An Adaptive Local Dependency Language Model: Relaxing the Naïve Bayes' Assumption

Ramesh Nallapati and James Allan
Center for Intelligent Information Retrieval
Department of Computer Science
University of Massachusetts
Amherst, MA 01003
{nmramesh, allan}@cs.umass.edu

ABSTRACT

We describe a new probabilistic approach in the language modeling framework that captures adaptively the local term dependencies in documents. The new model works by boosting scores of documents that contain topic-specific local dependencies and exhibits the behavior of the unigram model in the absence of such dependencies. Contributions of the current work include adapting van Rijsbergen's [14] work in the classical probabilistic framework to the language modeling framework and adaptive modeling of within-sentence dependencies.

We evaluated our model on two different tasks of the Topic Detection and Tracking (TDT) research program, namely Story Link Detection and Topic Tracking. Our results show that the Local dependency language model consistently outperforms the basic unigram model on both the tasks.

Categories and Subject Descriptors

H.3.3 **[Information Search and Retrieval]**: Retrieval models - *language models, dependencies*

General Terms

Algorithms

Key Words

local term dependencies, language modeling, probabilistic approaches, graphical models, co-occurrences, sentences, maximum spanning tree, story link detection, topic detection and tracking

1. INTRODUCTION

Language Models have been found to be very effective in several information retrieval tasks. In the Language Modeling approach, we measure relevance of a document (or a query) to a topic by the probability of its generation from the topic model [12]. One major assumption made in the unigram language modeling is the naïve Bayes' assumption of independence of all terms with respect to one another. This allows us to compute the probability of generation of a document D from a topic model M as the product of probabilities of generation of each term in the document, as shown in the following equation:

$$P(D|M) = \prod_{i=1}^{|D|} P(w_i|M) \quad (1)$$

w is a term in the document.

But to quote the famous probability theorist De Finetti, "dependence is the norm rather than the contrary" [5]. From our own understanding of natural language, we know that the assumption of term independence is a matter of mathematical convenience rather than a reality.

However, the 'bag of terms' approach of the unigram language model, as shown in equation 1, ignores all the dependencies and any other positional information of terms in the document. Hence, it seems desirable to have a more sophisticated model that is capable of capturing the semantics of documents rather than just the term distributions. A first step towards achieving this objective is capturing term dependencies, since dependencies establish associations between terms and may shed more light on the underlying semantics of the document than unigram term distributions alone.

The present work is an attempt to relax the naïve Bayes' assumption through capturing local term dependencies using a new variation of the language modeling approach.

The remainder of the report is organized as follows. Section 2 summarizes attempts made in the past in capturing dependencies. We present the methodology of the new *Adaptive Local Dependency* (ALD) language modeling approach in section 3. In this section, we describe the motivation behind the approach and several modeling issues involved. We evaluate the complexity of ALD model and compare it with that of the unigram model in section 4. In section 5, we present a few interesting insights into why we think our new model works better than the unigram model. Section 6 describes some of the implementation details, while section 7 presents a brief description of Topic Detection and Tracking paradigm and two subtasks called Link detection and Topic tracking. In section 8, we describe the experiments performed and present the results obtained on two different tasks. Section 9 ends the discussion with a few observations and remarks on the performance of the ALD model.

2. PAST WORK

A vast majority of IR models assume independence of terms because the assumption leads to a tractable representation and most

Submission for Mathematical/Formal Methods in Information Retrieval workshop, SIGIR 2003. Please do not cite or distribute this draft paper.

IR systems practically have worked well under this assumption. There have been very few successful attempts at modeling dependencies especially in the domain of formal probabilistic models.

Keith van Rijsbergen presented an approach to capture document level term dependencies within the framework of the Binary Independence Retrieval model [16]. His probabilistic modeling approach using Expected Mutual Information Measure (EMIM) scores between terms [14]. A maximum spanning tree is constructed, with the terms as the nodes and the EMIM scores as the weighted edges. The tree captures the maximum dependencies between terms in the document. These dependencies are used in computing the similarity score between two documents. However, the approach unfortunately did not show promising results.

In other related work, Robertson and Bovey [15] tried including term pairs that have observable dependencies as separate terms with weights slightly different from the sum of weights or in some other way to allow for specific dependencies.

Turtle and Croft [18] investigated the use of an explicit network representation of dependencies by means of Bayesian inference theory. The use of such a network generalizes existing probabilistic models and allows integration of several sources of evidence within a single framework.

Fung and Crawford [6] worked on concept based information retrieval that captures dependencies between 'concepts' using a Bayesian inference network. One drawback of this approach is that the user has to identify the concepts manually in each document.

Attempts were also made to capture term dependencies using the vector space model. The generalized vector space model [19] is one such example which showed encouraging results.

In the area of language modeling, most attempts at capturing dependencies have been in the form of multigram language models [17]. Bigram and trigram models, though highly successful in the speech recognition task, have not met with great success in the domain of information retrieval systems.

