A General Optimization Framework for Smoothing Language Models on Graph Structures

Qiaozhu Mei
Department of Computer Science
University of Illinois at Urbana-Champaign
Urbana,IL 61801
qmei2@uiuc.edu

Duo Zhang
Department of Computer Science
University of Illinois at Urbana-Champaign
Urbana,IL 61801
dzhang22@uiuc.edu

ChengXiang Zhai
Department of Computer Science
University of Illinois at Urbana-Champaign
Urbana,IL 61801
czhai@uiuc.edu

ABSTRACT

Recent work on language models for information retrieval has shown that smoothing language models is crucial for achieving good retrieval performance. Many different effective smoothing methods have been proposed, which mostly implement various heuristics to exploit corpus structures. In this paper, we propose a general and unified optimization framework for smoothing language models on graph structures. This framework not only provides a unified formulation of the existing smoothing heuristics, but also serves as a road map for systematically exploring smoothing methods for language models. We follow this road map and derive several different instantiations of the framework. Some of the instantiations lead to novel smoothing methods. Empirical results show that all such instantiations are effective with some outperforming the state of the art smoothing methods.

Categories and Subject Descriptors: H.3.3 [Information Search and Retrieval]: Retrieval Models

General Terms: Algorithms

Keywords: Language modeling, smoothing, graph structure, document and word graph

1. INTRODUCTION

Language models have attracted much attention in the information retrieval community recently due to their success in a variety of retrieval tasks [17, 5]. Fundamental to all the language models used for retrieval is the issue of smoothing, which has been shown to affect retrieval performance significantly [22]. Indeed, in the basic language modeling approaches [17, 7, 16, 24], the entire retrieval problem is essentially reduced to the problem of estimating a document language model which can be further reduced to how to smooth the document language model. In other more sophisticated use of language models, such as relevance models [13] and model-based feedback methods [23], improved smoothing of document language models is also shown to improve performance [14, 20].

Because of the importance of smoothing, it has been attracting attention ever since Ponte and Croft's pioneering work on applying language models to retrieval [17]. Since then many smoothing approaches have been proposed and tested. In early days, most smoothing methods relied on using a background language model, which is typically estimated based on the whole document collection, to smooth a document language model [17, 7, 16, 24]. Recently, corpus graph structures have been exploited to provide more accurate smoothing of document languages. The basic idea is to smooth a document language model with the documents similar to the document under consideration through either clustering or document expansion [14, 9, 10, 6, 18, 11, 20]. Such a *local* smoothing strategy can leverage document similarity structures to offer "customized" smoothing for each individual document; this is in contrast to the simple *global* smoothing strategy which smoothes all documents with the same background model.

The local smoothing strategy has been shown to be quite effective in several recent studies [14, 20, 11] and is the best smoothing strategy known so far. In virtually all these local smoothing methods, the smoothing formula eventually boils down to some form of interpolation of the original unsmoothed document language model (i.e., the maximum likelihood estimate of the unigram language model) and some "supporting" language models estimated based on documents similar to that document. Different smoothing methods differ in how they assign weights to all these different component language models. It is known that this weighting of component language models can significantly affect retrieval performance, and sometimes even a small difference in weighting can lead to visible difference in performance [14]. Unfortunately, none of the existing work offers a formal framework to optimize these weights; as a result, there is no guidance on how to further improve these existing smoothing methods or develop new (potentially better) smoothing methods. For example, it is unclear whether we can exploit other structures such as word similarity graphs to improve smoothing.

In this paper, we propose a general unified optimization framework for smoothing language models on graph structures. While not explicitly stressed in the existing work, we believe that graph structures are fundamental to smoothing;

indeed, we can interpret smoothing intuitively as to make the language models of those nodes close to each other on a graph structure similar to each other, so that the surface representing all the language models would be "smooth."

Our framework unifies two major heuristic goals in smoothing within a single objective function: (1) "Fidelity": A smoothed language model should not deviate too much from the original non-smoothed language model; (2) "Smoothness": After smoothing, the nodes that are close to each other on the graph would have similar (smoothed) language models; the closer the nodes are, the more similar their language models are.

This framework not only provides a principled formulation of the existing smoothing heuristics but also serves as a road map for systematically exploring smoothing methods for language models. We follow this road map and derive several different instantiations of the framework, including smoothing on document graphs and smoothing on word graphs, as well as smoothing document language models and query language models. Although document graphs have been used for smoothing in the previous work, to the best of our knowledge, no previous work has studied how to smooth language models with a word graph. Moreover, existing work on smoothing has mostly attempted to smooth document language models, while we also study how to use a word graph to smooth a query language model.

These smoothing methods are evaluated using several standard TREC test collections. The results show that all the derived smoothing methods can improve over the baseline global smoothing method significantly, indicating that our framework is reasonable. The derived smoothing methods either outperform or perform similarly to the corresponding state of the art local smoothing methods.

Given the importance of smoothing in the language modeling approach to information retrieval and given the generality of our framework, we hope that the framework can open up some promising new directions for finding an optimal way of smoothing language models.

The rest of the paper is organized as follows. In Section 2, we propose the general optimization framework for smoothing language models with graph structure, and introduce a unified solution. In Section 3, we follow the framework and introduce four instantiations of the general framework. We discuss the properties of those instantiations in Section 4 and evaluate them with empirical experiments in Section 5. Finally, we discuss related work and conclude in Section 6 and 7, respectively.

