Using Various Term Dependencies According to Their Utilities

Lixin Shi
DIRO, University of Montreal
CP. 6128, succursale Centre-ville
Montreal, H3C 3J7 Quebec, Canada
shilixin@iro.umontreal.ca

Jian-Yun Nie
DIRO, University of Montreal
CP. 6128, succursale Centre-ville
Montreal, H3C 3J7 Quebec, Canada
nie@iro.umontreal.ca

ABSTRACT
In this paper, we propose a model to integrate term dependencies. Different from previous studies, each pair of terms is assigned a different weight of dependency according to their utility to IR. The experiments show that our model can significantly outperform the previous dependency models using fixed weights.

Categories and Subject Descriptors
H.3.3 [**Information storage and retrieval**]: Information Search and Retrieval – *Retrieval Models*

General Terms
Algorithms, Performance, Experimentation, Theory.

Keywords
Discriminative Model, Language Model, Term Dependency, Dependency Strength.

1. INTRODUCTION
Traditional Information Retrieval (IR) approaches assume independence between terms, leading to the so-called bag-of-words models. However, terms in a natural language sentence are often dependent. For example, phrases such as "space program" and "black Monday" have drastically different meanings from those of their constituent words. The bag-of-words models are unable to consider their specific meanings. This problem has been the subject of a large number of studies. Within the language modeling framework, the typical approach is to create one or more additional models for documents to capture term dependencies of certain types. These models are combined with the traditional unigram model so that the documents in which the required dependencies between query terms are present are ranked higher. However, we observe that most previous approaches used fixed weights in such a combination. This is equivalent to say that any term dependency of the same type, say adjacency, has equal importance in the retrieval process. This is obviously untrue. For example, the adjacency between "black" and "Monday" in the expression "black Monday" corresponds to a strong dependency, while the one in "computer game" is less critical – even if the dependency between "computer" and "game" is ignored, the retrieval effectiveness using the unigram model would not be much degraded. The difference between the two cases lies in their utility for IR. On the one hand, if the meaning of the dependent terms together is compositional, then the omission to consider the dependency is not problematic. On the other hand, if the meaning is non-compositional (e.g. "black Monday"), then the consideration of the dependency in IR is crucial. The variable strength of dependency between terms has not been correctly coped with in previous approaches.

Another restriction in many previous studies is to consider adjacent words only. However, dependencies can span over more distant terms. For example, in "processor specifically designed for laptop computers", there is a strong dependency between the distant words "processor" and "laptop". Moreover, the strength of dependency between distant terms is not necessarily weaker than closer terms. For example, in "computer aided crime", the relation between "computer" and "crime" is much stronger and useful for IR than the adjacent pairs "computer aided" and "aided crime". Therefore, the strength of the dependency is not only a function of their distance.

In this paper, we propose a new dependency model to account for the above two aspects. In this model, each term dependency is weighted according to its strength and possible impact on the retrieval effectiveness. An important task is to determine such a strength and impact. We will propose a learning process for it using a set of features. Our experiments will show higher effectiveness of the approach than those in the literature.

2. RELATED WORDK
An obvious extension to a bag-of-words model is to combine it with higher-order n-gram models such as bigram and trigram models. However, the combination with bigram and trigram models leads to less improvement in retrieval effectiveness than one would expect [13]. The reason is twofold: (1) bigrams and trigrams only capture the dependencies between adjacent terms while many useful dependencies are between more distant terms. (2) a large number of the bigrams and trigrams do not correspond to true dependencies. To deal with the first problem, [14] proposes a less strict biterm model, which does not require the terms to occur in the specific order. To deal with dependencies between more distant terms, proximity models further consider the proximity of query terms in a document, such as [6][15][7][18]. The proximity language model (PLM) proposed in [18] performs empirically better than previous proximity models. Proximity information is integrated into the document model as follows:

$$\hat{\theta}^{B}_{D,i} = \frac{c(w_i; D) + \lambda Prox_B(w_i) + \mu P(w_i|C)}{|D| + \sum_{j=1}^{|V|} \lambda Prox_B(w_j) + \mu}$$