In a very recent work, Nallapati and Allan [11] have shown that modeling sentences as maximum spanning trees in the language modeling framework, similar to van Rijsbergen's [14] modeling of documents, holds some promise in capturing local dependencies. Their *SenTree* model does not perform as well as unigram model, but they found that a linear mixture of SenTree and unigram models betters the unigram performance, but only in regions of low false alarm.

The present work, the *Adaptive Local Dependency (ALD)* language model is an improvement on the SenTree model. The ALD model combines the features of both unigram and SenTree models into a single framework. We will show in our results section that the new model consistently outperforms the unigram model on two different TDT tasks.

3. METHODOLOGY

Our goal is to build a topic model from a story D_1 or a set of stories $\{D_1, .., D_n\}$ and then decide whether a new story, say D_{n+1}, is predicted by that model. This section describes the methodology of the new model.

3.1 Exploiting sentence structure

A universal feature of all documents is the syntactic structure of sentences. Each sentence conveys a complete idea or a concept through a specific ordering of terms sampled from the language vocabulary. The sampling and ordering of terms depends on the intended concept and the underlying grammar of the language. The concepts or the semantics of the entire document are ultimately expressed as a grouping of such ordered samples of terms called sentences. In other words, the semantics of a document are expressed through the syntax of sentences. Hence, we believe modeling sentences, rather than terms as individual entities, better captures the underlying semantics of the document.

As we have seen earlier, unigram language models completely ignore the syntactic formation of sentences in documents. In the present approach, we attempt to capture it by modeling a document as a collection of sentences rather than as a 'bag of terms'. We model each sentence graphically, as a forest of terms as we shall see later.

3.2 Probability of Sentence Generation

In the ALD language model, we assume each sentence to be independent of the other sentences. This assumption is certainly not valid but it is less stringent than the assumption of term independence. The assumption allows us to compute the probability of generation of a story from a topic model as follows:

$$P(D|M) = \prod_{S \in D} P(S|M) \tag{2}$$

where M is a topic model and S is a sentence in a story D. The tricky part is computing the probability of generation of each sentence. Ideally, one would compute the probability of generation of a sentence as follows:

$$P(S|M) = P(w_1, w_2, ..., w_n|M) \tag{3}$$

where w_i is the i-th term and n is the number of terms in the sentence S. We also assume that all the terms in the sentence are unique, *i.e.*, we ignore any repeated occurrences of terms within a sentence. This assumption allows us to formulate our model in a clean way.

The joint probability in the right hand side of equation 3 is almost impossible to estimate due to the sparse nature of training data. To overcome this problem, we model the sentence as a forest in an approach motivated by van Rijsbergen's work. [14].

Given a sentence S and a model M, we first define the following undirected, weighted graph $G(V, E)$ as shown below.

$$V = \{w | w \in S\} \tag{4}$$

$$E = \{(w_i, w_j) \mid w_i, w_j \in V; \quad P(w_i, w_j|M) > 0\} \tag{5}$$

and

$$\begin{aligned} \forall (w_i, w_j) &\in E, \\ W(w_i, w_j) &= I(w_i, w_j|M) \\ &= log \frac{P(w_i, w_j|M)}{P(w_i|M)P(w_j|M)} \end{aligned} \tag{6}$$

where $P(w_i, w_j|M)$ is the probability that w_i and w_j co-occur in the same sentence given the model M and $W(w_i, w_j)$, the weight assigned to the edge (w_i, w_j) is equal to the model specific pointwise mutual information measure [9].

In other words, the set of nodes V in the graph G corresponds to the terms in the sentence S and an edge exists between terms w_i and w_j with a weight $I(w_i, w_j|M)$ if and only if they have a non-zero joint probability of within-sentence-co-occurrence with respect to the model. An example sentence-graph G is illustrated in figure 1. The graph has three connected components C_1, C_2 and C_3 as shown and corresponds to a sentence S that consists of the terms $\{w_1, \ldots, w_8\}$. The thickness of the edges is indicative of the value of edge weight or the probability of within-sentence co-occurrence with respect to the model.

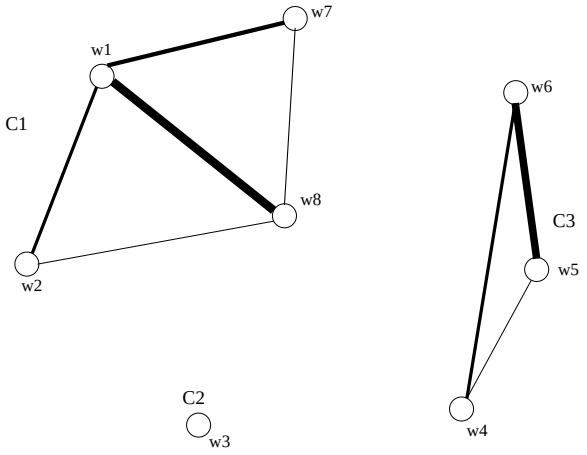

Figure 1: A sentence Graph

The first assumption we make is that the generation of terms in each connected component in the graph G is independent of terms in other connected components. If $\mathbf{w_C} = \{w|w \in C\}$ is the set of terms in a connected component C of G, then we can write

$$P(S|M) \approx \prod_{\mathbf{C} \in \mathbf{G}} P(\mathbf{w_C}|M) \tag{7}$$

where $P(\mathbf{w_C}|M)$ is the joint probability of all the terms in the connected component C. In the example of figure 1, the probability of sentence generation can be approximated as shown below.