2. SMOOTHING LANGUAGE MODELS WITH GRAPH STRUCTURE

To motivate our framework, we start with a brief discussion of the intuitions behind the major smoothing heuristics.

2.1 Intuitions

Given a non-smoothed document language model $P_{ML}(w|d)$ (i.e., a word distribution), all smoothing methods attempt to generate a smoothed language model $P(w|d)$ that can better represent the topical content of document d. An obvious improvement over $P_{ML}(w|d)$ that almost all the smoothing methods would do is to avoid assigning zero probabilities to words that are not seen in d. Indeed, this was a major motivation for smoothing discussed in [17]. In general, however, the purpose of smoothing is to improve the accuracy of the estimated language model, not just avoiding zero probabilities. The most useful resources for smoothing so far have been documents that are similar to d, and methods exploiting such document graph structures are among the best performing methods [20, 9, 14].

The general procedure of all the smoothing methods using corpus structures is as follows:

1. Construct a graph of documents where documents are connected with edges weighted according to the similarity between them [1].

2. For every document, estimate a structure-based language model based on its nearest neighbors, or a cluster which that the document belongs to.

3. Combine the structure-based language model with the original unsmoothed document language model.

How are these methods different from traditional global methods, such as the Jelinek-Mercer(JM) and the Dirichlet smoothing [24]? Intuitively, they all went beyond the initial goal of giving non-zero probabilities to unseen words. But why do they all take such general steps? What are they essentially trying to optimize? To explain this formally, let us first look at what each step is trying to achieve.

By constructing a similarity graph of documents, one ensures that similar documents are located close to each other. Using the argument of data manifold in machine learning, if we project the documents onto a hyperplane, we expect that documents with the largest similarity have the smallest distance on the hyperplane [1].

By estimating a document language model based on the neighbor documents or the closest clusters, one ensures that the language model representation of a document is not significantly different from the documents close to it.

However, with this objective alone, the smoothed language model could dramatically deviate from the original document. Indeed, an easiest solution is making all documents share the same representation. The combination with original language model ensures that the smoothed language model is not far from its original contents.

Thus we see that there are two major intuitive assumptions made in all such work: "*documents in the same cluster should have similar representation*" [21, 14], and "*neighbor documents should have similar representation*" [9, 20]. Such assumptions also appear in the literature of semi-supervised learning [25, 27], where the first one is referred as "*global consistency*" and the latter as "*local consistency*".

Indeed, if we project the graph structure of documents on a hyperplane, the language model $\{P(w|d)\}$ can be plotted as surfaces on top of the hyperplane.

Figure 1 visualizes such an intuitive explanation with synthetic data. The hyperplane shows a manifold structure of documents and the surface plots $P(w|d)$ for a given word w over different documents. The left figure shows the maximum likelihood estimate of $P(w|d)$, which has an unsmoothed surface; in contrast, the figure on the right side presents a smoothed surface (i.e., smoothed $P(w|d)$) with the basic shape of the surface preserved (i.e., being "loyal" to $P_{ML}(w|d)$).

[1]Although the graph structure is not explicitly mentioned in cluster-based methods [21, 14], this is a reasonable generalization.

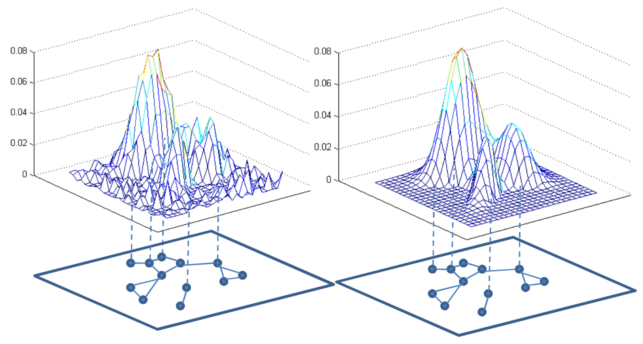

Figure 1: Illustration of smoothing language models on a graph structure. Left figure: unsmoothed model; Right figure: smoothed model

We can now see that graph structures are fundamental to smoothing and a better graph would presumably lead to better smoothing. For example, the simple smoothing strategy of interpolating with the collection language model can be regarded as smoothing on a clearly non-optimal graph where document d is assumed to have equal distance to all other documents in the collection (i.e., a "star" structure). Thus it should not be surprising that smoothing on a graph better reflecting the real semantic similarities between documents would be better, which has indeed been shown in the literature [9, 20].

A major challenge in any smoothing method is how to deal with the tradeoff between "fidelity" (stay close to the non-smoothed model) and "smoothness" (be the same as neighbors). In general, we have such a surface for each word w' ($P(w'|d)$). Thus conceptually we have as many surfaces as the number of words in our vocabulary, and optimizing this tradeoff for all the words can be very difficult without an explicit objective function.

How can we ensure that the tradeoffs for different nodes in the graph are all reasonable? In general, how can we smooth all such surfaces in a principled way and achieve the smoothing effect in Figure 1? The hyperplane in Figure 1 illustrates a manifold of documents. Does that have to be a graph of documents? We can imagine that the hyperplane presents other types of graphs, such as a word graph, and plot the language models as different surfaces on that hyperplane. In the following section, we introduce a general framework to smooth language models on a general graph structure.