CIKM'10, October 26–30, 2010, Toronto, Ontario, Canada.

where $c(w_i; D)$ is the count of word w_i in D, $Prox_B(w_i)$ is proximity centrality of term w_i, and $P(w_i|C)$ is collection language model for smoothing. The best method to compute $Prox_B(w_i)$ turned out to be the following one:

$$P\_SumProx(q_i) = \sum_{q_j \in Q, q_j \neq q_i} f\left(Dis(q_i, q_j; D)\right)$$

where $f(x) = para^{-x}$, and $Dis(q_i, q_j; D)$ is the minimum distance between q_i and q_j in document D.

Metzler and Croft [8] proposed another method to capture the term dependency: a Markov random field (MRF) model for IR. A Markov random field is a graphical model in which a graph G consists of query nodes q_i (each representing a term) and a document node D. The ranking function is defined based on the following joint distribution over the random variables in G:

$$P_\Lambda(Q, D) = \frac{1}{Z_\Lambda} \prod_{c \in C(G)} \psi(c; \Lambda) \overset{rank}{=} \sum_{c \in C(G)} \log \psi(c; \Lambda)$$

where $Q = q_1, \ldots, q_n$ is a query, $C(G)$ the set of cliques in G, Z_Λ a normalization factor, and each $\psi(c; \Lambda)$ a non-negative potential function over the clique configuration (c) parameterized by Λ. Three types of potential function are defined in [8]: on clique of single query term, on ordered term clique and on unordered term clique. Each potential function is defined as a language modeling estimation smoothed by the collection, and the parameters $\lambda_T, \lambda_O, \lambda_U$ are weights associated to the models. Two specific dependency models are proposed – full dependency model (MRF-FD) which considers all dependencies and sequential dependency model (MRF-SD) which only considers dependence between adjacent query terms. In practice, MRF-FD is difficult to implement because of its complexity, especially when the query becomes long.

We notice that all the methods described above assign a fixed parameter to each component model. This means that all dependencies of a given type (e.g. adjacency or proximity) are assumed to have equal importance in the whole retrieval process. Although this makes the model easier to implement, the assumption is not reasonable. This fact has also been observed by Bendersky et al. [1]. In order to consider the variable impact of term dependencies, they extended the MRF-SD model so that the parameters become dependent on the individual term or term pair: $\lambda(q_i) = \sum_{j=1}^{k_{uni}} w_j^{uni} g_j^{uni}(q_i)$,$\lambda(q_i, q_{i+1}) = \sum_{j=1}^{k_{bi}} w_j^{bi} g_j^{bi}(q_i, q_{i+1})$ in which the functions $g_j(\cdot)$ is a feature defined over unigrams or bigrams, and w_j is its weight, a free parameter to be estimated. Documents are ranked according to the following equation:

$$P(D|Q) \overset{rank}{=} \sum_{q_i \in Q} \lambda(q_i) f_T(q, D) + \sum_{q_i q_{i+1} \in Q} \lambda(q_i, q_{i+1}) [f_O(q_i q_{i+1}, D) + f_U(q_i q_{i+1}, D)]$$

The above model is called Weighted MRF-SD (WSD). This extension goes in the same direction as the method we propose in this paper. However, the above method is still limited in the two following aspects:

- Term dependency is limited to two adjacent terms.
- The ordered term bigram and unordered term biterm are assigned the same importance, which is not reasonable.

In our model, we remove the above two limitations.

