Improve Latent Semantic Analysis based Language Model by Integrating Multiple Level Knowledge

Rong Zhang and Alexander I. Rudnicky

School of Computer Science, Carnegie Mellon University
5000 Forbes Avenue, Pittsburgh, PA 15213, USA
{rongz, air}@cs.cmu.edu

ABSTRACT

We describe an extension to the use of Latent Semantic Analysis (LSA) for language modeling. This technique makes it easier to exploit long distance relationships in natural language for which the traditional *n*-gram is unsuited. However, with the growth of length, the semantic representation of the history may be contaminated by irrelevant information, increasing the uncertainty in predicting the next word. To address this problem, we propose a multi-level framework dividing the history into three levels corresponding to document, paragraph and sentence. To combine the three levels of information with the *n*-gram, a Softmax network is used. We further present a statistical scheme that dynamically determines the unit scope in the generalization stage. The combination of all the techniques leads to a 14% perplexity reduction on a subset of Wall Street Journal, compared with the trigram model.

1. INTRODUCTION

Statistical language modeling plays an important role in various areas of natural language processing including speech recognition, machine translation and information retrieval. These applications need to know the probability of word string $\mathbf{w} = (w_1, w_2, ..., w_T)$. Using the Bayes rule, the probability $P(\mathbf{w})$ can be represented by the product of conditional probabilities of the next word given its history.

$$P(\mathbf{w}) = \prod_{t=1}^{T} P(w_t \mid \mathbf{h}_t) \quad (1)$$

where $\mathbf{h}_t$ denotes the history of word w_t that $\mathbf{h}_t = (w_1, w_2, ..., w_{t-1})$. The goal of language modeling is to provide an accurate estimation for $P(w_t \mid \mathbf{h}_t)$.

The *n*-gram model is the most commonly used language model. In such models the history is truncated into the most recent *n*-1 words with the assumption that the language is generated from a *n*-1 order Markov source. That is,

$$P(w_t \mid \mathbf{h}_t) \approx P(w_t \mid w_{t-n+1}, ..., w_{t-1}) \quad (2)$$

Such simplification benefits *n*-gram model by reducing the number of free parameters into an affordable range, and making it possible, in combination with smoothing and back-off techniques, to train on a limited size corpus. However, this assumption has obvious weakness that it doesn't take into account the contexts wider than *n*-1 words. Therefore, *n*-gram models lose a great deal of useful language information, such as long distance word correlations, syntactic constraints and semantic consistence.

Many attempts have been made in the last two decades [1] to solve these problems. Some successful examples include the class model, lattice model, caching model, decision tree model, maximum entropy model, whole sentence model, structured model, and the model based on Latent Semantic Analysis (LSA). The LSA based language model [2][3][4][5][6] aims to use the semantic relationship between words to increase the accuracy of prediction. Different from the trigger modeling which has the same motivation, the LSA approach maps the words and histories into a semantic space using Singular Value Decomposition (SVD) technique, and measures their similarities with geometric distance metrics, e.g. the inner product between vectors.

This paper presents our work on this new language modeling technique, including a multi-level framework to represent the history information, a statistical scheme to determine the unit scope, and a Softmax network to combine various parts. The combination of all the techniques leads to a 14% perplexity reduction on a subset of Wall Street Journal compared with a trigram model.

2. LANGUAGE MODELING BASED ON LATENT SEMANTIC ANALYSIS

2.1 Latent Semantic Analysis

LSA is a widely used statistical technique in the information retrieval community [7]. The primary assumption is that there exists some underlying or latent structure in the occurrence pattern of words across documents, and LSA can be used to estimate this latent structure. This is accomplished by two steps: construct a word-document co-occurrence matrix and reduce its dimensions using the SVD technique.

Let Γ, $|\Gamma| = M$, be the vocabulary of interest, and Ψ the training corpus comprising *N* documents. Define $\mathbf{X}$ as the co-occurrence matrix with *M* rows and *N* columns, in which each element $x_{i,j}$ corresponds to the word-document pair (w_i, d_j). Usually $x_{i,j}$ is expressed by the form of *tf-idf*, that *tf* is the normalized term frequency that conveys the importance of a word to a document, and *idf* is called the inverse document frequency that shows the discriminating ability of a word in the whole corpus.