2.2 A General Framework

From Figure 1, we see that the notion of *smoothing* discussed in this paper is different from "assigning non-zero probabilities to unseen words," but aims at "achieving consistency on a graph structure." For continuous functions like time series, smoothing is usually done with a regularizer with well-defined derivatives. A graph structure, however, defines a discrete domain. Discrete regularization has been cast as an optimization problem in machine learning literature [26]; we apply it here to define a general optimization framework for smoothing language models.

Formally, let us introduce the following definitions:

- $G = \langle V, E\rangle$: a graph defined in a retrieval problem. $u, v \in V$ are vertices and $(u, v) \in E$ is an edge.
- f_u: a smoothed value-based representation of vertex u in the graph G (e.g., $P(w|d_u)$).
- $\tilde{f}_u$: a non-smoothed (initial) value of u (e.g., $P_{ML}(w|d_u)$).
- $w(u)$: a weight of the importance of vertex u in G.
- $w(u, v)$: a weight of the importance of edge (u, v).

Note that G can be either directed or undirected. In this paper, we only focus on the undirected case, and propose the following general optimization framework for smoothing language models on the graph structure.

$$\begin{aligned} O(C) &= (1-\lambda)\sum_{u\in V} w(u)(f_u - \tilde{f}_u)^2 \\ &+ \lambda \sum_{(u,v)\in E} w(u,v)(f_u - f_v)^2 \qquad (1) \end{aligned}$$

This objective function generalizes the intuitions in Section 2.1 well: The first term guarantees that the smoothed language model does not deviate too much from its original value, especially for more important vertices (controlled by $w(u)$); the second term, also known as a harmonic function in semi-supervised learning, guarantees the consistency of the language model on the graph. The right term can also be written as $\lambda \sum_{u\in V}\sum_{v\in V} w(u, v)(f_u - f_v)^2$ where $w(u, v) = 0$ when there is no edge between u, v.

This framework is general. Clearly, a different selection of $\{G, f_u, \tilde{f}_u, w(u), w(u, v)\}$ leads to a different smoothing strategy. This flexibility provides us a road map for understanding the existing smoothing strategies, as well as exploring novel smoothing methods.

To use such a road map, any reasonable instantiation of the framework must satisfy the following constraints.

1. f_u and f_v are comparable, that is, when f is fully smoothed on G, f_u and f_v should be the same value.
2. $w(u)$ offers a reasonable weighting that captures the importance of vertex u in G.
3. $w(u, v)$ offers a reasonable weighting that captures the importance of edge (u, v) in G.

When we are smoothing language models, we could instantiate f_u as the probability of a word given a document, i.e., $P(w|d)$, and associate u with d, or w, or both. In this case, there is an additional constraint, i.e., $\sum_w P(w|d) = 1$.

What are reasonable instantiations of $w(u)$ and $w(u, v)$? Intuitively, $w(u, v)$ could be instantiated as the closeness of two vertices in the graph, or the similarity of u and v. Using the similarity of two vertices to weight an edge has been commonly adopted in existing literature ([9, 20]). How about $w(u)$? There are also many studies in the context of link analysis to model the importance of a vertex on a graph, e.g., in-degree, PageRank [2], and HITS [8]. In this paper, we use the degree of vertex u as the importance weight of u:

$$w(u) = Deg(u) = \sum_{v\in V} w(u, v) \qquad (2)$$

Minimizing $O(C)$ in Equation 1 will achieve the smoothness of the surface in Figure 1. For instance, if we select $\{G = D, f_u = P(w|d_u), \tilde{f}_u = P_{ML}(w|d_u), w(u, v) = Cosine(d_u, d_v)\}$, the smoothing framework boils down to smoothing language models with document similarity graph, where edges are weighted with cosine similarity. To smooth the language models $\{P(w|d)\}_{d \in C}$, one needs to find a solution of $\{P(w|d_u)\}_{d_u \in C} = \arg\min_{\{f_u\}_{u \in V}} \sum_w O_w(C)$, subject to the constraint $\sum_w P(w|d) = 1$, which achieves the smoothness of multiple surfaces.

2.3 The Smoothing Procedure

Generally, to minimize $O(C)$ in Equation 1, we can compute the first-order partial derivatives of $O(C)$, that is,

$$\frac{\partial O(C)}{\partial f_u} = 2(1-\lambda)Deg(u)(f_u - \tilde{f}_u) + 2\lambda \sum_{v \in V} w(u,v)(f_u - f_v)$$

Let $\frac{\partial O(C)}{\partial f_u} = 0$ and plug in Equation 2, we have

$$f_u = (1-\lambda)\tilde{f}_u + \lambda \sum_{v \in V} \frac{w(u,v)}{Deg(u)} f_v \qquad (3)$$

Clearly, a solution of Equation 3 could minimize $O(C)$ in Equation 1. To get such a solution, we can start with $f_u = \tilde{f}_u$ and execute Equation 3 until converging. In practice, we do not need to wait until complete convergence; a few iterations of Equation 3 can already give an improved $O(C)$. We further discuss leave convergence in Section 4.

3. INSTANTIATIONS ON DOCUMENT AND TERM GRAPHS

In Section 2, we have shown that smoothing document language models with a document graph [14, 9, 20] is an instantiation of the general framework. It is by no means the only one. Indeed, any reasonable instantiation of $\{G, f_u, \tilde{f}_u, w(u), w(u,v)\}$ which satisfies the predefined constraints will lead to a variant strategy of smoothing. In this section, we explore different instantiations of the framework.