3. OUR APPROACH

In this paper, we propose a dependency model within the framework of discriminative models. This framework has the advantage of being able to include some distant dependencies without having to consider all the dependencies as in MRF. A typical discriminative model is formulated as follows [4][11]:

$$P(Rel|D, Q) = \frac{1}{Z} \exp\left(\sum_i^n \lambda_i f_i(Q, D)\right) \quad (1)$$

where $f_i(Q, D)$ is a feature function with weights λ_i and Z a normalization constant. In our model, in addition to unigrams, we consider the term dependencies between the following types term pair: (1) Ordered bigrams, (2) Unordered co-occurrence dependency within some distances. Several window sizes will be used: 2, 4, 8 and 16. Let us use C_w to denote term co-occurrences within the window size w. In particular, C_2 considers unordered adjacent terms, or biterms [14]. The ranking function is extended from Equation (1) to the following one:

$$P(Rel|D, Q) = \sum_{q_i \in Q} \lambda_U(q_i|Q) f_U(q_i, D) + \sum_{q_i q_{i+1} \in Q} \lambda_B(q_i, q_{i+1}|Q) f_B(q_i q_{i+1}, D) + \sum_{w \in W} \sum_{q_i, q_j \in Q, i \neq j} \lambda_{C_w}(q_i, q_j|Q) f_{C_w}(q_i, q_j, D)$$

This model contains three classes of features: unigram features $f_U(q_i, D)$, bigram features $f_B(q_i q_{i+1}, D)$ and co-occurrence features $f_{C_w}(q_i, q_j, D)$ where w is the window size. Each feature is associated with a function $\lambda(\cdot)$ denoting the importance of the feature for the query Q. This function allows us to take into account the dependencies between bigrams and co-occurring terms according to their strength and utility. This is fundamentally different from most previous models (except [1]) in which a fixed weight is assigned to the whole component model rather than to individual features.

The discriminative feature functions we use are as follows:

$$f_U(q_i, D) = P_U(q_i|Q)\, log\, P_U(q_i|D)$$

$$f_B(q_i q_{i+1}, D) = P_B(q_i q_{i+1}|Q)\, log\, P_B(q_i q_{i+1}|D)$$

$$f_{C_w}(q_i, q_j, D) = P_{C_w}(\{q_i . q_j\}|Q)\, log\, P_{C_w}(\{q_i . q_j\}_w|D)$$

where $\{q_i . q_j\}_w$ denote a pair of co-occurring terms q_i and q_j in document within a window of size w. For the ranking purpose, we will simply fix $\lambda_U(q_i|Q)$ at a constant 1, and try to vary the other λ functions for bigrams and co-occurring terms. We will use a set of features to determine the importance of a bigram and co-occurring terms in a query (see Section 4).

For the query models, we will simply use Maximum Likelihood (ML) estimation as follows, where t_R is an item of type R (a unigram, a bigram or a pair of co-occurring terms) and $c(t_R; Q)$ its count in the query:

$$P_R^{ml}(t_R|Q) = \frac{c(t_R; Q)}{|Q|_R}, \quad R \in \{U, B, C_2, C_4, C_8, C_{16}\}$$

For the document models on different types of item, we use Dirichlet smoothing as follows:

$$P_R(t_R|D) = \frac{c(t_R;D) + \mu_R \cdot P_R(t_R|C)}{|D|_R + \mu_R}$$

where $c(t_R;D)$ is the number of times the item t_R occurs in document D (within a window for C_w); $P_R(t_R|C) = \frac{\sum_{D \in C} c(t_R;D)}{\sum_{D \in C} |D|_R}$ is the collection language model; μ_R is a Dirichlet prior for the corresponding type of model; and $|D|_R$ is the document length in the expression of R, i.e. the total number of unigrams, bigrams or co-occurring terms within the corresponding window size. For instance, $|D|_U$ is the number of terms (unigrams) in the document, $|D|_{C_8}$ is the number of possible co-occurring terms in D.