$$\begin{aligned} & P(w_1,..,w_8|M) \approx P(\mathbf{w_{C_1}}|M)P(\mathbf{w_{C_2}}|M)P(\mathbf{w_{C_3}}|M) \\ = \; & P(w_1,w_2,w_7,w_8|M)P(w_3|M)P(w_4,w_5,w_6|M) \end{aligned} \tag{8}$$

Let's now consider how to approximate the joint probability of terms in a connected component C of G, given by $P(\mathbf{w_C}|M)$. Here, we follow the approach laid out by van Rijsbergen [14]. We assume that the joint probability is equal to the product of first order dependencies as an approximation to the chain rule shown below.

$$\begin{aligned} P(w_1,...,w_n|M) &\approx P_t(w_1,...,w_n|M) \\ &= \prod_{i=1}^{n} P(w_{m_i}|w_{m_{j(i)}},M) \quad 0 \le j(i) < i \end{aligned} \tag{9}$$

where $(m_1,...,m_n)$ is a permutation of the integers $1,..,n$ and $j(i)$ is a function mapping i into integers less than i and

$$P(w_{m_i}|w_{m_0},M) = P(w_{m_i}|M) \tag{10}$$

The permutation and the function $j(.)$ together define a dependence tree and the corresponding joint distribution $P_t()$ is called a probability distribution of first-order tree dependence.

A problem that remains is finding the dependence tree that best approximates the original joint distribution. We are aided here by a result by Chow and Liu [2] that showed that if the edge weights between the terms are measured by expected mutual information measure, then the best approximation P_t that has the least relative entropy with the true joint is given by the maximum spanning tree (MST) on the variables $w_1, w_2, ..., w_n$. Once the MST is found, we can compute the approximate joint $P_t(w_1,...w_n)$ by traversing breadth-wise or depth-wise on the MST starting from any leaf node and defining the direction of the edges to be same as the direction of traversal. It can be easily shown using Bayes' rule that the approximate joint $P_t(w_1,...w_n)$ remains the same irrespective of the starting node chosen.

For example, let's consider the graph in figure 1. For each component, we build a maximum spanning tree and define the approximate joint by the directed edges as shown in figure 2. The direction of conditioning is marked by the direction associated with an edge. As the figure indicates, we chose vertex w_2 as the starting leaf node of component C_1. Similarly, w_4 is the starting node of component C_3. Component C_2 is a singleton node, hence there is no tree associated with it. The approximate joint probability of terms in each component is given as follows.

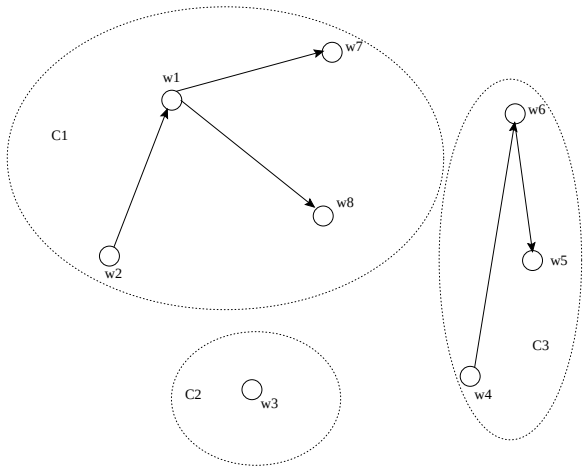

Figure 2: Bayesian networks defined on each connected component's maximum spanning tree

$$\begin{aligned} P(\mathbf{w_{C_1}}|M) &\approx P(w_7|w_1,M)P(w_8|w_1,M) \times \\ &\quad P(w_1|w_2,M)P(w_2|M) \\ P(\mathbf{w_{C_2}}|M) &= P(w_3|M) \\ P(\mathbf{w_{C_3}}|M) &\approx P(w_5|w_6,M)P(w_6|w_4,M)P(w_4|M) \end{aligned} \tag{11}$$

Thus, we see that each sentence is modeled as a model-dependent forest consisting of trees and singleton nodes.

3.3 Estimating the model parameters

As we have seen in section 3.2, to estimate the probability of sentence generation, all we need are the conditional probabilities of the form $P(w_i|w_j,M)$ for the dependencies in the MST and unigram probabilities of the form $P(w_i|M)$ for the leaf nodes in the MST and singleton nodes that have no dependencies associated with them. Typically these parameters of the model are estimated from a single document or a set of documents. Let D be the document from which we estimate the model parameters. We use the maximum likelihood estimates for the model parameters as shown below:

$$P(w_i|M) \approx \lambda \frac{C(w_i,D)}{\sum_{w \in D} C(w,D)} + (1-\lambda)\frac{C(w_i,GE)}{\sum_{w \in GE} C(w,GE)} \tag{12}$$

where $C(w,D)$ is the number of occurrences of term w in document D and GE is a large corpus of general English. We have used a smoothing parameter λ to prevent zero probabilities of unigrams.