3.1 Smoothing with a Document Graph

The most common instantiation is smoothing with document graphs (i.e., $V = D$). Although not in a unified way, many heuristics have been proposed to smooth using document graphs. Document language models are adjusted by receiving weights from the cluster it belongs to [21, 14], or by propagating weights from the nearest neighbors [9, 18, 20, 19]. A commonly used instantiation of $w(u,v)$ is the cosine similarity of two documents. As shown in Section 2.2, a reasonable instantiation of f_u and $\tilde{f}_u$ is $f_u = P(w|d_u)$ and $\tilde{f}_u = P_{ML}(w|d_u)$. Plugging these in Equation 3, we have

Smooth Document Language Models ($f_u = P(w|d_u)$):

$$P(w|d_u) = (1-\lambda)P_{ML}(w|d_u) + \lambda \sum_{v \in V} \frac{w(u,v)}{Deg(u)} P(w|d_v) \quad (4)$$

One may notice that we did not utilize the constraint $\sum_w P(w|d) = 1$. As a matter of fact, when we start with $P(w|d_u) = P_{ML}(w|d_u)$, this constraint is naturally satisfied after every iteration of Equation 4.

Another interesting instantiation of f_u is simply the relevance score of a document d_u for a query q.

Smooth Relevance Scores ($f_u = s(q, d_u)$):

$$s(q,d_u) = (1-\lambda)\tilde{s}(q,d_u) + \lambda \sum_{v \in V} \frac{w(u,v)}{Deg(u)} s(q,d_v) \qquad (5)$$

This process bypasses language models, but smoothes the relevance score directly. Similar heuristics appear in existing work where corpus structure is utilized to rerank documents with score propagation and regularization [10, 6, 18, 11]. In our experiments, we will show that this instantiation is not as effective as smoothing document language models.

3.2 Smoothing with a Word Graph

Recent work in natural language processing has introduced new ways to represent documents with graph structures of words [15]. Indeed, in many scenarios we are accessible to a graph of terms (e.g., a word similarity graph, an entity-relation graph, or an ontology graph). Although most existing work smoothes language models with a document graph, we show that the general smoothing framework can also be instantiated with word graphs, which leads to novel smoothing procedures.

In such smoothing procedures, G is now a word graph (i.e., $V = W$). $w(u,v)$ can be instantiated with either co-occurrence or mutual information of two words. We introduce the following novel instantiations:

Smooth Document Language Models ($f_u = \frac{P(w_u|d)}{Deg(u)}$):

$$P(w_u|d) = (1-\lambda)P_{ML}(w_u|d) + \lambda \sum_{v \in V} \frac{w(u,v)}{Deg(v)} P(w_v|d) \quad (6)$$

Note that unlike a document graph where all documents are treated equally, there is significant prior knowledge of using different words. Some words tend to be used more than others even if they are highly related (e.g., car v.s. vehicle). Some words are more important than others and thus should be assigned a larger value. In this instantiation, we use $f_u = \frac{P(w_u|d)}{Deg(u)}$ rather than $P(w_u|d)$ to make f_u and f_v comparable. Please note that $Deg(u)$ is just one choice of denominator which captures the importance of a word (which is proportional to the pagerank value on G). One could use other choices such as idf. This results in a different denominator from the one used in Equation 4 and Equation 6 ($Deg(u)$ v.s. $Deg(v)$). $\sum_w P(w|d) = 1$ is now naturally satisfied after any iteration of Equation 6. This smoothing strategy is not well studied in existing IR literature. Interestingly, if we use a similar instantiation, $f_u = P(d_u|w)/Deg(u)$ on a document graph based on document similarity, we would be able to derive the term count propagation smoothing method proposed in [19].

The instantiations discussed so far aim at smoothing document language models, where $f_u \propto P(w_u|d)$. When systematically examining variations of the framework, one may naturally ask whether this is the only way to instantiate f_u. In language modeling retrieval models, the query language model is also an important component. Can we also leverage the framework to smooth query language models? The answer is yes. Let $f_u = \frac{P(w_u|q)}{Deg(u)}$, we have

Smooth Query Language Models ($f_u = \frac{P(w_u|q)}{Deg(u)}$):

$$P(w_u|q) = (1-\lambda)P_{ML}(w_u|q) + \lambda \sum_{v \in V} \frac{w(u,v)}{Deg(v)} P(w_v|q) \quad (7)$$

This smoothing method is related to query expansion (i.e., to add new terms to the query and adjust the weights of query words). In model-based feedback [23], where a new query model is estimated from feedback documents, we expect that this novel smoothing strategy can also be applied.

4. DISCUSSION OF THE FRAMEWORK

The objective function of smoothing in Equation 1 is related to existing work of discrete regularization in semi-supervised learning and manifold learning [27, 1, 25, 26]. Many different regularizers and objective functions have been proposed in that context. We use Equation 1 instead of others because it is general, has many nice properties, and has natural connections to many existing concepts and models. Some recent work [6] also used one of them for the problem of retrieval score regularization.

