

Exploiting Syntactic, Semantic and Lexical Regularities in Language Modeling via Directed Markov Random Fields

Shaojun Wang* Shaomin Wang† Russell Greiner* Dale Schuurmans* Li Cheng*

*University of Alberta

†Massachusetts Institute of Technology

Abstract

We present a directed Markov random field (MRF) model that combines n -gram models, probabilistic context free grammars (PCFGs) and probabilistic latent semantic analysis (PLSA) for the purpose of statistical language modeling. Eventhough the composite directed MRF model potentially has an exponential number of loops and becomes a context sensitive grammar, we are nevertheless able to estimate its parameters in cubic time using an efficient modified EM method, *the generalized inside-outside algorithm*, which extends the inside-outside algorithm to incorporate the effects of the n -gram and PLSA language models. We generalize various smoothing techniques to alleviate the sparseness of n -gram counts in cases where there are hidden variables. We also derive an analogous algorithms to find the most likely parse of a sentence and to calculate the probability of initial subsequence of a sentence, all generated by the composite language model. Our experimental results on the Wall Street Journal corpus show that we obtain significant reductions in perplexity compared to the state-of-the-art baseline trigram model with Good-Turing and Kneser-Ney smoothing techniques.

Index Terms — Language modeling, lexical information, syntactic structure, semantic content, Markov chain, probabilistic context free grammar, probabilistic latent semantic analysis, directed Markov random field, expectation maximization, latent variable models, generalized inside-outside algorithm

1 Introduction

The goal of statistical language modeling is to accurately model the probability of naturally occurring word sequences in human natural language. The dominant motivation for language modeling has traditionally come from the field of speech recognition [24], however statistical language models have recently become more widely used in many other application areas, such as information retrieval [17], machine translation [11], optical character recognition, spelling correction, document classification, and bioinformatics [16, 19, 48].

There are various kinds of language models that can be used to capture different aspects of natural language regularity. The simplest and most successful language models are the Markov chain (n -gram) source models, first explored by Shannon in his seminal paper [49]. These simple models are effective at capturing local lexical regularities in text. Subsequently, a wide variety of smoothing methods have been developed to address the problem of estimating rare events for these models [13]. The resulting smoothed n -gram language models have become a key component of state of the art speech recognizers, by helping to resolve acoustic ambiguities by placing higher probability on more likely word strings.

While Markov chains are efficient at encoding local word interactions, natural language clearly has a richer structure than can be conveniently captured by an n -gram model. For example, attempting to increase the order of an n -gram to capture longer range dependencies in natural language immediately runs into the curse of dimensionality [6].

Many recent approaches have been proposed to capture and exploit different aspects of natural language regularity with the goal of outperforming the simple n -gram model. For example, the structural language model [5, 12, 26, 43, 51] effectively exploits syntactic regularities to achieve greater accuracy than the n -gram model, and the semantic language model [3, 22, 47] exploits document-level semantic regularities to acheive similar improvements. Unfortunately each of these language models only targets some specific, distinct linguistic phenomena [46]. The key question [4, 34, 44] we are investigating is how to model natural language in a way that simultaneously accounts for the lexical information inherent in a Markov chain

model, the hierarchical syntactic structure captured in a stochastic branching process, and the semantic content embodied by a bag-of-words mixture of log-linear models—all in a unified probabilistic framework.

Several techniques for combining language models have been investigated. The most commonly used method is simple linear interpolation [12, 45], where each individual model is trained separately and then combined by a weighted linear combination. The weights in this case are trained using held out data. Even though this technique is simple and easy to implement, it does not generally yield effective combinations because the linear additive form is too blunt to capture subtleties in each of the component models. Another approach is based on Jaynes’ maximum entropy (ME) principle [29, 45]. This approach has several advantages over other methods for statistical modeling, such as introducing less data fragmentation, requiring fewer independence assumptions, and exploiting a principled technique for automatic feature weighting, and has since become a dominate technique in statistical natural language processing. It is now well known that for complete data, the ME principle is equivalent to maximum likelihood estimation (MLE) in an undirected Markov random field. In fact, these two problems are exact duals of one another [7]. The major weakness with maximum entropy methods, however, is that they can only model distributions over explicitly observed features, whereas in natural language we encounter hidden semantic [3] and syntactic information [12]. Recently we [53, 54] proposed the latent maximum entropy (LME) principle, which extends standard maximum entropy estimation by incorporating hidden dependency structure. In previous work [54], we have used the LME principle for statistical language modeling. However, the authors have been unable to incorporate PCFGs in this framework, because the tree-structured random field component creates intractability in calculating the feature expectations and global normalization over an infinitely large configuration space. Previously we had envisioned that MCMC sampling methods [1, 33] would have to be employed, leading to enormous computational expense.

In this paper, instead of using an undirected MRF model, we present a unified generative *directed Markov random field model* framework that combines n -gram models, PCFG and PLSA. Unlike undirected MRF models where there is a global normalization factor over an infinitely large configuration space, which often causes computational difficulty, the directed MRF model representation for the composite n -gram/syntactic/semantic model only requires local normalization constraints. More importantly it satisfies certain factorization property which greatly reduces the computational burden and makes the optimization tractable. We review the most popular language models in section 2 and propose the composite n -gram/syntactic/semantic language model in section 3. In sections 4, by exploiting the factorization properties of the composite model, we propose a simple yet efficient and exact EM iterative optimization method, *the generalized inside-outside algorithm*, which enhances the well known inside-outside algorithm [2, 31] to incorporate the impact of the n -gram model and PLSA. To cope with sparse data in n -gram component, we extend standard smoothing techniques to handle hidden variables. In section 5, we present a *generalized left-to-right inside* procedure to compute the probability of an initial subsequence in the composite language model. Given that n -gram, PCFG and PLSA models have been well studied for several decades, it is striking that this procedure has gone undiscovered until now. Finally, we give experimental results in section 6 and point out future research in section 7.

2 A Composite Trigram/Syntactic/Semantic Language Model

Natural language encodes messages via complex, hierarchically organized sequences. The local lexical structure of the sequence conveys surface information, while the syntactic structure, encoding long range dependencies, carries deeper semantic information; see Figure 1. Various models have been proposed to model each specific linguistic phenomenon. Below, we briefly review the most popular ones.