The following maximum likelihood estimates are used to compute the joint and conditional probabilities:

$$P(w_i, w_j|M) \approx \frac{1}{|S|} \sum_{s \in S_{ij}} \frac{1}{|s|_{C_2}} \quad (13)$$

$$P(w_i|w_j, M) \approx \frac{1}{|S_j|} \sum_{s \in S_{ij}} \frac{1}{|s| - 1} \quad (14)$$

where S is the set of all sentences in D, S_{ij} is the set of sentences in D that contain the words w_i and w_j and S_j is the set of sentences in D that contain the word w_j. Note that we have not used any smoothing in estimating the joint and conditional probabilities.

3.4 Likelihood ratio

Finally, the story's relevance score with respect to a model is given in terms of likelihood ratio which is defined by the following equation:

$$Score(D, M) = \frac{P(D|M)}{P(D|GE)} = \frac{\prod_{S \in D} P(S|M)}{\prod_{w \in D} P(w|GE)} \quad (15)$$

Note that in computing the probability of document generation with respect to the general English model $P(D|GE)$, we use the unigram model. Computing the likelihood ratio with respect to the general English model provides a sound basis for comparison of relevance and also serves as a normalizing factor to sentence and document lengths.

4. COMPUTATIONAL COMPLEXITY

In this subsection we discuss the time and space complexity of implementing the ALD model with respect to a single story as the input. We will assume that the story contains N sentences and at most T terms per sentence.

The running time of each step in the algorithm is presented below:

1. Constructing the graph G of a sentence S with weighted edges: This requires computing pointwise EMIM for all pairs of terms in the sentence, each of which can be done in constant time using hash tables. Thus, this step has a complexity of $O(T^2)$.

2. Building a maximum spanning tree for each of the component. The worst case arises when G is a fully connected graph. We use a greedy algorithm to build the MST. The running time of this step is $O(T^2 Log(T))$ if we use disjoint-set forest implementation with union by rank and path compression heuristics [3].

3. Computing the probability of occurrence of the sentence from the topic model and the background models: This requires generating the conditional probabilities of each edge in the MST and the unigram probabilities of the start node and any other singleton nodes, each of which can be done in constant time. Hence, this step has a complexity of $O(T)$, which is the size of the MST.

Thus, the overall running time per sentence is $O(T^2 Log(T))$. Thus, for the entire document, the complexity is simply given by $O(N) \times O(T^2 Log(T)) = O(T^2 N Log(T))$. In comparison, the unigram model has a time complexity of only $O(TN)$.

Data set	Unigram	ALD
Train Set (6,361 pairs)	3	15
Test Set (27,538 pairs)	10	70

Figure 3: Comparison of average runtime of unigram and ALD models on training and test sets in the link detection task: All values are in seconds.

In practice, we notice that the run time of the ALD model is only about 5-7 times higher than that of the unigram model as figure 3 indicates. We believe that despite the slightly lower speed, it is still fast enough for most real time applications, considering the fact that the ALD model can process as many as 24,000 story pairs a minute.

Let us now look at the space-complexity of the model. Clearly, we need to index the joint and conditional probabilities for each pair of terms (w_i, w_j) in a sentence S in order to build the graph G (equation 5) and evaluate the joint probability of a sentence over the maximum spanning tree of each component in G. Thus each sentence needs a space of $O(T^2)$, hence each document needs a space of $O(T^2 N)$. The unigram model, on the other hand, has a space-complexity of only $O(NT)$.

5. DISCUSSION

In this section we will try to justify modeling within-sentence dependencies and explain the intuition behind why we expect the new model to perform better than the traditional unigram model. Apart from effectiveness, we will also discuss the features of the algorithm that makes it efficient.

5.1 Why model sentences?

We believe that the most significant dependencies between terms occur within a sentence in the form of noun-phrases (*e.g.,* gray-whale, balloon-adventure, Navy-sailor), proper names (*e.g.,* Timothy-McVeigh, Bill-Clinton), associations (*e.g.,* Glenn-NASA, asteroid-earth, Schindler-Oscar, Bush-Washington), etc. Dependencies do exist outside sentence boundaries too, but we posit that they do not tend to be as strong. This intuition explains several attempts in the past to incorporate noun phrases, proper name associations, etc. explicitly in retrieval models [4]. We believe our model includes all of those localized dependencies naturally into a unified probabilistic framework. From a computational viewpoint, modeling only within-sentence dependencies saves us considerable computational costs in comparison to modeling dependencies over the entire document (section 4), at the same time hopefully not compromising too much on effectiveness.

5.2 How might it help improve effectiveness?

Recall that our task is to compute the probability of generation of story D_2 with respect to a model $M(D_1)$ estimated from story D_1. We will focus on the generative probability $P(S|M(D_1))$ of a particular sentence S in D_2 for illustration.