4.1 Connection with Existing Models

Equation 6 and Equation 7 look similar to the updating formula of PageRank [2], except that $P_{ML}(w|d)$ is used instead of the uniformed jumping probability $1/N$. Indeed, Equation 6 can be rewritten as

$$P(w_u|d) = \sum_{v \in V} ((1-\lambda)P_{ML}(w_u|d) + \lambda \frac{w(u,v)}{Deg(v)}) P(w_v|d), \quad (8)$$

which is essentially computing a stationary distribution of a positive-recurrent Markov chain, where

$$p(v \to u) = (1-\lambda)P_{ML}(w_u|d) + \lambda \frac{w(u,v)}{Deg(v)},$$

which is guaranteed to converge. Note that the initial value of u would not be forgotten like in PageRank, because $P_{ML}(w_u|d)$ has been embedded into transition probabilities. Intuitively, we can imagine that when an author is composing a document, he would randomly walk along such a Markov chain of words, and write down a word whenever he passes it.

PageRank-like propagation has been used in [10, 18] to rerank top retrieved documents, combined with other features in a heuristic way.

What about Equation 4 and 5 for the document graph, or more generally Equation 3? They are not like PageRank now because a different denominator $Deg(u)$ is used instead of $Deg(v)$. What is this essentially modeling? In fact, when we transform the graph in a certain way (by adding nodes and reassigning transition probabilities), one could see that Equation 4 is actually computing the "absorption probability" – the probability of each node to be absorbed to a termination state in such a transformed Markov chain. This is also guaranteed to converge.

Note that if we only apply Equation 4 once, the smoothing procedure is very similar to the method proposed in [20]. The difference is that they are using $f_u = c(w, d_u)$, which we have shown to be not as reasonable as $P(w|d_u)$.

How about language model smoothing with collection distribution (e.g., Jelinek-Mercer smoothing) and cluster structures [14, 11]? Intuitively, these models are smoothing document language models with global structures (e.g., collection, clusters), so that the estimated language models could satisfy global consistency (i.e., documents in the same global structure has similar representation). [20, 9] explored nearest neighbors, which guarantees local consistency of language models. A regularization framework like Equation 1, as shown in [25], satisfies both local and global consistency on a manifold structure.

4.2 Selection of Graphs

The smoothing framework we proposed is general and orthogonal to the selection of graphs. As long as the graph is constructed in a reasonable way (two related vertices are connected with a higher weighted edge, and are expected to have similar representations), the objective function in Equation 1 can be applied.

Thus either a fully connected graph (with well scaled edge weights) [25, 26], or a k-nearest-neighbor (kNN) graph [9, 10, 20] can be used for smoothing. We show our experimental results with kNN graphs. On the other hand, any reasonable distance/similarity measure could be applied to compute $w(u, v)$.

In scenarios that the number of documents/terms is too large, as in the case of a commercial search engine, one may think of using smaller subgraphs. Our proposed framework can be easily adapted to subgraphs. Of course, we need to ensure that the smoothed value of vertices in the subgraph are comparable with the values of the unsmoothed vertices (i.e., vertices not in the subgraph). The representations of both smoothed documents and unsmoothed documents with Equation 4 are all language models, which do not have a scaling problem. As for term graphs (e.g., Equation 6), it is easy to prove that

$$\sum_u P(w_u|d) = \sum_u P_{ML}(w_u|d).$$

This formula indicates that the probability of smoothed terms would not affect the probability mass of unsmoothed terms.

5. EXPERIMENTS

In Section 2.2 and Section 3, we proposed a general framework of smoothing language models and introduced various instantiations with document graph and word graph, resulting in several different smoothing strategies. In this section, we evaluate the effectiveness of these strategies empirically. Experiment results show that all the instantiations of the smoothing framework outperform the non-structural smoothing methods. Two instantiated approaches also outperform the state-of-the-art graph-based smoothing methods.

5.1 Experiment Setup

We use four representative TREC data sets: AP88-90, LA (LA Times), SJMN (San Jose Mercury News 1991), and TREC8. They are identical to the data sets used in [20], with the same source, query, and preprocessing procedure. The first three data sets are also identical to the data sets used in [14]. The basic statistics of the data sets are presented in Table 1. We used the title field of a query/topic description to simulate short keyword queries in our experiments.

	#documents	avg dl	queries	#total qrel
AP88-90	243k	273	51-150	21819
LA	132k	290	301-400	2350
SJMN	90k	266	51-150	4881
TREC8	528k	477	401-450	4728

Table 1: Basic information of data sets

For every data set, we construct a k-Nearest-Neighbor (kNN) graph of all documents, as well as a kNN graph of words. The edge weight of two documents is measured with the cosine similarity, formally

$$sim(d_1, d_2) = \frac{\sum_w c(w, d_1) \times c(w, d_2)}{\sqrt{\sum_w c(w, d_1)^2 \times \sum_w c(w, d_2)^2}}$$

.

The edge weight of two words is set to their mutual information [4]. We do not include the most frequent terms (which appear in $> 50\%$ documents) or the most infrequent terms (which appear in < 15 documents in TREC8, or < 7 documents in other data sets). This provides us with a word graph of $40k \sim 60k$ vertices. We control the density of the graph by adjusting the number of nearest neighbors. To ensure that the graph is undirected, we add an edge between u and v if either u is in v's k-nearest neighbors, or the converse.