The Markov chain source model for natural language was first explored by Shannon [49] to capture local lexical regularities. The commonly used trigram model, or second order Markov chain model, is constructed by assuming all histories with the same previous two words belong to the same equivalence class. The maximum likelihood estimate of a trigram probability given a training corpus can be calculated by the relative frequency count. Many smoothing techniques have been proposed to address the problem of rare events for these models.

There are two approaches to modeling syntactic structure in natural language. The simplest approach uses a probabilistic context-free grammar (PCFG) to express the distribution over word sequences [27, 31, 33].

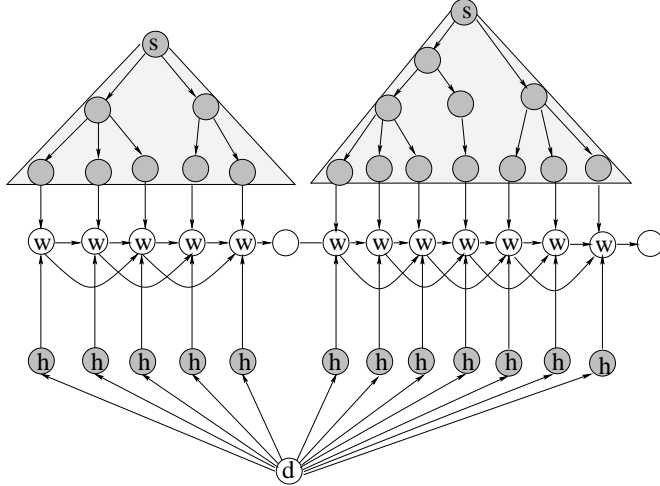


Figure 1: The observables in natural language consist of words, sentences, and documents; whereas the hidden data consists of sentence-level syntactic structure and document-level semantic content. The figure illustrates a composite chain/tree/table model incorporating these aspects, where light nodes denote observed information and dark nodes/triangles denote hidden information.

However, a more complicated approach [12, 43] uses a parser to uncover phrasal heads, words and their corresponding nonterminal tags; all of which stand in an important relation to the current word in the context of prediction.

A document can be viewed as a collection of semantically homogeneous sentences. Given a large number of documents, latent semantic analysis (LSA) attempts to discover compact semantic representations of text data that go beyond simple lexical-level word co-occurrences. This is achieved by mapping a high-dimensional vector representation of documents (term-frequency vectors) to a lower dimensional representation in a so-called latent semantic space. Semantic relations between words and documents can then be easily defined in terms of their proximity in the semantic space by dimensionality reduction techniques [3, 22].

Let X denote a set of random variables $(X_\tau)_{\tau \in \Gamma}$ taking values in a (discrete) probability spaces $(\mathcal{X}_\tau)_{\tau \in \Gamma}$ where Γ is a finite set of states. We define a (discrete) directed Markov random field to be a probability distribution \mathcal{P} which admits a recursive factorization if there exist non-negative functions, $k^\tau(\cdot, \cdot)$, $\tau \in \Gamma$ defined on $\mathcal{X}_\tau \times \mathcal{X}_{pa(\tau)}$, such that $\sum_{x_\tau} k^\tau(x_\tau, x_{pa(\tau)}) = 1$ and \mathcal{P} has density

$$p(x) = \prod_{\tau \in \Gamma} k^\tau(x_\tau, x_{pa(\tau)}) \quad (1)$$

If the recursive factorization respects to a graph \mathcal{G} , then we have a Bayesian network [32]. But broadly speaking, the recursive factorization can respect to a more complicated representation other than a graph which has a fixed set of nodes and edges.

Assume that we use a trigram Markov chain to model local lexical information, a probabilistic context free grammar to model the syntactic structure and a probabilistic latent semantic analysis (PLSA) [22] to model its semantic content of natural language. Each of these models can be represented as a directed Markov random field model. If we combine these three models, we obtain a composite model that is represented by a rather complex chain-tree-table directed MRF model.

A context free grammar (CFG) G is a 4-tuple $(\Sigma, \mathcal{V}, \mathcal{R}, S)$ [23] that consists of: a set of non-terminal symbols Σ whose elements are grammatical phrase markers; a vocabulary of $\mathcal{V} = \{v_1, \dots, v_M\}$ whose elements, words v_i , are terminal symbols of the language; a sentence “start” symbol $S \in \Sigma$; and a set of grammatical production rules \mathcal{R} of the form: $A \rightarrow \gamma$, where $A \in \Sigma$ and $\gamma \in (\Sigma \cup \mathcal{V})^*$. A probabilistic context free grammar (PCFG) is a CFG with a probability assigned to each rule, such that the probabilities of all rules expanding a given nonterminal sum to one; specifically, each right-hand side has a conditional probability given the left-hand side of the rule. A PCFG is a branching process [15, 36] and its distribution can be

represented in a form of (1), thus it can be treated as a directed MRF model eventhough the straightforward representation as a complex directed graphical model is problematic [35], since given observed leaf nodes, there are potentially exponentially many distinct parse tree structures.

A PLSA [22] is a generative probabilistic model of word-document co-occurrences using the bag-of-words assumption is described as follows: (1) choose a document d_k with probability $\theta(d_k)$, (2) select a semantic class h with probability $\theta(d_k \rightarrow h)$, (3) pick a word w with probability $\theta(h \rightarrow w)$. Since only pair of (d_k, w) is being observed, as a result, the joint probability model is a mixture of log-linear model with the expression $p(d_k, w) = \theta(d_k) \sum_h \theta(h \rightarrow w) \theta(d_k \rightarrow h)$. Typically the number of documents, words in the vocabulary, and latent class variables is on the order of 100,000, 10,000 and hundreds, respectively. Thus latent class variables function as bottleneck variables to constrain word occurrences in documents. Similar generative and Bayesian models for PLSA were also developed by Pritchard et al. [42] for analyzing population structure using multilocus genotype data.

