a measure of document word distribution turned out to be the important factors, in addition to a distinction in estimating the probability of seen and unseen words. We used this information as a basis to develop parameter-free smoothing methods. These proposals should enable us to further analyse how performance is influenced by smoothing.

Replicating [11] we show that Dirichlet smoothing is not very sensitive to the setting of its parameter μ. We can also conclude that longer documents can be ranked better than short documents, most likely because they contain more information. We have also looked at the possibility of interaction between using a relevance model [5] for query expansion and smoothing methods. No direct interaction was found, but we did find that retrieval without a relevance model resulted in different relevant documents than retrieval with a relevance model. This needs further analysis. A combination of the results with and without the relevance model seemed to improve performance, but the improvement was not significant. As such we cannot draw any conclusions from our initial experiments. Future work will consist of experiments that further analyze the characteristics of both the data and the retrieval methods used, further improving the overview of the interaction between language modeling methods and retrieval data.

REFERENCES

[1] G. Cao, J.-Y. Nie, and J. Bai. Integrating word relationships into language models. In *Proceedings of the ACM SIGIR Conference '05*, pages 298–305, 2005.

[2] S. F. Chen and J. Goodman. An empirical study of smoothing techniques for language modeling. Technical Report TR-10-98, Harvard University, 1998.

[3] J. Gao, J.-Y. Nie, G. Wu, and G. Cao. Dependence language model for information retrieval. In *Proceedings of the ACM SIGIR Conference '04*, pages 170–177, 2004.

[4] J. Lafferty and C. Zhai. Document language models, query models, and risk minimization for information retrieval. In *Proceedings of the ACM SIGIR Conference '01*, pages 111–119, 2001.

[5] V. Lavrenko and W. B. Croft. Relevance based language models. In *Proceedings of the ACM SIGIR Conference '01*, pages 120–127, 2001.

[6] X. Liu and W. B. Croft. Cluster-based retrieval using language models. In *Proceedings of the ACM SIGIR Conference '04*, pages 186–193, 2004.

[7] D. E. Losada and L. Azzorpardi. An analysis on document length retrieval trends in language modeling smoothing. *Information Retrieval*, 11:109–138, 2008.

[8] D. R. H. Miller, T. Leek, and R. M. Schwartz. A hidden Markov model information retrieval system. In *Proceedings of the ACM SIGIR Conference '99*, pages 214–221, 1999.

[9] T. Tao, X. Wang, Q. Mei, and C. Zhai. Language model information retrieval with document expansion. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics*, pages 407–414, 2006.

[10] C. Zhai and J. Lafferty. Model-based feedback in the language modeling approach to information retrieval. In *CIKM '01: Proceedings of the tenth international conference on Information and knowledge management*, pages 403–410, 2001.

[11] C. Zhai and J. Lafferty. A study of smoothing methods for language models applied to information retrieval. *ACM Trans. Inf. Syst.*, 22(2), 2004.

[12] C. Zhai and J. D. Lafferty. A risk minimization framework for information retrieval. *Inf. Proces. Man.*, 42:31–55, 2006.