Putting all together, we have the following final model:

$$\begin{aligned} P(Rel|D,Q) &\overset{rank}{=} \sum_{q_i \in Q} P_U^{ml}(q_i|Q) \log P_U(q_i|D) \\ &+ \sum_{q_i q_{i+1} \in Q} \lambda_B(q_i, q_{i+1}|Q) P_B^{ml}(q_i q_{i+1}|Q) \log P_B(q_i q_{i+1}|D) \\ &+ \sum_{w \in W} \sum_{\substack{q_i, q_j \in Q \\ i \neq j}} \lambda_{C_w}(q_i, q_j|Q) P_C^{ml}(\{q_i . q_j\}|Q) \log P_{C_w}(\{q_i . q_j\}_w|D) \end{aligned} \quad (2)$$

To give an example to illustrate the model, let us imagine a query of three words "*abc*". The first component of the model considers the unigrams *a, b* and *c*. The second component concerns the bigrams *ab* and *bc*. The third component considers the co-occurring term pairs {*a,b*}, {*b,c*} and {*a,c*}. These co-occurring term pairs will be evaluated in different window sizes in documents.

4. PARAMETERS ESTIMATEION

We have the following free parameters to be estimated: (1) Dirichlet priors μ for each component language model (2) Dependence strength λ for each bigram and pair of co-occurring terms.

Determining the Dirichlet Priors

It is intuitive to see that a longer document expression (e.g. with a larger window size) leads to a higher sparsity. This situation will require a large μ. Therefore, the Dirichlet priors are set according the document length in the bigram and co-occurrence expressions. We set μ_B, μ_{C_2}, μ_{C_4}, μ_{C_8} and $\mu_{C_{16}}$ proportionally to 1000, 1000, 3000, 7000 and 15000 respectively.

Learning the Importance of a Dependency

For a query Q_i consisting of n_i query terms, we have the following parameters to estimate: $\boldsymbol{\lambda}_{Bi} = \lambda_{Bi,1}, \dots, \lambda_{Bi,n_i-1}$, $\boldsymbol{\lambda}_{Ci} = \lambda_{Ci,1}, \dots \lambda_{Ci,\binom{n_i}{2}}$, where $C = C_2, C_4, C_8, C_{16}$. The set of parameters for Q_i is $\Lambda_i = \{\boldsymbol{\lambda}_{Bi}, \boldsymbol{\lambda}_{C_2 i}, \boldsymbol{\lambda}_{C_4 i}, \boldsymbol{\lambda}_{C_8 i}, \boldsymbol{\lambda}_{C_{16} i}\}$. We propose to determine these parameters using a set of training data using SVM. The training data is obtained as follows: for each query Q_i, we try to determine the best parameters Λ_i^* so as to maximize the output E (MAP in our case): $\Lambda_i^* = argmax_{\Lambda_i} E(R_{\Lambda_i}; T_i)$. To find Λ_i^*, we use the coordinate-level ascent algorithm proposed in [9]. Once Λ_i^* is found, the training data can be transformed into a set of pairs $\{(x_i, z_i)\}$, where x_i is a bigram or a pair of co-occurring terms, and z_i is the optimal $\lambda_R^*(x_i)$ ($R \in \{B, C_2, C_4, C_8, C_{16}\}$) found above.

Then we use epsilon Support Vector Machine Regression (ϵ-*SVR*) [17] to train $\lambda_R(\cdot)$ based on a set of features. The features we use in this paper for a pair of terms include the pointwise mutual information in a large independent collection and in the current text collection, tf and idf values, the distance between the terms, the binary value of term pair occurs in Termium[1] and the title of Wikipedia articles[2], etc. In our experiments, we use the LIBSVM[3] toolkit, and choose the radial basis kernel function. In our experiments, 10-fold cross validation is used.

5. EXPERIMENTS

5.1 Experimental Settings

Our experiments are performed on the collections listed in Table 1.

Table 1. Characteristics of document collections and queries

Coll.	#doc	Size (GB)	Query IDs	#Qry	Average Qry. Len
Disk1	510,637	1.26	1-200	199	3.72
Disk2	231,219	0.90	1-200	199	3.72
Disk4	293,710	1.28	251-350	100	2.85
Disk5	262,367	0.95	301-450	150	2.43
WT10g	1,692,096	11.03	451-550	97	2.44

We performed the following common pre-processing on all documents: Some unimportant fields and tags are removed; Stopwords are removed using a 625-stopword list from Lemur[4] toolkit; Words are stemmed by Porter stemmer.