When a PCFG is combined with a trigram model and PLSA, the grammar becomes context sensitive. If we view each uvw trigram as $uv \rightarrow w$, where $u, v, w \in \mathcal{V}$, then the composite trigram/syntactic/semantic language model can be represented as a directed MRF model, where the generation of nonterminals remains the same as in PCFG, but the generation of each terminal depends additionally on its surrounding context; i.e., not only its parent nonterminal but also the preceding two words as well as its semantic content nodes. In this case the sentence “start” symbol generates random trees with *trigrams linking leaf nodes*, the document node generates categorical table with *trigrams as random walk nodes*. In analogy with an auto-regressive HMM [8], the combined syntactic trigram model is in fact an auto-regressive PCFG; the combined semantic trigram model is in fact an auto-regressive PLSA; see Figure 2.

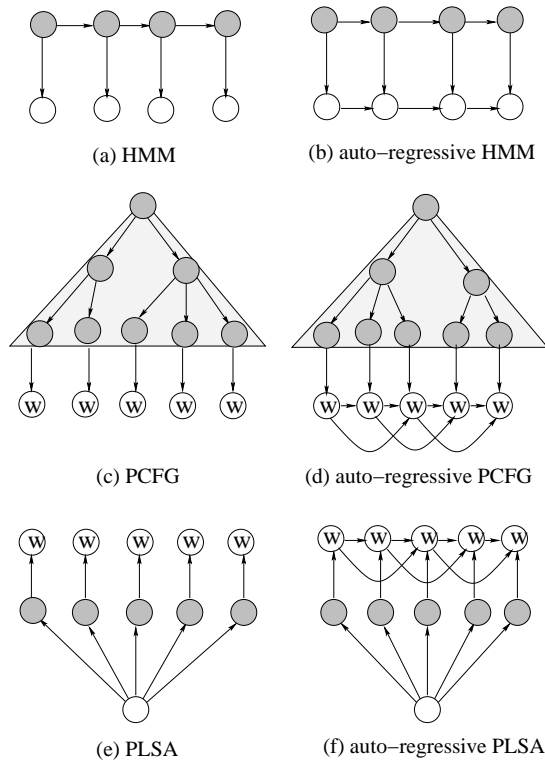


Figure 2: Comparing (a) HMM vs. (b) auto-regressive HMM, and (c) PCFG vs. (d) auto-regressive PCFG models, and (e) PLSA v.s. (f) auto-regressive PLSA. Our composite trigram/syntactic/semantic model is in fact an auto-regressive PCFG-PLSA.

Just as the inside-outside algorithm for PCFGs is the natural counterpart of the forward-backward algorithm for HMMs, the generalized inside-outside algorithm for the combined trigram/PCFG/PLSA model we derive below is the natural counterpart of the forward-backward algorithm for auto-regressive HMMs [8].

3 Training Algorithm for the Composite Language Model

We are interested in learning a composite trigram/syntactic/semantic model from data. We assume we are given a training corpus \mathcal{W} consisting of a collection of documents \mathcal{D} , where each document contains a collection of sentences, and each sentence W is composed of a sequence of words from a vocabulary \mathcal{V} . For simplicity, but without loss of generality, we assume that the PCFG component of the composite model is in Chomsky normal form. That is, each rule is either of the form $A \rightarrow BC$ or $A \rightarrow w$ where $B, C \in \Sigma, w \in \mathcal{V}$. When combined with trigram and PLSA models, the terminal production rule $A \rightarrow w$ becomes $uvAh \rightarrow w$. By examining Figure 1, it should be clear that the likelihood of the observed data under this composite model can be written as below:

$$\begin{aligned} \mathcal{L}(\mathcal{W}, \theta) &= \prod_{d \in \mathcal{D}} \left(\prod_l p_\theta(d, W_l) \right) \\ &= \prod_{d \in \mathcal{D}} \left(\prod_l \left(\sum_t \prod_{u,v \in \mathcal{V}, A \rightarrow w \in \mathcal{R}} \left(\sum_{h \in \mathcal{H}} \theta(d \rightarrow h) \theta(uvAh \rightarrow w) \right)^{n(uvwA; d, W_l, t)} \right. \right. \\ &\quad \left. \left. \prod_{A \rightarrow BC \in \mathcal{R}} \theta(A \rightarrow BC)^{n(A \rightarrow BC; d, W_l, t)} \right) \right) \end{aligned} \quad (2)$$

where $p_\theta(d, W_l)$ is the probability of generating sentence W_l in document d , $n(uvwA; d, W_l, t, h)$ is the count of trigrams uvw with non-terminal symbol A in sentence W_l of document d with parse tree t and $n(A \rightarrow BC; d, W_l, t)$ is the count of nonterminal production rule $A \rightarrow BC$ in sentence W_l of document d with parse tree t . The parameters $\theta(d \rightarrow h), \theta(uvAh \rightarrow w), \theta(A \rightarrow BC)$ are normalized so that

$$\begin{aligned} \sum_{w \in \mathcal{V}} \theta(uvAh \rightarrow w) &= 1 \\ \sum_{BC \in \Sigma} \theta(A \rightarrow BC) &= 1 \\ \sum_{h \in \mathcal{H}} \theta(d \rightarrow h) &= 1 \end{aligned} \quad (3)$$

Thus we have a constrained optimization problem, and there will be a Lagrange multiplier for $uvAh$, non-terminal A and document d .

3.1 Estimating Parameters of the Composite Model

At a first glance, it seems that estimating parameters of the composite model is intractable since the composite directed MRF model is a kind of context sensitive grammar [23] and potentially has exponential number of loops, which suggests that loopy belief propagation [41, 55] and/or variational approximation methods [52] have to be used. It turns out that this is not the case. It is well known that many problems for context sensitive grammars are NP hard [19, 23], however for a subclass of context sensitive grammars, as we will shown below there is an efficient cubic time and exact recursive EM iterative optimization procedure to perform this task.

Following Lafferty's [30] derivation of the inside-outside formulas for updating the PCFG parameters from a general EM [18] algorithm, we derive the generalized inside-outside algorithm for the composite trigram/syntactic/semantic model. For each sentence, since the actual tree used to derive each sentence and semantic content of each word are hidden, we need to apply an iterative EM procedure to obtain a local maximum likelihood estimate for the composite model parameters. Starting at some initial parameters θ , the generalized inside-outside algorithm reestimates the parameters to obtain new parameters θ' such that $\mathcal{L}(\mathcal{W}, \theta') \geq \mathcal{L}(\mathcal{W}, \theta)$. The process is repeated until the likelihood has converged to a local maximum. It turns out that the E-step for inside-outside algorithm for context free grammar needs to be generalized to take into account the trigram model and PLSA model.