Consider the graph G of S defined with respect to the model of D_1. Let w_i and w_j be two terms in S. As shown in equation 5, an edge (w_i, w_j) exists in G if and only if the probability of within-sentence-co-occurrence $P(w_i, w_j|M)$ is non-zero. Since we estimate $P(w_i, w_j|M)$ (see equation 13) entirely from D_1's statistics, it follows that an edge (w_i, w_j) exists in G only if the terms w_i and w_j co-occur in at least one sentence in D_1. Call all term pairs (w_i, w_j) of S that co-occur within a sentence boundary in D_1 'ac-

tive' term pairs. Thus all edges in G correspond to active term pairs of S. If the edge (w_i, w_j) happens to be a part of the maximum spanning tree of one of the components C in G, then, the approximate generative probability of the sentence $P(S|M(D_1))$ will contain a term of the form $P(w_i|w_j, M)$ (see equation 11 for example). Thus in effect, the ALD model searches for active term pairs in S and computes their conditional probabilities, while computing unigram scores for all other terms. In the absence of any active term pairs, the ALD model smoothly collapses to a unigram generative model since the graph G would then only consist of independent singleton nodes.

While it is certainly not guaranteed by theory, it turns out that the maximum likelihood estimate of the conditional probability $P(w_i|w_j, M)$ is almost always an order of magnitude higher than the corresponding unigram estimate $P(w_i|M)$. This can be attributed to the fact that co-occurrence of any pair of terms is typically a much rarer event than each of their occurrences. Thus, if the sentence S contains many active term pairs, the ALD model boosts the generative probability $P(S|M)$ via the conditional probabilities.

We expect that the sentence S contains considerable number of active term pairs if D_2, the document that contains S, is on the same topic as that of D_1 and none otherwise. In such a scenario, if D_2 is on-topic, the score assigned by ALD model is typically much higher than unigram score. If D_2 is off-topic with respect to D_1, the generative probability computed by the ALD model corresponds to unigram score. Thus we expect that on-topic pairs and off-topic pairs are well-separated by the ALD model resulting in better performance.

5.3 What makes it efficient?

A simple unigram model has proven not only very effective in most IR tasks, but is also one of the most efficient algorithms in terms of its time and space complexity, mainly owing to its simplicity. Any model that incorporates more information than simple term distributions alone seems to invariably loses out to unigram model on efficiency. We believe that we have managed to improve on the effectiveness of the unigram model considerably while limiting the loss in efficiency within reasonable bounds for all practical purposes.

As we have seen in section 4, the time complexity of the ALD model is only a factor of $O(TLogT)$ higher than that of the unigram model, while the space complexity is higher only by a factor of $O(T)$, where T is the maximum number of terms in a sentence.

One of the reasons for its efficiency being almost comparable with that of the unigram model is the fact that we take into account only within-sentence co-occurrences as dependencies, thereby reducing the number of first-order dependencies to evaluate from a maximum of $O((TN)^2)$ to just $O(T^2N)$, a reduction by a factor of $O(N)$.

Another important reason that makes the model suitable for most real-life systems is the fact that the model does not require one to estimate the conditional probabilities $P(w_i|w_j, M)$ over the entire space $|V| \times |V|$, where $|V|$ is the vocabulary size of the topic of M. Recall that a necessary (but not sufficient) condition that we compute conditional probabilility $P(w_i|w_j, M)$ for a pair of terms (w_i, w_j) is that they are active (see subsection 5.2) in document D from which we estimate the model M. Thus, for each document D from which we estimate a model $M(D)$, it is enough for us to index *a priori* the conditional probabilities of only those term pairs that co-occur in the same sentence in D. The number of these term pairs is much less than the entire space resulting in considerable savings.

There is another factor that keeps our model very efficient: as equation 15 indicates, we compute only unigram probabilities to estimate the generative probability of a document from the general English model GE. Note also that we do not use smoothing in estimating the conditional probabilities $P(w_i|w_j, M)$ (see equation 14). This effectively means that we never have to compute a conditional probability distribution at the corpus level, which would have been considerably expensive. Note that since conditional probabilities are evaluated only for active term-pairs, there is no zero-probability problem and hence smoothing is rendered unnecessary. Although smoothing is found to improve effectiveness in many applications, in our experiments, we have observed that smoothing the estimates of conditional probabilities with conditional estimates from a general English model did not result in any significant improvement in effectiveness. Hence we have abandoned smoothing the conditional probability estimates in return for huge savings in time and space.

5.4 Comparison with the Sentree model

The current model is an improvement on the SenTree model [11]. In the SenTree model, for any sentence S, we construct a maximum spanning tree (MST) over the fully-connected sentence graph with respect to the statistics of the topic model M. Thus, even when topic-related dependencies may exist only among a small fraction of term-pairs in a sentence, the MST imposes a dependency structure over the entire sentence. As a result, most of the dependencies evaluated by the SenTree model turn out to be spurious, resulting in degraded performance [11]. The ALD model seeks to overcome this drawback by adaptively modeling only topically relevant dependencies and exhibiting a unigram-like behavior by default. Thus it combines the best properties of both the SenTree and the unigram model into a single theoretical framework.