Two weaknesses impair the application of such co-occurrence matrices. First, the dimensions *M* and *N* can be very large, e.g.

in our experiments M = 20000 and N = 75000. Second, the word and document vectors are often sparse and contain noise due to corpus limitations. SVD, a technique closely related to the eigenvector decomposition and factor analysis, is used to address these problems. SVD decomposes the word-document matrix into three components.

$$\mathbf{X} \approx \hat{\mathbf{X}} = \mathbf{U}\mathbf{S}\mathbf{V}^T \quad (3)$$

where $\mathbf{S}$ is the R×R diagonal matrix of singular values, $\mathbf{U}$ is the M×R matrix of eigenvectors derived from the word-word correlation matrix given by $\mathbf{X}\mathbf{X}^T$, and $\mathbf{V}$ is the N×R matrix of eigenvectors derived from the document-document correlation matrix given by $\mathbf{X}^T\mathbf{X}$. The value of R is typically chosen from the range 100 to 300, which is a trade-off between two considerations. That is, it should be large enough to cover all the underlying structure in the data, and at the same time, it should be small enough to filter out irrelevant details.

The result of LSA is the mapping from the discrete word and document sets, Γ and Ψ, to the R dimension continuous space that is associated with concepts, e.g. the word w_i is represented by the vector $\mathbf{u}_i$. The advantage of this mapping is that the semantic similarity in the words or texts, previously compared by manual categorization based on explicit knowledge, is reflected by the locations of their vectors and can be easily computed using linear algebra. Such advantage makes LSA a well-suited technique for many natural language processing tasks.

2.2 LSA based Language Modeling

Long distance correlation between words is a common phenomenon in natural language caused by either the closeness of meaning, e.g. the word "stock" and "bond" are both likely to occur in a financial news, or the syntactic collocation, e.g. "but also" is expected to occur somewhere after the "not only". Traditional n-gram model is unsuited for representing such phenomenon since the distances involved are often beyond n-1 words. However, such phenomenon is easy to model using the LSA framework where the correlation of words can be measured in the semantic space.

In this approach, the next word w_t is represented by the vector $\mathbf{u}_{w_t}$ which is derived from LSA, and the history $\mathbf{h}_t = (w_1, w_2, ..., w_{t-1})$ is treated as a pseudo-document which vector is as follows.

$$\mathbf{v}_{\mathbf{h}_t} = \mathbf{y}_{\mathbf{h}_t}^T \mathbf{U}\mathbf{S}^{-1} \quad (4)$$

where $\mathbf{y}_{\mathbf{h}_t}$ is the M dimension *tf-idf* vector for the pseudo-document. The closeness between w_t and its history $\mathbf{h}_t$ is measured by the cosine of the angle between $\mathbf{u}_{w_t}\mathbf{S}^{\frac{1}{2}}$ and $\mathbf{v}_{\mathbf{h}_t}\mathbf{S}^{\frac{1}{2}}$.

$$Sim(w_t, \mathbf{h}_t) = cos(\mathbf{u}_{w_t}\mathbf{S}^{\frac{1}{2}}, \mathbf{v}_{\mathbf{h}_t}\mathbf{S}^{\frac{1}{2}}) \quad (5)$$

Because the value range of $Sim(w_t, \mathbf{h}_t)$ is within [-1,1], we need to transform it to a probability metrics. One transformation is as follows.

$$P_{LSA}(w_t \mid \mathbf{h}_t) = \frac{\pi - cos^{-1}(Sim(w_t, \mathbf{h}_t))}{\sum_{w_i}[\pi - cos^{-1}(Sim(w_i, \mathbf{h}_t))]} \quad (6)$$

LSA based language model is further combined with n-gram model. (7) shows one form of the combinations.

$$P(w_t \mid \mathbf{h}_t) = \frac{P_{n-gram}(w_t \mid w_{t-n+1}, ..., w_{t-1}) \frac{P_{LSA}(w_t \mid \mathbf{h}_t)}{P(w_t)}}{\sum_{w_i}\left[P_{n-gram}(w_i \mid w_{t-n+1}, ..., w_{t-1}) \frac{P_{LSA}(w_i \mid \mathbf{h}_t)}{P(w_i)}\right]} \quad (7)$$

where $P(w_i)$ is the standard unigram probability.