To apply the EM algorithm, we consider the auxiliary function

$$Q(\theta', \theta) = \sum_d \sum_l \sum_{H_l} \sum_t p_\theta(H_l, t|d, W_l) \log \frac{p_{\theta'}(d, W_l, H_l, t)}{p_\theta(d, W_l, H_l, t)} \quad (4)$$

where

$$p_\theta(d, W_l, H_l, t) = \prod_{h \in \mathcal{H}} \theta(d \rightarrow h)^{n(d, W_l, h)} \prod_{u, v \in \mathcal{V}, A \rightarrow w \in \mathcal{R}, h \in \mathcal{H}} \theta(uvAh \rightarrow w)^{n(uvAh \rightarrow w; d, W_l, t, h)} \prod_{A \rightarrow BC \in \mathcal{R}} \theta(A \rightarrow BC)^{n(A \rightarrow BC; d, W_l, t)}$$

where $n(d, W_l, h)$ is the count of semantic content h in sentence W_l of the document d , $n(uvAh \rightarrow w; d, W_l, t, h)$ is the count of trigrams uvw , the non-terminal symbol A and semantic content h in sentence W_l of document d with parse tree t and H_l is the semantic content sequence of the sentence W_l .

Taking the derivative of $Q(\theta', \theta)$ with respect to $\theta'(A \rightarrow BC)$ gives

$$\frac{\partial Q(\theta', \theta)}{\partial \theta'(A \rightarrow BC)} = \sum_{d \in \mathcal{D}} \sum_l \sum_{H_l} \sum_t \frac{p_\theta(H_l, t|d, W_l) n(A \rightarrow BC; d, W_l, t)}{\theta'(A \rightarrow BC)}$$

Similarly, taking the derivative of $Q(\theta', \theta)$ with respect to $\theta'(uvAh \rightarrow w)$ gives

$$\frac{\partial Q(\theta', \theta)}{\partial \theta'(uvAh \rightarrow w)} = \sum_{d \in \mathcal{D}} \sum_l \sum_{H_l} \sum_t \frac{p_\theta(H_l, t|d, W_l) n(uvAh \rightarrow w; d, W_l, t, h)}{\theta'(uvAh \rightarrow w)}$$

and taking the derivative of $Q(\theta', \theta)$ with respect to $\theta'(d \rightarrow h)$ gives

$$\frac{\partial Q(\theta', \theta)}{\partial \theta'(d \rightarrow h)} = \sum_{d \in \mathcal{D}} \sum_l \sum_{H_l} \sum_t \frac{p_\theta(H_l, t|d, W_l) n(d \rightarrow h; d, W_l, h)}{\theta'(d \rightarrow h)}$$

Because of the normalization constraints (2), the reestimated parameters of the composite model are then the normalized conditional expected counts:

$$\begin{aligned} \theta'(A \rightarrow BC) &= \frac{\sum_{d \in \mathcal{D}} \sum_l \sum_{H_l} \sum_t p_\theta(H_l, t|d, W_l) n(A \rightarrow BC; d, W_l, t)}{\sum_{B, C \in \Sigma} \sum_{d \in \mathcal{D}} \sum_l \sum_{H_l} \sum_t p_\theta(H_l, t|d, W_l) n(A \rightarrow BC; d, W_l, t)} \\ \theta'(uvAh \rightarrow w) &= \frac{\sum_{d \in \mathcal{D}} \sum_l \sum_{H_l} \sum_t p_\theta(H_l, t|d, W_l) n(uvAh \rightarrow w; d, W_l, t, h)}{\sum_{w \in \mathcal{R}} \sum_{d \in \mathcal{D}} \sum_l \sum_{H_l} \sum_t p_\theta(H_l, t|d, W_l) n(uvAh \rightarrow w; d, W_l, t, h)} \\ \theta'(d \rightarrow h) &= \frac{\sum_l \sum_{H_l} \sum_t p_\theta(H_l, t|d, W_l) n(d \rightarrow h; d, W_l, h)}{\sum_{h \in \mathcal{H}} \sum_l \sum_{H_l} \sum_t p_\theta(H_l, t|d, W_l) n(d \rightarrow h; d, W_l, h)} \end{aligned} \quad (5)$$

This looks the same as the PCFG model [30].

Thus we need to compute the conditional expected counts:

$$\begin{aligned} &\sum_{d \in \mathcal{D}} \sum_l \sum_{H_l} \sum_t p_\theta(H_l, t|d, W_l) n(A \rightarrow BC; d, W_l, t) \\ &\sum_{d \in \mathcal{D}} \sum_l \sum_{H_l} \sum_t p_\theta(H_l, t|d, W_l) n(uvAh \rightarrow w; d, W_l, t, h) \\ &\sum_l \sum_{H_l} \sum_t p_\theta(H_l, t|d, W_l, h) n(d \rightarrow h; d, W_l, h) \end{aligned}$$

In general, the sum requires summing over an exponential number of parse trees. However, just as with standard PCFGs, it is easy to check that the following equations still hold

$$\begin{aligned} \sum_{H_l} \sum_t p_\theta(H_l, t|d, W_l) n(A \rightarrow BC; d, W_l, t) &= \frac{\theta(A \rightarrow BC)}{p_\theta(d, W_l)} \frac{\partial p_\theta(d, W_l)}{\partial \theta(A \rightarrow BC)} \\ \sum_{H_l} \sum_t p_\theta(H_l, t|d, W_l) n(uvAh \rightarrow w; d, W_l, t, h) &= \frac{\theta(uvAh \rightarrow w)}{p_\theta(d, W_l)} \frac{\partial p_\theta(d, W_l)}{\partial \theta(uvAh \rightarrow w)} \\ \sum_{H_l} \sum_t p_\theta(H_l, t|d, W_l) n(d \rightarrow h; d, W_l, h) &= \frac{\theta(d \rightarrow h)}{p_\theta(d, W_l)} \frac{\partial p_\theta(d, W_l)}{\partial \theta(d \rightarrow h)} \end{aligned}$$

and it turns out that there is an efficient way of computing the partial derivative on the righthand side, *the generalized inside-outside algorithm*.