After we smooth all the document language models and obtain the smoothed $P'(w|d)$ and possibly also a smoothed query language model $P(w|q)$, we use further smooth $P'(w|d)$ using Dirichlet prior [22] and obtain

$$P''(w|d) = \frac{|d|}{|d| + \mu} P'(w|d) + \frac{\mu}{|d| + \mu} P(w|C)$$

We then use the KL-divergence retrieval model to rank documents, where each document is scored based on the negative KL-divergence of the query language model and $P''(w|d)$ [12].

The additional Dirichlet smoothing is to model noise in the query and has been used in most existing work on smoothing language models with corpus structure [14, 9, 20]. When $P'(w|d)$ is unsmoothed (i.e., the maximum likelihood estimate), it boils down to the Dirichlet prior model, which is used as our baseline.

5.2 Basic Results

In Section 3, we introduced four different instantiations of smoothing, namely: smoothing document language model with document graph (**DMDG**); smoothing relevance score with document graph (**DSDG**); smoothing document language model with word graph (**DMWG**); and smoothing query language model with word graph (**QMWG**).

We compare these four methods in Table 2 along with the best results of the Dirichlet smoothing. In all our experiments, the cutoff of relevant documents is set as 1000. We see that all the four smoothing instantiations outperform the non-structural smoothing baseline. DMDG and DMWG outperform Dirichlet prior consistently and significantly.

Among the four proposed methods, we see that smoothing document language model tends to achieve better performance than smoothing query language model or the relevance score. One possible explanation is that smoothing document models is superior to smoothing query language models, since that the short keyword query only conveys very sparse information about the user's information need. Expanding a query in a wrong direction could hurt the retrieval performance.

Regularizing relevance scores has been discussed in existing literature [6], where a similar approach to DSDG is used. Clearly, we see that smoothing relevance scores alone is not as effective as smoothing the document language models. Indeed, by dealing with the richer representation of document language models, one has more flexibility in controlling the core retrieval modules to achieve better performance.

Using a document graph vs. a query graph to smooth document language models perform similarly, which is appealing since smoothing with word graph has not been well explored in the existing literature, suggesting potential room for further leveraging a word graph for smoothing.

5.3 Tuning Parameters

The selection of smoothing strategies would introduce a different set of new parameters. Generally, when a kNN graph is used, the size of the graph could be controlled by tuning the parameter k. In Equation 1, we introduced a parameter λ to balance the consistency of the language models on the graph, and the fidelity to the maximum likelihood estimates. Figure 2 presents the sensitivity of the retrieval performance to these introduced parameters.

From the left plot, we see that the performance is relatively stable over different k, even if a document only has a few (e.g., 10) neighbors. This is different from the observation made in [20], where smoothing with a larger number (≥ 100) of neighbors significantly outperforms a small number (≤ 50) of neighbors. This is because when propagations are processed iteratively, a few closest neighbors could well capture the local consistency.

Similar patterns are observed from other data sets and smoothing methods. We set the number of neighbors to 100 for a document graph and 50 for a word graph, unless specifically noted.

From the plot in the middle of Figure 2, we see that when λ is larger, the estimated language models would achieve more consistency on the surface at the expense of deviating more from the original estimates. When λ is set smaller, less smoothing effect is added to the language models. Setting λ in the range of $0.3 \sim 0.7$ usually yields good retrieval performance.

Finally, how many iterations do we have to run for the equations proposed in Section 3? In fact, we could even find a closed form solution for such a regularizer [25, 6]. However, its computation is time consuming. Intuitively, most smoothing effect occurs in the first a few iterations, which is indeed confirmed in the right plot of Figure 2, where we see that the performance becomes quite stable after a few iterations. Thus we may just run the algorithm for a few iterations in practice.

5.4 Comparison with existing methods

We now compare our methods with some state of the art smoothing methods.

	CBDM	DELM	DMDG	DMDG (1 iter.)
AP88-89	0.233	0.250	**0.254**	0.252
LA	0.259	**0.265**	0.260	0.258
SJMN	0.217	0.227	**0.235**	0.229
TREC	N/A	0.267	**0.271**	0.270

Table 3: Performance (MAP) comparison with existing smoothing methods

Liu and Croft [14] proposed a method (denoted as CBDM) to smooth document language models with cluster language models. [20] also proposed a smoothing strategy (denoted as DELM) to expand a document with its nearest neighbors. Both methods utilized the document similarity and are related to the DMDG instantiation we proposed. As we use the identical data sets and preprocessing procedures with their work, we compare the performance of our DMDG method with the best results reported in their papers.

We see that DMDG consistently outperforms CBDM. It also outperforms DELM on three data sets except for LA, but with a smaller improvement. This is not surprising, because the DELM method is very similar to our DMDG in-

Data		Dirichlet	DMDG	DMWG†	DSDG	QMWG
AP88-90	MAP	0.217	0.254 (+17.1%***)	0.252 (+16.1%***)	0.239 (+10.1%***)	0.239 (+10.1%)
	pr@10	0.432	0.447 (+3.5%*)	0.461 (+6.7%***)	0.453 (+4.9%**)	0.451 (+4.4%)
LA	MAP	0.247	0.258 (+4.5%**)	0.257 (+4.0%**)	0.251 (+1.6%**)	0.247 (0.0%*)
	pr@10	0.287	0.290 (+1.0%)	0.301 (+4.8%*)	0.285 (-0.7%)	0.287 (0.0%)
SJMN	MAP	0.204	0.231 (+13.2%***)	0.229 (+12.3%***)	0.225 (+10.3%***)	0.219 (+7.4%)
	pr@10	0.298	0.320 (+7.4%***)	0.326 (+9.3%***)	0.329 (+10.4%***)	0.326 (+9.4%)
TREC	MAP	0.257	0.271 (+5.4%***)	0.271 (+5.4%**)	0.261 (+1.6%)	0.260 (+1.2%)
	pr@10	0.450	0.468 (+4.0%*)	0.466 (+3.6%*)	0.464 (+3.1%)	0.474 (+5.3%**)