The LSA based language model was first proposed by Bellegarda [2][3][4]. Experiments on Wall Street Journal data showed this model can achieve significant improvement on both perplexity and word error rate.

3. MULTI-LEVEL REPRESENTATION FOR HISTORY

The LSA representation for the history has several drawbacks. First, the context is treated as "a bag of words" that the word order is ignored. Compared with n-gram, the ignorance of word order results in the loss of information such as syntactic constraints. Fortunately, this point is partly addressed by the integration with traditional n-gram.

Another drawback is, with the growth of text length, the representation of the history may be contaminated by irrelevant information, increasing the uncertainty to predict the next word. For example, the contamination can be caused by the function words, e.g. "the" and "of". The function words don't carry much semantic information and are assigned a small global weight. However they still play a dominant role in the history due to their large number of occurrences. Sometimes the "noise" generated by function words may surpass the voice of the content words. As a result, the effectiveness of the LSA model is diluted.

Another reason responsible for the contamination is often neglected. Computed on the level of document, LSA attempts to represent all the semantic information in the document by a single low dimensional vector. Nevertheless, the variety of the topic, the complexity of text structure, plus the flexibility of personal expression and the ambiguity of natural language, make it impossible to realize this goal.

To address these problems, we proposed a multi-level framework to represent the history. In this framework, the history is divided into three levels: the document level, the paragraph level and the sentence level. Each level has its own vector in the semantic space.

Document: $\mathbf{v}_\mathbf{d} = \mathbf{y}_\mathbf{d}^T \mathbf{U}\mathbf{S}^{-1}$ (8)

Paragraph: $\mathbf{v}_\mathbf{p} = \mathbf{y}_\mathbf{p}^T \mathbf{U}\mathbf{S}^{-1}$ (9)

Sentence: $\mathbf{v}_\mathbf{s} = \mathbf{y}_\mathbf{s}^T \mathbf{U}\mathbf{S}^{-1}$ (10)

where $\mathbf{y_d}$, $\mathbf{y_p}$ and $\mathbf{y_s}$ denote the *tf-idf* vectors for the histories starting from the current document, paragraph and sentence.

The three-level framework can be understood as hierarchical representation associated with topic, sub-topic and local detail. We hope the contamination from the function words is reduced in the paragraph and sentence level histories. Meantime, the framework including three-level information is expected to be superior than the representation based on only one level. The next sections discuss two implementation issues: how to determine the unit scope in the generalization stage, and how to combine the framework with *n*-gram.

4. DYNAMIC SCOPE SELECTION

In our corpus, the boundaries for document, paragraph and sentence have been marked, e.g. each sentence is separated by "<s>" and "</s>". While we can use these marks to locate the starting point of a unit in the training stage, it's illegal to use them in the generalization stage. Thus we need to figure out a strategy to determine the unit scope for the generalization. A simple solution is to maintain a fixed size caching window. For example, we may assume that only the last 10 words belong to the current sentence while any context older than it is irrelevant. Another choice was proposed by [4] that adopted an exponential decay factor to discount history.

We address this issue based on the statistical result from the corpus. Let *x* be the variable denoting the length of unit *u*, and *u* can be the document, paragraph or sentence. First, the probabilities $P_u(x)$ are computed for each *u* from the training corpus. And then we compute the distribution functions $F_u(n)$ from $P_u(x)$.

$$F_u(n) = P_u(x \geq n) = \sum_{x=n}^{\infty} P_u(x) \tag{11}$$

$F_u(n)$ are represented by the solid curves in the Figure 1, 2 and 3. It's easy to prove that $F_u(n)$ are mono-decreasing functions with $F_u(0) = 1$ and $F_u(\infty) = 0$. Therefore, we can choose $F_u(n)$ as the decay function for history. Namely the discount factor for the word w_{t-n} is defined as

$$Decay_u(w_{t-n}) = F_u(n) \tag{12}$$

In our experiments, we further employ easy-computed functions to approximate $F_u(n)$.

$$F_{document}(n) \approx \alpha^n \tag{13}$$

$$F_{paragraph}(n) \approx exp^{-(\alpha_1 * n + \beta_1)^2} \tag{14}$$

$$F_{sentence}(n) \approx exp^{-(\alpha_2 * n + \beta_2)^2} \tag{15}$$

The dashed lines in the Figure 1, 2 and 3 show the approximation results; history length, in words, is shown on the abscissa. In our corpus, the maximum observed sizes are: document, 6000 words; paragraph, 505 words; sentence, 225 words. Note that (13), the decay function for document level history is similar to that in [4].