Let $A \Rightarrow \gamma$ denote that, beginning with a nonterminal A , we can derive a string γ of words and non-terminals by applying a sequence of rewrite rules from the grammar *without the flowing-out trigrams but with the flowing-in and in-between trigrams and PLSA nodes*, where flowing-in and flowing-out trigrams are those that at least one word not belong to γ , in-between trigrams are those that all words belong to γ and in-between PLSA nodes are those that the word node belongs to γ .

Suppose the position of a rule $A \rightarrow BC$ within a tree t for sentence $W_l = (w_1, \dots, w_N)$ in document d can be specified by a triple $(i, j, k), i \leq j \leq k$. The partial derivative of the probability $p_\theta(S \rightarrow W_l \text{ in } d) = p_\theta(d, W_l)$ with respect to the parameter $\theta(A \rightarrow BC)$ only involves those parse trees which use the rule $A \rightarrow BC$. Consider the event “ $S \rightarrow W_l$ in d using $A \rightarrow BC$ in position (i, j, k) ”. Because of the Markov property of the directed MRF model, the probability of this event can be written as a product of four terms, i.e., *the factorization property*, as follows:

$$\begin{aligned} & p_\theta(S \rightarrow W_l \text{ in } d; \text{ using } A \rightarrow BC \text{ in position } (i, j, k)) \\ &= \theta(A \rightarrow BC) p_\theta(B \Rightarrow w_i \cdots w_j; W_l \text{ in } d) \\ & \quad p_\theta(C \Rightarrow w_{j+1} \cdots w_k; W_l \text{ in } d) \\ & \quad p_\theta(S \Rightarrow w_1 \cdots w_{i-1} A w_{k+1} \cdots w_N; W_l \text{ in } d) \end{aligned}$$

See Figure 3 for an illustration. The *key insight* toward a solution for the composite model is that, in comparison with the PCFG model, there are additional trigrams which connect the decomposition in position (i, j, k) . These dependencies encode additional information from the trigram model, and significantly influence the parameter estimation of the nonterminal grammatical production rules (the impact of the PLSA model is implicitly considered, this will become clear when we derive the estimation formula for the terminal grammatical production rules). The factorization property is the crucial constituent for the success to derive an efficient and exact recursive algorithm.

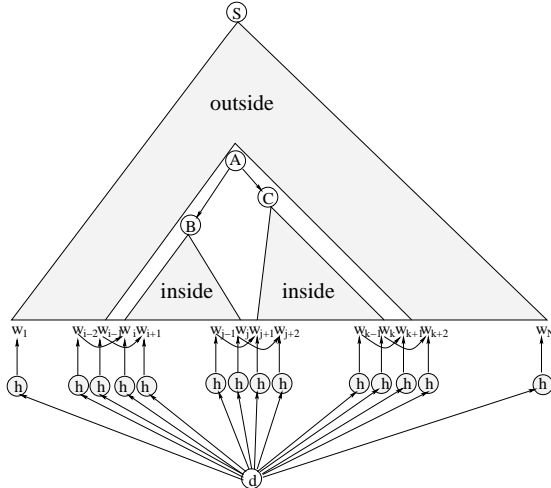


Figure 3: Inside and outside probabilities for nonterminal production rule in the composite language model, where each component is influenced by the injected trigram and PLSA models.

From this it is not difficult to see that

$$\frac{\partial p_\theta(S \rightarrow W_l \text{ in } d)}{\partial \theta(A \rightarrow BC)} = \sum_{i \leq j \leq k} p_\theta(B \Rightarrow w_i \cdots w_j; W_l \text{ in } d) p_\theta(C \Rightarrow w_{j+1} \cdots w_k; W_l \text{ in } d) p_\theta(S \Rightarrow w_1 \cdots w_{i-1} A w_{k+1} \cdots w_N; W_l \text{ in } d)$$

Thus, the conditional expected number of times that the rule $A \rightarrow BC$ is used in generating the sentence

$W_l \in \mathcal{W}$ in document d using the model θ is given by

$$\begin{aligned} & \sum_{H_l} \sum_t p_\theta(H_l, t | d, W_l) n(A \rightarrow BC; d, W_l, t) \\ = & \frac{\theta(A \rightarrow BC)}{p_\theta(W_l \text{ in } d)} \left(\sum_{i \leq j \leq k} \beta_{ik}(A; W_l \text{ in } d) \alpha_{ij}(B; W_l \text{ in } d) \alpha_{j+1k}(C; W_l \text{ in } d) \right) \end{aligned}$$

where

$$\alpha_{ij}(A; W_l \text{ in } d) = p_\theta(A \Rightarrow w_i \cdots w_j; W_l \text{ in } d)$$

i.e., the inside probability that the nonterminal A and the trigram parent nodes of w_i, w_{i+1} and document node d derive the word subsequence $w_i \cdots w_j$ in the sentence W_l of document d ; and

$$\beta_{ik}(B; W_l \text{ in } d) = p_\theta(S \Rightarrow w_1 \cdots w_{i-1} A w_{k+1} \cdots w_N; W_l \text{ in } d)$$

i.e., the outside probability that beginning with the start symbol S , trigram parent nodes of w_{k+1}, w_{k+2} and document node d , we can derive the sequence $w_1 \cdots w_{i-1} A w_{k+1} \cdots w_N$ in the sentence W_l of document d . See Figure 4 for illustration.

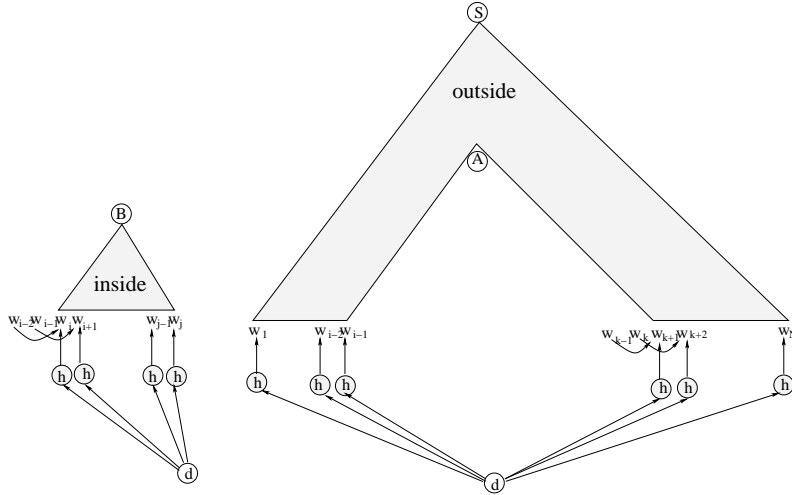


Figure 4: Inside probability α_{ij} and outside probability β_{ik} in the composite language model.