Table 2: Basic Results: Graph-based smoothing outperforms non-structural smoothing
†For efficiency reason, we conduct the DMWG smoothing by reranking the top 3000 results returned by Dirichlet method. All other methods are applied on the complete set of documents. We measure the statistical significance of the improvement using Wilcoxon test. *,**,*** means that the improvement hypothesis is accepted at significance level 0.1, 0.05, 0.01 respectively.

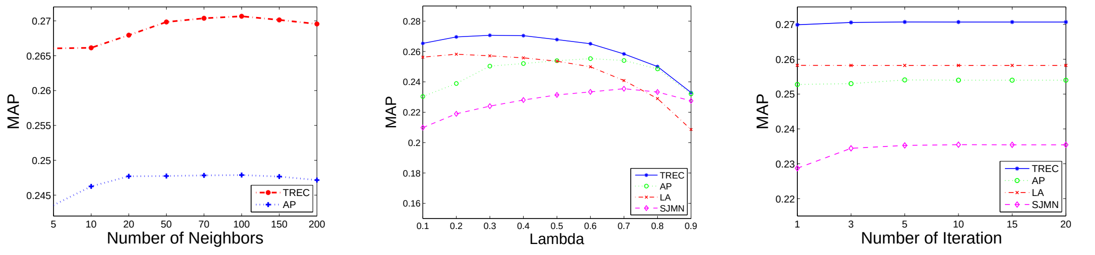

Figure 2: The sensitivity of parameters (k, λ, and iteration)
All results are obtained by DMDG. $k = 100$, $iteration = 10$ unless specified. Similar patterns are observed from other methods.

stantiation if only one iteration of Equation 4 is processed. The difference is that DELM also expands the document length besides smoothing the language models. The performance of DMDG with only 1 iteration is also presented in Table 3, from which we see that running more iterations improves retrieval performance. Similar improvements are achieved with the DMWG method. This comparison shows that the optimization framework of smoothing performs better than exploring either local or global consistency alone.

5.5 Combination with Pseudo-feedback

Pseudo feedback has been proved to be effective to update the query model using collection information [23]. It is interesting to see whether pseudo feedback could bring additional improvements to the graph-based smoothing methods, and whether it could benefit from a graph structure.

[20] has already proved that combining pseudo feedback and document graph achieves better results than both the feedback model and the smoothing model alone. In this work, we intend to explore whether combining the feedback model and the word graph would improve performance, because smoothing with a word graph is our novel exploration. Model-based feedback [23] combines the original query language model with a feedback model estimated from the top ranked documents. One natural thought is to smooth the feedback model with a word graph and Equation 7. An alternative way is to use the updated query model to retrieve the smoothed documents (i.e., with DMWG method) while the document language models are smoothed. We explored both strategies, and summarize the results in Table 4.

We use the model-based method in [23] to estimate a feedback model from the top 5 retrieved documents. In this process, we fix the noise parameter to be 0.9. We tune the combination parameter α (as in [23]) to get the optimal performance of pseudo-feedback. From Table 4, we see that both ways of combination improved performance. Combining pseudo feedback and DMWG significantly improves both DMWG and pseudo feedback. For the FB+DMWG model, we simply reuse the optimal parameters for Feedback and DMWG individually, without further tuning.

	FB	FB+QMWG	DMWG	FB†	FB+DMWG
AP	0.271	**0.273**	0.252	0.266	**0.271** **
LA	0.258	**0.267**	0.257	0.257	**0.267** **
SJMN	0.245	**0.246**	0.229	0.241	**0.249** **
TREC	0.278	**0.280**	0.271	0.278	**0.292** ***

Table 4: Performance (MAP) of combination of word graph and pseudo feedback
†to be consistent with DMWG, we use the same setup feedback (rerank the top 3000 docs and output 1000). This usually yields to a reduced performance than ranking all the documents.

6. RELATED WORK

Most of the related work has already been discussed in the previous sections of the paper. Here we give a brief summary of all the related work.

Smoothing language models [3] for information retrieval has been a fundamental and challenge problem in IR research. Traditional smoothing methods explore the collection information in an unstructured way [17, 3, 24]. Our work lies in the context of language model smoothing, but we utilize the graph structures from the collection to smooth document and query language models.

Recent research has explored document similarity graphs to smooth language models [21, 14, 9, 18, 20]. Cluster structures [21, 14], nearest neighbors [9, 20], and propagation-based methods [18, 11] have been proposed with various heuristic solutions. Our proposed optimization framework is a reasonable generalization of all such approaches, which provides a unified solution to this problem, as well as a road map for exploring novel smoothing approaches.

One of the concrete instantiations of our proposed framework, smoothing relevance scores with document graph, is related to score reranking work, such as [10, 6, 11]. [6] explores a regularization approach which is related to the objective function we introduced. However, they focus on regularizing the relevance scores, while we explore the general problem of smoothing language models. Experimental results show that smoothing relevance score alone is not as effective as smoothing document language models.