In terms of efficiency too, the ALD model outperforms the Sentree model. The Sentree model requires smoothing of the conditional probabilities with the general English estimates to overcome the zero-probability problem, which involves huge time and space overheads. The ALD model, on the other hand, computes conditional probabilities only for 'active' term pairs and hence does not require smoothing.

6. IMPLEMENTATION DETAILS

In this section, we discuss some of the system implementation details that are not covered in the discussion on methodology of the model.

During the process of training the model, we noticed that our expectation that active term pairs occur only in on-topic story pairs (see section 5) is not strictly valid. In fact, our data analysis revealed that many common term pairs such as 'people say', 'officials stated' co-occur within a sentence boundary in many off-topic story-pairs. Such term co-occurrences do not carry much information about the topical content of the stories, but the ALD model incorrectly assigns edges between such terms. These 'spurious' edges ultimately creep in as conditional probabilities and boost the scores of off-topic pairs. This results in a large number of false alarms resulting in a degradation in the performance of the model.

To suppress the effect of these spurious term pairs, we modified the definition of edges in equation 5 to include the following extra condition.

$$E = \{(w_i, w_j) \mid w_i, w_j \in V; \\ P(w_i, w_j|M) > 0;\ L(w_i)L(w_j) > C\} \qquad (16)$$

where

$$L(w) = \frac{P(w|M)}{P(w|GE)} \quad (17)$$

$L(w)$ is the likelihood ratio of the probability of term w given the model with respect to a general English model. It is similar to the familiar tf-idf weight and tells us the relative importance of the term with respect to the topic model. Thus, the modified definition of an edge in equation 16 requires the model to form an edge (w_i, w_j) only if the 'relative importance' of both the terms exceeds a threshold C. We empirically determined the best value of C to be 2500 and fixed it at this value on all our runs.

6.1 Tools

Clearly, the ALD model requires us to identify the boundaries of sentences in documents. We have used a simple heuristic-rule based sentence segmenter to detect sentence boundaries. We refer the reader to [11] for more information on the heuristic rules used.

Additionally, we have used a list of 423 most frequent words to remove stop words from stories. Stemming is done using the Porter stemmer [13] while the indexer and the ALD model are implemented using Java.

7. TOPIC DETECTION AND TRACKING

The new model we present in this work is expected to address some of the issues in Topic Detection and Tracking (TDT) [1]. TDT is a research program investigating methods for automatically organizing news stories by the events that they discuss. It includes several evaluation tasks, each of which explores one aspect of that organization. In this section, we describe two tasks called Link Detection and Topic Tracking.

7.1 Link Detection task

Link Detection requires determining whether or not two randomly selected stories discuss the same topic. In the language modeling approach to link detection, we build a topic model $M(D_1)$ from one of the stories D_1 in the pair (D_1, D_2). We then compute the generative probability of the second story D_2 from the model $M(D_1)$ as shown below.

$$Score(D_1, D_2) = P(D_2|M(D_1)) \quad (18)$$

Sometimes we may compute a *two-way score* to add symmetry to the formula, as shown below:

$$score(D_1, D_2) = \frac{1}{2}(P(D_2|M(D_1)) + P(D_1|M(D_2))) \quad (19)$$

If the score exceeds a pre-determined threshold, the system decides the two stories are linked. The system's performance is evaluated using a topic-weighted DET curve [10] that plots miss rate against false alarm over a large number of story pairs, at different values of decision-threshold. A Link Detection cost function C_{link} is then used to combine the miss and false alarm probabilities at each value of threshold into a single normalized evaluation score [20]. We use the minimum value of C_{link} as the primary measure of effectiveness and show DET curves to illustrate the error tradeoffs.

7.2 Topic Tracking

The TDT topic tracking task is defined to be the task of associating incoming stories with topics that are known to the system. Each target topic is defined by a set of training stories. The tracking task is then to classify correctly all subsequent stories as to whether or not they discuss the target topic.

Similar to the link detection task, we estimate a model M_T for topic T from the set of its training stories. Given any subsequent story D, we measure the probability of its generation with respect to the model $P(D|M_T)$. The system decides that the story is on-topic if the score exceeds a pre-defined threshold. We use topic weighted DET curves and a minimum cost value to evaluate the system's performance, as in link-detection task.

8. EXPERIMENTS AND RESULTS

In this section, we describe the experiments we performed on link-detection and topic-tracking and present the results we obtained.

8.1 Story Link Detection

The training set we used is a subset of the TDT2 corpus that includes six months of material drawn on a daily basis from six English-only news sources that are manually transcribed (when the source is audio). The collection comprises 6,361 story pairs and relevance judgments for each of them. The test set is the TDT3 corpus consisting of the 27,538 manually transcribed story pairs from multiple languages. If the source is non-English, we have used machine translations that are made available in the corpus. To derive the general English unigram model, we have used the same corpora as we performed experiments on.