Similarly consider the event “ $S \rightarrow W_l$ using $uvAh \rightarrow w$ in d in position (i)”. Because of the Markov property of the directed Markov random field model, the probability of this event can be written as a product of four terms, again *the factorization property*, as follows:

$$\begin{aligned} & p_\theta(S \rightarrow W_l \text{ in } d; \text{ using } uvAh \rightarrow w \text{ in position } (i)) \\ = & \delta_{uvw}(w_{i-2}w_{i-1}w_i) \left(\theta(d \rightarrow h) \theta(uvAh \rightarrow w) \right) \\ & p_\theta(S \Rightarrow w_1 \cdots w_{i-1} A w_{i+1} \cdots w_N; W_l \text{ in } d) \end{aligned}$$

See Figure 5 for illustration. Again the *key insight* toward a solution for the composite model is that comparing with the PCFG model, there are additional trigram and PLSA nodes which connect the decomposition in position (i) to encode the information of both trigram and PLSA models and make influential impact for parameter estimation of the grammatical production rules $uvAh \rightarrow w$. Again, the factorization property is the crucial constituent for the success to derive an efficient and exact recursive algorithm.

Thus we have

$$\sum_{H_l} \sum_t p_\theta(H_l, t | d, W_l) n(uvAh \rightarrow w; d, W_l, t)$$

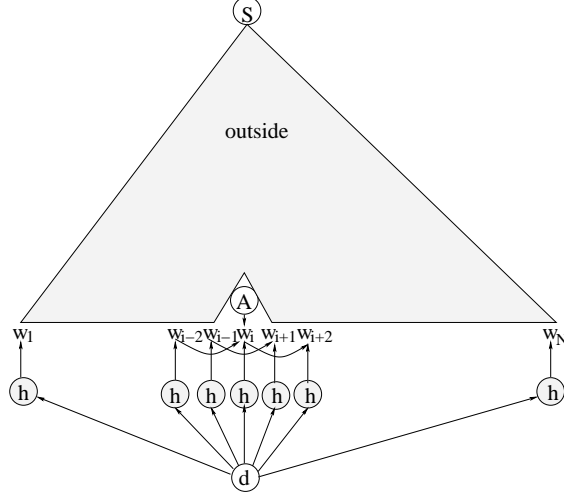


Figure 5: Inside and outside probabilities for terminal production rule in the composite language model, where each component is influenced by the injected trigram and PLSA models.

$$= \frac{\theta(uvAh \rightarrow w)}{p_\theta(W_l \text{ in } d)} \sum_{1 \leq i \leq N} \delta_{uvhw}(w_{i-2}w_{i-1}w_i)\theta(d \rightarrow h)\beta_{ii}(A; W_l \text{ in } d)$$

where δ is the indicator function.

Similarly consider the event “ $d \rightarrow W_l$ in d using $d \rightarrow h$ in position (i)”. Because of the Markov property of the directed Markov random field model, the probability of this event can be written as a product of three terms as follows:

$$\begin{aligned} & p_\theta(S \rightarrow W_l \text{ in } d; \text{ using } d \rightarrow h \text{ in position } (i)) \\ &= \sum_{A \in \Sigma} p_\theta(S \Rightarrow w_1 \cdots w_{i-1}Aw_{i+1} \cdots w_N; W_l \text{ in } d) \\ & \quad \left(\theta(d \rightarrow h)\theta(w_{i-2}w_{i-1}Ah \rightarrow w_i) \right) \end{aligned}$$

Thus we have

$$\begin{aligned} & \sum_{H_l} \sum_t p_\theta(H_l, t | d, W_l) n(d \rightarrow h; d, W_l) \\ &= \frac{\theta(d \rightarrow h)}{p_\theta(W_l \text{ in } d)} \sum_{1 \leq i \leq N} \sum_{A \in \Sigma} \theta(w_{i-2}w_{i-1}Ah \rightarrow w_i)\beta_{ii}(A; W_l \text{ in } d) \end{aligned}$$

Just as in the PCFG case, there is an efficient recursive method for computing the α 's and β 's using the CKY chart-parsing algorithm [56]. The method for doing this is almost the same as for PCFG and is implicit in the following recursive formulas:

$$\begin{aligned} \alpha_{ij}(A; W_l \text{ in } d) &= \sum_{BC} \sum_{i \leq k \leq j} \left(\theta(A \rightarrow BC)\alpha_{ik}(A; W_l \text{ in } d)\alpha_{k+1j}(C; W_l \text{ in } d) \right) \\ \alpha_{ii}(A; W_l \text{ in } d) &= \sum_h \left(\theta(d \rightarrow h)\theta(w_{i-2}w_{i-1}Ah \rightarrow w_i) \right) \\ \beta_{ij}(A; W_l \text{ in } d) &= \left(\sum_{B,C} \sum_{k < i} \theta(B \rightarrow CA)\alpha_{ki-1}(C; W_l \text{ in } d)\beta_{kj}(B; W_l \text{ in } d) \right. \\ & \quad \left. + \sum_{B,C} \sum_{k > j} \theta(B \rightarrow AC)\alpha_{j+1k}(C; W_l \text{ in } d)\beta_{ik}(B; W_l \text{ in } d) \right) \\ \beta_{1N}(A; W_l \text{ in } d) &= \delta_S(A; W_l \text{ in } d) \end{aligned}$$

Comparing with the case of PCFGs, we notice that the only modification is in the definition of α_{ii} which explicitly encode trigram and PLSA models, and then propagate their effects to the rest of α s and β s via the above recursive formulas.