The proposed optimization framework is also related to the graph-based learning theme in machine learning. Similar objective functions have been explored in [27, 25, 26]. Their main focuses are machine learning problems such as semi-supervised learning and spectral clustering, while we explore graph structures to smooth language models for retrieval.

7. CONCLUSIONS

In this paper, we proposed a general optimization framework for smoothing language models with graph structures. The proposed framework not only gives a principled justification for many of the heuristics and a unified solution to smoothing language models with graphs, but also provides a road map for the exploration of novel smoothing methods. Following such a road map, we introduced four instantiations of smoothing methods, including two novel methods of smoothing document and query models with a word graph. Empirical results show that all proposed instantiations significantly improve over the unstructured smoothing methods. The two methods of smoothing document language models outperform the state of the art smoothing methods.

The proposed framework opens up many interesting questions for future research: Can we combine a document graph and a word graph to enhance the performance? Can we explore other types of graphs, like a query-dependent graph? How can we explore traditional IR heuristics in this unified framework? All these questions should be further studied.

Acknowledgments

We thank the anonymous reviewers for their useful comments. This work is in part supported by the National Science Foundation under award numbers IIS-0347933, IIS-0713571, and IIS-0713581. The first author is supported by a Yahoo! PhD Fellowship.

8. REFERENCES

[1] M. Belkin and P. Niyogi. Laplacian eigenmaps for dimensionality reduction and data representation. *Neural Comput.*, 15(6):1373–1396, 2003.

[2] S. Brin and L. Page. The anatomy of a large-scale hypertextual web search engine. *Comput. Netw. ISDN Syst.*, 30(1-7):107–117, 1998.

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

[4] K. W. Church and P. Hanks. Word association norms, mutual information, and lexicography. *Comput. Linguist.*, 16(1):22–29, 1990.

[5] W. B. Croft and J. Lafferty, editors. *Language Modeling and Information Retrieval*. Kluwer Academic Publishers, 2003.

[6] F. Diaz. Regularizing ad hoc retrieval scores. In *Proceedings of CIKM' 05*, pages 672–679, 2005.

[7] D. Hiemstra and W. Kraaij. Twenty-one at TREC-7: Ad-hoc and cross-language track. In *Proceedings of TREC 7*, pages 227–238, 1998.

[8] J. M. Kleinberg. Authoritative sources in a hyperlinked environment. *J. ACM*, 46(5):604–632.

[9] O. Kurland and L. Lee. Corpus structure, language models, and ad hoc information retrieval. In *Proceedings of SIGIR' 04*, pages 194–201.

[10] O. Kurland and L. Lee. Pagerank without hyperlinks: structural re-ranking using links induced by language models. In *Proceedings of SIGIR '05*, pages 306–313.

[11] O. Kurland and L. Lee. Respect my authority!: Hits without hyperlinks, utilizing cluster-based language models. In *Proceedings of SIGIR '06*, pages 83–90.

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

[13] V. Lavrenko and B. Croft. Relevance-based language models. In *Proceedings of SIGIR'01*, pages 120–127.

[14] X. Liu and W. B. Croft. Cluster-based retrieval using language models. In *Proceedings of SIGIR' 04*.

[15] R. Mihalcea and D. R. Radev, editors. *Textgraphs: Graph-based methods for NLP*, 2006.

[16] D. H. Miller, T. Leek, and R. Schwartz. A hidden Markov model information retrieval system. In *Proceedings of SIGIR 1999*, pages 214–221, 1999.

[17] J. M. Ponte and W. B. Croft. A language modeling approach to information retrieval. In *Proceedings of SIGIR 1998*, pages 275–281, 1998.

[18] T. Qin, T.-Y. Liu, X.-D. Zhang, Z. Chen, and W.-Y. Ma. A study of relevance propagation for web search. In *Proceedings of SIGIR 2005*, pages 408–415, 2005.

[19] A. Shakery and C. Zhai. Smoothing document language models with probabilistic term count propagation. *Information Retrieval*, 11(2):139–164, 2008.

[20] T. Tao, X. Wang, Q. Mei, and C. Zhai. Language model information retrieval with document expansion. In *Proceedings of HLT/NAACL 2006*, pages 407–414.

[21] J. Xu and W. B. Croft. Cluster-based language models for distributed retrieval. In *Proceedings of SIGIR' 99*, pages 254–261, 1999.

[22] C. Zhai and J. Lafferty. A study of smoothing methods for language models applied to information retrieval. *ACM Transactions on Information Systems*.

[23] C. Zhai and J. Lafferty. Model-based feedback in the language modeling approach to information retrieval. In *Proceedings of CIKM' 01*, pages 403–410, 2001.

[24] C. Zhai and J. Lafferty. A study of smoothing methods for language models applied to ad hoc information retrieval. In *Proceedings of ACM SIGIR'01*, pages 334–342, Sept 2001.

[25] D. Zhou, O. Bousquet, T. N. Lal, J. Weston, and B. Schölkopf. Learning with local and global consistency. In *NIPS*, 2004.

[26] D. Zhou and B. Schölkopf. Discrete regularization. *Semi-supervised learning*, pages 221–232, 2006.

[27] X. Zhu, Z. Ghahramani, and J. D. Lafferty. Semi-supervised learning using gaussian fields and harmonic functions. In *ICML*, pages 912–919, 2003.