We first trained the unigram model on the training set and used its best performance (at $\lambda = 0.2$) as the baseline for all the experiments. In our training experiments on the ALD model, we performed a search for the best performing values of the model parameters, namely the unigram smoothing parameter λ and the edge threshold C. The DET curve of the best performing values of $\lambda = 0.2$ and $C = 2500$ is shown in figure 4. It is clear from the plot that the ALD model outperforms the unigram model. This is reflected in the fact that we were able to bring down the minimum cost C_{link} by 7% from that of the unigram model as shown in the same figure.

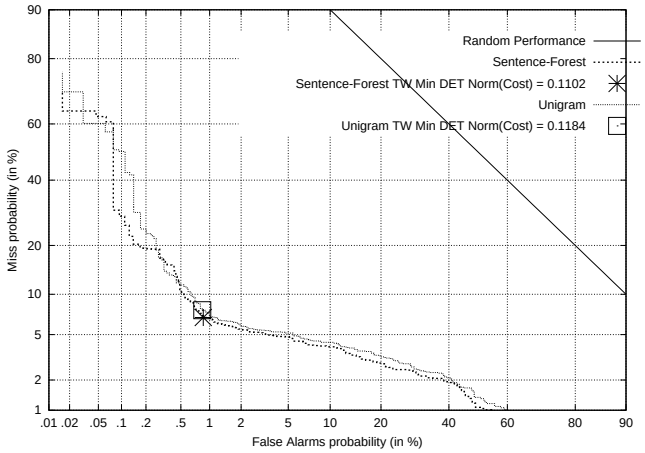

Figure 4: Performance of ALD model on the training set (Link detection task)

Having found the best performing values of various parameters, we now ran the system on the test set with the parameters set to the best training values. The performance of the system on the test set is shown in figure 5. Once again, we notice that the ALD model consistently outperforms the unigram model resulting in a reduction of about 4% in the minimum cost.

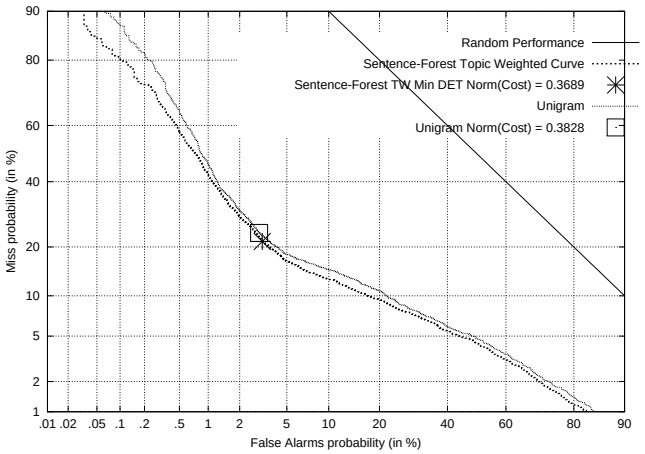

Figure 5: Performance of ALD model on the test set (Link detection task)

8.2 Topic Tracking

We used the multi-lingual TDT3 corpus in our tracking experiments. The general English unigram model is derived from about 40,000 stories in the TDT2 corpus. We used manual transcriptions and machine translations to English wherever necessary. There were 38 topics to track and we provided one English training story per topic ($N_t = 1$). The topic model of each topic is computed from the single training story and remains static (no adaptation is used) during the entire run. We used the best parameters obtained from training in the link detection task in our runs of unigram and ALD models.

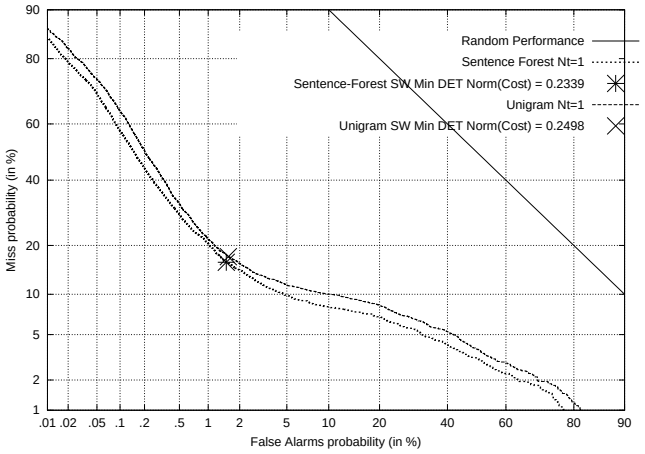

Figure 6: Performance of ALD model on Tracking task

Figure 6 compares the performance of the ALD model with that of the unigram model. We notice trends similar to the link detection task. The minimum cost of the ALD model is about 6.4% lower than that of the unigram model. In both figures 5 and 6, we note that the ALD DET curve dominates the unigram curve, so it's even better than reducing the minimum cost: false alarm rate is reduced at all miss rates.