Chi et al. [14, 15] proved that the maximum likelihood estimate of production rule probabilities for a PCFG yield a *proper distribution*, i.e., there is no probability mass lost to infinitely large trees [10]. Similarly we can show that the maximum likelihood estimate of production rule probabilities for this composite trigram/syntactic/semantic model always yield a proper distribution.

Theorem 1 *Let Ω be the set of finite parse trees, \hat{p} be any intermediate iteration of the EM procedure within the generalized inside-outside algorithm. Then $\hat{p}(\Omega) = 1$.*

Proof: See appendix. ■

It is easy to modify the algorithm to extract the most probable parse for a given sentence under the composite model. Briefly speaking, when we add each nonterminal A to a box (i, j) in a chart, we keep a record of which has the highest likelihood of rewriting A . That is, we determine which nonterminals B, C and index k maximize the probability $\theta(A \rightarrow BC)\alpha_{ik}(B)\alpha_{k+1,j}(C)$. When we reach the topmost nonterminal S , we can then “traceback” to construct the Viterbi parse. Comparing with the case of PCFGs, the only difference is in the definition of α_{ii} .

3.2 Smoothing Techniques of the Composite Model

One severe problem in language modeling is the so called sparse data problem caused by the sparseness of n -gram counts appeared in the training data. To combat the sparse data problem in language modeling, various smoothing techniques [13] have been proposed. Basically smoothing is the technique to adjust the maximum likelihood estimate to prevent zero probabilities with the attempt to produce more accurate estimate. Current smoothing techniques only handle explicit counts, but in our case there are hidden variables A and h in parameter estimation formula for $\theta(uvAh \rightarrow w)$. In this section, we show how to extend smoothing methods to situations where there exist hidden variables.

Notice that the sparse data problem arises from trigram counts. The Good-Turing estimate [20] is central to combat this problem. The Good-Turing estimate states that for any trigram that occurs n times, we should pretend that it occurs n^* times where

$$n^* = (n + 1) \frac{r_{n+1}}{r_n} \quad (6)$$

where r_n is the number of trigrams that occur exactly n times in the training data. To convert this count to a probability, we just normalize: for a trigram uvw with n counts, we take

$$P_{GT}(uvw) = \frac{n^*}{N} \quad (7)$$

where $N = \sum_{n=0}^{\infty} r_n n^*$. In practice, the Good-Turing estimate is not used by itself, instead it is often enhanced by back-off technique to combine higher-order models with lower-order models necessary for good performance.

A procedure of replacing a count n with a modified count n^* is called “discount” and we define the ratio $\rho_n = \frac{n^*}{n}$ as a discount coefficient ρ_n . The ρ_n are calculated as follows: large counts are taken to be reliable, so they are not discounted. In particular, Katz [28] takes $\rho_n = 1$ for all $n \geq k$ for some k . The discount ratios for the lower counts $n \leq k$ are derived from the Good-Turing estimate applied to the global trigram distribution and is given as

$$\rho_n = \frac{\frac{n^*}{n} - \frac{(k+1)r_{k+1}}{r_1}}{1 - \frac{(k+1)r_{k+1}}{r_1}} \quad (8)$$

When we use (5) to estimate $\theta(uvAh \rightarrow w)$, we use the expected count of $n(uvAh \rightarrow w)$ where A and h are hidden. However, when the trigram uvw has count $n(uv \rightarrow w) > 0$, if we discount the expected count

of $n(uvAh \rightarrow w)$ by the ratio $\rho_n(uvw)$, then we discount the trigrams by the same ratio $\rho_n(uvw)$ since $\sum_{A \in \Sigma, h \in \mathcal{H}} \rho_n(uvw) n(uvAh \rightarrow w) = \rho_n(uvw) n(uv \rightarrow w)$. Therefore instead of using iterative parameter estimation of (4), we use smoothed iterative parameter estimation as the following,

$$\theta'_s(uvAh \rightarrow w) = \frac{\rho_n(uvw) \sum_{d \in \mathcal{D}} \sum_l \sum_{H_l} \sum_t p_\theta(H_l, t | d, W_l) n(uvAh \rightarrow w; d, W_l, t, h)}{\sum_{w \in \mathcal{R}} \rho_n(uvw) \sum_{d \in \mathcal{D}} \sum_l \sum_{H_l} \sum_t p_\theta(H_l, t | d, W_l) n(uvAh \rightarrow w; d, W_l, t, h)}$$

When the trigram uvw has count $n(uv \rightarrow w) = 0$, we backoff to the corresponding bigram parameters and let

$$\theta_s(uvAh \rightarrow w) = \eta(uvw) \cdot \theta_s(vAh \rightarrow w)$$

and

$$\eta(uvw) = \frac{1 - \sum_{w: n(uvw) > 0} \theta_s(uvAh \rightarrow w)}{1 - \sum_{w: n(uvw) > 0} \theta_s(vAh \rightarrow w)}$$

Similarly we can use Kneser-Ney smoothing [38, 39], and the smoothed iterative parameter estimation as the following,

$$\theta'_s(uvAh \rightarrow w) = \frac{n_{1+}(\cdot vw) \sum_{d \in \mathcal{D}} \sum_l \sum_{H_l} \sum_t p_\theta(H_l, t | d, W_l) n(uvAh \rightarrow w; d, W_l, t, h)}{\sum_{w \in \mathcal{R}} n_{1+}(\cdot vw) \sum_{d \in \mathcal{D}} \sum_l \sum_{H_l} \sum_t p_\theta(H_l, t | d, W_l) n(uvAh \rightarrow w; d, W_l, t, h)}$$

where $n_{1+}(\cdot vw) = |\{u : c(uvw) > 0\}|$ is the number of different words u that precede vw in the training data.

4 Computing the Probability of Initial Subsequence Generation

In automatic speech recognition or statistical machine translation, we are presented with words one at a time, in sequence. Therefore, we would like to calculate the probability $p_\theta(S \rightarrow w_1 w_2 \cdots w_k \cdots)$; that is, the probability that an arbitrary word sequence $w_1 w_2 \cdots w_k$ is the initial subsequence of a sentence generated by the composite trigram/syntactic/semantic language model. We derive the *generalized left-to-right inside* algorithm to perform this computation by following the work of [25], which assumes that a PCFG model is used.