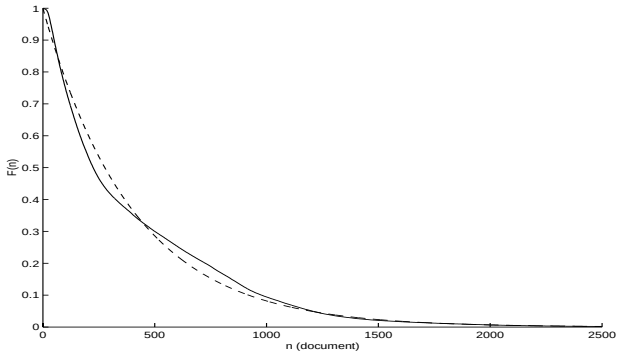

Figure 1 *F*(*n*) and approximation for document

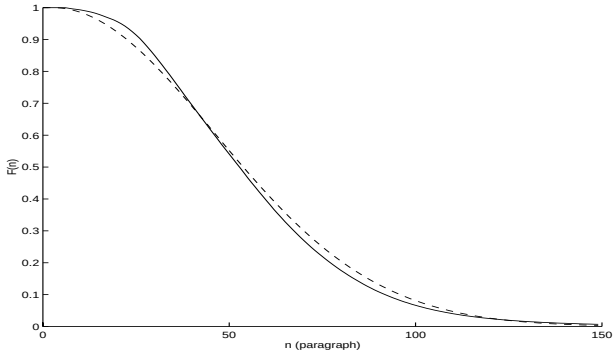

Figure 2 *F*(*n*) and approximation for paragraph

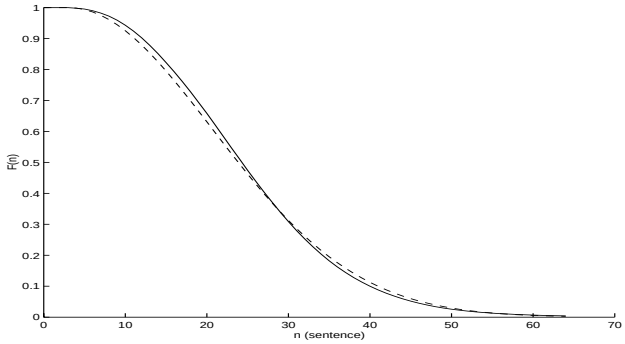

Figure 3 *F*(*n*) and approximation for sentence

5. SOFTMAX COMBINATION

The Softmax network is a powerful tool to learn probabilistic events, e.g. language modeling [6][8]. In a Softmax network, the activation function for the output node *i* is

$$O_i = \frac{exp^{g_i}}{\sum_j exp^{g_j}} \tag{16}$$

where g_i is the net value for node *i*. Apparently, the value of O_i is within (0, 1), and the sum of O_i is equal to 1. These properties make O_i a good mimic for probability.

We use Softmax network to combine LSA based model with *n*-gram model. Our Softmax network has 20000 output nodes, each of which represents a word in the vocabulary. There are four inputs for each word, corresponding to its *n*-gram

probability and its closeness to the three level histories separately.

$$I_i^1 = log(P_{trigram}(w_i \mid w_{t-2}, w_{t-1})) \quad (17)$$

$$I_i^2 = \pi - cos^{-1}(Sim(w_i, \mathbf{h_d})) \quad (18)$$

$$I_i^3 = \pi - cos^{-1}(Sim(w_i, \mathbf{h_p})) \quad (19)$$

$$I_i^4 = \pi - cos^{-1}(Sim(w_i, \mathbf{h_s})) \quad (20)$$

where $\mathbf{h_d}$, $\mathbf{h_p}$ and $\mathbf{h_s}$ denote the histories starting from the current document, paragraph and sentence. To speed up the training process, we applied two simplifications to the network architecture: abandon the hidden layer and tie the input weights. Consequently, the net value of node *i* has the form of

$$g_i = \lambda_1 I_i^1 + \lambda_2 I_i^2 + \lambda_3 I_i^3 + \lambda_4 I_i^4 \quad (21)$$