8.3 Comparison to state-of-the-art TDT systems

The experiments we have performed on the link detection task do not correspond to any official conditions, but the tracking run does correspond to the standard evaluation conditions of TDT 2001. The best system in that evaluation was Limsi [8] which achieved a minimum cost of 0.0959 (compared to the unigram cost of 0.2498 and ALD minimum cost of 0.2339). We note that the state-of-the-art TDT systems like Limsi [8] and Relevance Models [7] typically rely on elaborate boosting techniques such as document expansion and unsupervised adaptation while employing unigram model as their core technology.

The goal of the present work is to demonstrate that incorporating local dependencies within the language modeling framework improves effectiveness. Hence we have used the unigram model as our baseline in all our runs. Although the performance of our system is well short of the state-of-the-art systems, we believe our model is a contribution at a fundamental level, and it can be easily extended to accommodate several boosting techniques used in the state-of-the-art systems.

9. CONCLUSIONS AND FUTURE WORK

In this work, we have presented a new approach of modeling a document by capturing local dependencies in documents adaptively through maximum spanning trees. Although this approach is built towards addressing a specific TDT task, we believe that the generality of the model permits one to apply it to any text classification or retrieval task.

Our experiments show that the ALD model outperforms the baseline unigram model on different data sets and on two different TDT tasks. To the best of our knowledge, this is the only probabilistic model that performs better than unigram model using statistics from the document alone. As such, we think the most important contribution of this study is the evidence we have provided that the performance of the existing IR systems can be improved by employing more sophisticated language models that capture positional and other syntactic information in documents.

As part of our future work, we would like to test the performance of our model on other IR tasks such as ad-hoc retrieval. We also envision building a query or document expansion device using the ALD model as the basic framework. Our hope is that the new expansion device may better other expansion models based on the unigram approach since our basic model outperforms the unigram model.

Acknowledgments

We would like to thank Andrew McCallum and Victor Lavrenko for their valuable comments. This work was supported in part by the Center for Intelligent Information Retrieval and in part by SPAWARSYSCEN-SD grant numbers N66001-99-1-8912 and N66001-02-1-8903. Any opinions, findings and conclusions or recommendations expressed in this material are the author(s) and do not necessarily reflect those of the sponsor.

10. REFERENCES

[1] Allan, J., Lavrenko, V. and Swan, R. Explorations Within Topic Tracking and Detection, *Topic Detection and Tracking: Event-based Information Organization*, James Allan, Editor, Kluwer Academic Publishers, 197-224, 2002.

[2] Chow, C.K. and Liu, C. N., Approximating discrete probability distributions with depepndecy trees, *IEEE Transactions on Information theory*, IT-14, 462-467 (1968).

[3] Cormen, T. H., Leiserson, C. E. and Rivest, R. L. *Introduction to Algorithms*, MIT Press, 1990.

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

[5] De Finetti, B. *Theory of Probability*, 1:146-161, Wiley, London 1974.

[6] Fung, R. M., Crawford, S. L., Appelbaum, L. A. and Tong, R. M. An architecture for probabilistic concept-based information retrieval, *ACM SIGIR*, 455-467, 1990.

[7] Lavrenko, V., Allan, J., DeGuzman, E., LaFlamme,D., Pollard, V. and Thomas, S. Relevance models for Topic Detection and Tracking, *Proceedings of the Conference on Human Language Technology (HLT)*, 2002.

[8] Lo, Y. and Gauvain, J., The Limsi Topic Tracking System for TDT 2001, *Proceedings of TDT 2001 workshop*.

[9] Manning, C. D. and Schutze, H. Foundations of Statistical Natural Language Processing, MIT Press, 1999.

[10] Martin, A., Doddington, G., Kamm, T. and Ordowski, M. The DET curve in assessment of detection task performance, *EuroSpeech*, 1895–1898, 1997.

[11] Nallapati, R. and Allan, J. Capturing Term Dependencies using a Language Model based on Sentence Trees, *ACM CIKM* 383-390, 2002.

[12] Ponte, J. M. and Croft, W. B. A Language Modeling Approach to Information Retrieval, *ACM SIGIR*, 275-281, 1998.

[13] Porter, M. F. An algorithm for suffix stripping, *Program*, 14(3):130-137, 1980.

[14] van Rijsbergen, K., *Information Retrieval*, Butterworths, 1979.

[15] Robertson, S. E. and Bovey, J. D. Statistical Problems in the Application of Probabilistic Models to Information Retrieval, *Technical Report, Center for Information Science, City University*, 1982.

[16] Robertson, S. E. and Sparck Jones, K. Relevance weighting of search terms, *Journal of the American Society for Information Sciences*, 27(3):129-146, 1976.

[17] Song, F. and Croft, W. B. A General Language Model for Information Retrieval, *Information and Knowledge Management*, 1999.

[18] Turtle, H. R. and Croft, W. B. *Inference Networks for Document Retrieval*, ACM SIGIR, 1-24, 1990.

[19] Wong, S. K. M., Ziarko, W. and Wong, P. C. N. Generalized Vector Space Model in Information Retrieval, *ACM SIGIR* 18-25, 1985.

[20] The Topic Detection and Tracking evaluation phase 2 plan, *http://www.nist.gov/speech/tests/tdt/tdt98/doc/tdt2.eval .plan.98.v3.7.pdf*.