5.2 Experimental Result Using Our Models

We compare our model with the following baseline models: unigram language model (Uni), MRF-SD, weighted MRF-SD (WSD), and PLM. For MRF-SD and PLM, we use a grid search to find the best MAP for each collection. The step for searching $\lambda_T, \lambda_O, \lambda_U$ for MRF-SD is 0.05. The search space of $para$ in the PLM model is from 1.5 to 1.9 and λ from 3 to 9. To compare with the WSD model, we simulate the implementation in [1] but without using features from *Google n-grams corpus* and *Microsoft 2006 RFP query logs*, which were used in [1].

In our model (DLM), the weights are assigned to individual term pairs. The weights are determined by cross validation: for each collection, 9/10 of the queries are used in turn as training data while the remaining 1/10 of the queries are used as test queries. In Table 2, we report the average effectiveness obtained in the cross validation.

Compared to the other models, we can see that this model performs generally better. The only exception is on Disk4 data, compared to PLM. In a number of cases, the differences with the other models are statistically significant. This result shows that the model we propose in this paper can indeed lead to additional gains in retrieval effectiveness. This result suggests that the two extensions we brought to this model, i.e. the consideration of more distant term dependencies and the weighting of individual term pairs, are indeed important factors that should be incorporated into IR models.

The improvements in some cases are not statistically significant. However, we have to notice that the weights we obtain from cross validation are far from optimal, while for the other models we tune the parameters to their best. So, the above comparison gave

[1] A phrase dictionary built for French/English machine translation, which contains 853K phrases.
[2] http://download.wikimedia.org/enwiki/, on 2007-12-12
[3] http://www.csie.ntu.edu.tw/~cjlin/libsvm/
[4] http://www.lemurproject.org/

Table 2. Comparing our models to baselines in MAP (‡ t.test<0.01, † t.test<0.05)

Coll.	Uni.	MRF-SD best $\lambda_T, \lambda_O, \lambda_U$		WSD		PLM best $para, \lambda$		DLM (trained by 10-fold cross validation)					DLM *(ideal)*
	MAP	MAP	cf. Uni.	MAP	cf. SD	MAP	cf. SD	MAP	cf. Uni	cf. SD	cf.WSD	cf.PLM	MAP
Disk1	.2382	.2453	+3.0%‡	.2478	+1.0%	.2425	-1.2%	**0.2486**	+4.3% ‡	+1.3%†	+0.3%	+2.5%‡	*0.2895*
Disk2	.2340	.2480	+6.0% ‡	.2504	+1.0%	.2447	-1.3%	**0.2520**	+7.7% ‡	+1.6%†	+0.6%	+3.0%‡	*0.3002*
Disk4	.1845	.1956	+6.0%	.1974	+0.9%	**.2011**	+2.8%	0.1999	+8.3%	+2.2%	+1.3%	-0.6%	*0.2342*
Disk5	.2365	.2461	+4.1% ‡	.2476	+0.6%	.2469	+0.3%	**0.2517**	+6.4% ‡	+2.3%†	+1.7%	+1.9%	*0.2894*
WT10g	.2042	.2169	+6.2% †	.2212	+1.9%	.2181	+0.5%	**0.2258**	+10.6%‡	+4.1%†	+2.1%	+3.5%	*0.2738*

considerable advantages to the other models. In order to see the potential of a model with the above two extension, we try to determine the best weights for each individual term pair by a coordinate-level ascent search as explained in Section 4. The ideal case is shown in the last column of Table 2. We can see that the optimal effectiveness is far beyond what we can obtain by cross validation. This shows that there is a large room for further improvements using our model.