Moreover, the output of node *i*, which is the estimation for $P(w_t = i \mid \mathbf{h}_t)$, is

$$P(w_t = i \mid \mathbf{h}_t) = O_i = \frac{exp^{\sum_{k=1}^{4} \lambda_k I_i^k}}{\sum_j exp^{\sum_{k=1}^{4} \lambda_k I_j^k}} \quad (22)$$

Optimization of the weights is achieved by maximizing the perplexity of the corpus.

$$L = \sum_t log(P(w_t \mid \mathbf{h}_t)) \quad (23)$$

6. EXPERIMENTS

6.1 Data Set

The data set used in our experiments were extracted and processed from the Wall Street Journal 1987 to 1989. In our corpus, the boundaries for document, paragraph and sentence were carefully marked. Our training set consists of 75000 documents comprising about 34 million word tokens. We chose the most frequent 20000 words to constitute the vocabulary. To build the LSA based model, a 20000*75000 word-document matrix was constructed on the training set, and the SVD algorithm was applied to it. In our experiments, the number of singular value was set to 100. Our test set has 5000 documents and about 422K word tokens. We also maintained a cross-validation set that contains another 5000 documents and about 400K word tokens. The Softmax network is optimized on the cross-validation set.

6.2 Experimental Results

In our experiment, we compared the perplexities computed from eight combinations between trigram and LSA based models. We used the dynamic scope selection scheme described in Section 4 to discount the old history. The results are showed in Table 1.

The best performance was achieved by the combination of trigram and the LSA model that includes all the three-level histories. Compared with the traditional trigram, a 14.3% perplexity reduction was obtained. Compared with the combination between trigram and document level model, which is adopted by original LSA based model, there is also a 5.2% reduction.

We also noticed that the models containing the paragraph level history outperform those without it. This may suggest that sub-topic or "focus" information constitutes the most effective scope for LSA-derived semantic information.

Combinations	Perplexity
Trigram	88.82
Trigram + D.	80.24
Trigram + P.	77.36
Trigram + S.	79.48
Trigram + D. + P.	77.11
Trigram + D. + S.	77.80
Trigram + P. + S.	77.69
Trigram + D. + P. + S.	76.09

Table 1 Performance of *n*-gram and LSA based models
D.: Document level model; P.: Paragraph level model; S.: Sentence level model.

ACKNOWLEDGEMENT

This research was sponsored in part by the Space and Naval Warfare Systems Center, San Diego, under Grant No. N66001-99-1-8905. The content of the information in this publication does not necessarily reflect the position or the policy of the US Government, and no official endorsement should be inferred.

REFERENCES

[1] Ronald Rosenfeld, "Two decades of statistical language modeling: Where do we go from here", Proc. of IEEE, vol. 88, no. 8, pp. 1270-1278, Aug. 2000.

[2] Jerome R. Bellegarda, "A multi-span language modeling framework for large vocabulary speech recognition", IEEE Trans. on Speech and Audio Processing, vol. 6, no. 5, pp. 456-467, Sept. 1998.

[3] Jerome R. Bellegarda, "Large vocabulary speech recognition with multi-span statistical language models", IEEE Trans. on Speech and Audio Processing, vol. 8, no. 1, pp. 76-84, Jan. 2000.

[4] Jerome R. Bellegarda, "Exploiting latent semantic information in statistical language modeling", Proc. of IEEE, vol. 88, no. 8, pp. 1279-1296, Aug. 2000.

[5] Noah Coccaro and Daniel Jurafsky, "Towards better integration of semantic predictors in statistical language modeling", Proc. of ICSLP-1998, Volume 6, 2403-2406.

[6] Yoshua Bengio, Rejean Ducharme, and Pascal Vincent, "A neural probabilistic language model," Advances in Neural Information Processing Systems, 2001.

[7] Ricardo Baeza-Yates and Berthier Ribeiro-Neto, "Modern information retrieval", Addison-Wesley, 2000.

[8] Wei Xu and Alexander I. Rudnicky, "Can artificial neural networks learn language models", Proc. of ICSLP-2000.