6. CONCLUSIONS

Terms in documents and queries are often dependent. A model that ignores term dependencies is prone to retrieving much noise. However, a model that considers all the terms to be equally dependent also runs the danger to connect terms that are not strongly dependent and impose such a false dependency as a requirement in the retrieval process. As a result, such a model may miss relevant documents in which the false dependency does not appear. If one treats all the dependencies of the same kind in a unique way (i.e. by assigning a unique weight) as is done in most previous models, one will end up by assigning a moderate unique weight to the dependencies because of the above danger. The real problem is that term dependencies vary largely: a pair of terms such as "black Monday" is strongly dependent, and the use of this dependency in the retrieval process is highly beneficial; while other pairs of terms have weaker dependencies and can be treated separately. Therefore, each pair of terms should be treated in its own way according to the strength of the dependency and the usefulness of considering the pair of terms together. This is the goal of the approach proposed in this paper.

Our model extends the existing dependency models on the two following aspects:

- We assign weights to individual term pairs rather than to types of dependency;
- We consider dependencies between more distant terms, while different distances are also treated differently.

Our experimental results on TREC data showed that our model can consistently outperform the existing approaches. It is clear that the model has a great potential to be significantly better than the state-of-the-art methods, given the optimal effectiveness we could expect to obtain with it. The difference between the result obtained by our implementation and the ideal case suggests several promising avenues to pursue in the future:

- The set of features used to determine the weights of pairs of terms could be extended;
- Other methods for learning the weights could perform better;
- Finally, we may need a larger amount of training data to correctly learn the weights. Query logs (or clickthrough data) could be a valuable resource to this end.

These are some of the aspects that we will investigate in the future.

7. REFERENCES

[1] Bendersky, M., D. Metzler and W. B. Croft. Learning concept importance using a weighted dependence model. *WSDM-2010*, pp.31-40.

[2] Croft, W.B., H.R. Turtle, and D. D. Lewis. The use of phrases and structured queries in information retrieval. *ACM SIGIR-1991*, pp.32-45.

[3] Fagan, J. L. Automatic phrase indexing for document retrieval: An examination of syntactic and non-syntactic methods. *ACM SIGIR*, pp.91-101, 1987.

[4] Gao, J., H. Qi, X. Xia, J-Y. Nie. Linear discriminant model for information retrieval, *SIGIR-2005,* pp.290-297.

[5] Gao, J., J.-Y. Nie, G. Wu, and G. Cao. Dependence language model for information retrieval. *SIGIR-2004*, pp.170-177.

[6] Ken, E.M. Some aspects of proximity searching in text retrieval systems. *J. of Info. Sci.*, vol.18, pp.89-98, 1991.

[7] Lv, Y. and C. Zhai. Positional language models for information retrieval. *SIGIR-2009,* pp.299-306.

[8] Matzler, D. and W. B. Croft. A Markov random field model for term dependencies. *SIGIR-2005*, pp.472-479.

[9] Metzler, D. and W. B. Croft. Linear feature-based models for information retrieval. *Info. Retrieval*, 10(3) pp.257-274, 2007.

[10] Nallapati, R. and J. Allan. Capturing term dependencies using a language model based on sentence trees. *CIKM-2002*, pp.383-390.

[11] Nallapati, R. Discriminative models for information retrieval. *SIGIR-2004*, pp.64-71.

[12] Ribeiro-Neto, B.A. and R. Muntz. A belief network model for IR. *SIGIR-1996*, pp.253-260.

[13] Song, F. and W. Croft. A general language model for information retrieval. *CIKM-1999*, pp.316-321.

[14] Srikanth, M. and R. Srihari. Biterm language models for document retrieval. *SIGIR-2002*, pp. 425-426.

[15] Tao, T. and C. Zhai. An exploration of proximity measures in information retrieval. *SIGIR-2007,* pp.295-302.

[16] Turtle, H. R. and W. B. Croft. Inference networks for document retrieval. *SIGIR 1990*, pp.1-24.

[17] Vapnik, V.N. *Statistical Learning Theory*. Wiley, 1998.

[18] Zhao, J. and Y. Yun. A Proximity language model for information retrieval. *SIGIR-2009*, pp.291-298.