In the following, we basically use the same notation as in [25]. Let $p_\theta(A \ll i, j)$ denote the sum of the probabilities of all trees with root node A and document node d resulting in word sequences whose initial subsequence is $w_i \cdots w_j$. Thus

$$\begin{aligned} p_\theta(A \ll i, j) &= \alpha_{ij}(A) + \sum_{x_1 \in \mathcal{V}} p_\theta(A \rightarrow w_i \cdots w_j x_1) + \sum_{x_1 x_2 \in \mathcal{V}^2} p_\theta(A \rightarrow w_i \cdots w_j x_1 x_2) + \cdots \\ &+ \sum_{x_1 \cdots x_n \in \mathcal{V}^n} p_\theta(A \rightarrow w_i \cdots w_j x_1 \cdots x_n) + \cdots \end{aligned} \quad (9)$$

Using this notation, the desired probability $p_\theta(S \rightarrow w_1 w_2 \cdots w_k \cdots)$ is denoted by $p_\theta(S \ll 1, k)$.

Let $p_\theta^L(A \rightarrow B) = \sum_{B_2 \in \Sigma} p_\theta^L(A \rightarrow B_1 B_2)$ be the sum of the probabilities of all the rules $A \rightarrow B_1 B_2$ whose first lefthand side element is $B_1 = B$. Define $p_\theta^L(A \Rightarrow B) = \sum_{\gamma \in (\Sigma \cup \mathcal{V})^*} p_\theta(A \Rightarrow B \gamma)$ as the sum of probabilities of all trees with root node A that produce B as the leftmost first nonterminal. This term converges, since our underlying composite syntactic/semantic/trigram model p_θ is proper.

Using this definition, we get

$$\begin{aligned}
p_\theta(A \ll i, i) &= \sum_{h \in \mathcal{H}} p_\theta(w_{i-2} w_{i-1} A h \rightarrow w_i) p_\theta(d \rightarrow h) \\
&\quad + \sum_{B \in \Sigma} p_\theta^L(A \Rightarrow B) \sum_{h \in \mathcal{H}} p_\theta(w_{i-2} w_{i-1} B h \rightarrow w_i) p_\theta(d \rightarrow h) \\
&= \alpha_{i,i}(A) + \sum_{B \in \Sigma} p_\theta^L(A \Rightarrow B) \alpha_{i,i}(B)
\end{aligned} \tag{10}$$

Define the sum of probabilities of all trees with root node A whose last leftmost production results in leaves B_1 and B_2 as

$$p_\theta^L(A \Rightarrow B_1 B_2) = p_\theta(A \rightarrow B_1 B_2) + \sum_{C \in \Sigma} p_\theta^L(A \Rightarrow C) p_\theta(C \rightarrow B_1 B_2) \tag{11}$$

Obviously,

$$\begin{aligned}
p_\theta(A \ll i, i+n) &= \sum_{B, B_2 \in \Sigma} p_\theta^L(A \rightarrow B_1 B_2) \left(\alpha_{i,i}(B_1) p_\theta(B_2 \ll i+1, i+n) + \alpha_{i,i+1}(B_1) p_\theta(B_2 \ll i+2, i+n) + \dots \right. \\
&\quad \left. + \alpha_{i,i+n-1}(B_1 \Rightarrow w_i \dots w_{i+n-1}) p_\theta(B_2 \ll i+n, i+n) + p_\theta(B_1 \ll i, i+n) \right)
\end{aligned}$$

since to generate the initial subsequence $w_i w_{i+1} \dots w_{i+n}$, some rule $A \rightarrow B_1 B_2$ must first be applied and then the first part of the subsequence must be generated from B_1 and its remaining part from B_2 .

Define the function

$$\begin{aligned}
R(B_1, B_2) &= \left(\alpha_{i,i}(B_1) p_\theta(B_2 \ll i+1, i+n) + \alpha_{i,i+1}(B_1) p_\theta(B_2 \ll i+2, i+n) + \dots \right. \\
&\quad \left. + \alpha_{i,i+n-1}(B_1 \Rightarrow w_i \dots w_{i+n-1}) p_\theta(B_2 \ll i+n, i+n) + p_\theta(B_1 \ll i, i+n) \right)
\end{aligned}$$

Then following the same recursive derivation as in [25], we have

$$\begin{aligned}
p_\theta(A \ll i, i+n) &= \sum_{B, B_2 \in \Sigma} p_\theta^L(A \Rightarrow B_1 B_2) R(B_1, B_2) + \sum_{B_1 \in \Sigma} p_\theta^L(A \rightarrow B_1) p(B_1 \ll i, i+n) \\
&= \sum_{B, B_2 \in \Sigma} \left[p_\theta(A \rightarrow B_1 B_2) + \sum_{C_1 \in \Sigma} p_\theta^L(A \rightarrow C_1) p_\theta(C_1 \rightarrow B_1 B_2) \right] R(B_1, B_2) \\
&\quad + \sum_{C_1, C_2 \in \Sigma} p_\theta^L(A \rightarrow C_1) p_\theta^L(C_1 \rightarrow C_2) p(C_2 \ll i, i+n) \\
&= \dots \dots \dots \\
&= \sum_{B, B_2 \in \Sigma} p_\theta^L(A \Rightarrow B_1 B_2) R(B_1, B_2) + \sum_{C_1, \dots, C_k \in \Sigma, k \rightarrow \infty} \left(p_\theta^L(A \rightarrow C_1) \prod_{l=2}^k p_\theta^L(C_{l-1} \rightarrow C_l) p(C_k \ll i, i+n) \right)
\end{aligned}$$

We have shown that the maximum likelihood estimate of the composite language yields a proper distribution in Theorem 1, thus the last term of the above equation tends to 0 as k grows without limit. Then using definition (11) and successive resubstitutions, we get the final formula

$$\begin{aligned}
p_\theta(A \ll i, i+n) &= \sum_{B, B_2 \in \Sigma} p_\theta^L(A \Rightarrow B_1 B_2) R(B_1, B_2) \\
&= \sum_{B, B_2 \in \Sigma} p_\theta^L(A \Rightarrow B_1 B_2) \left(\sum_{j=1}^n \alpha_{i,i+j-1}(B_1) p_\theta(B_2 \ll i+j, i+n) \right)
\end{aligned} \tag{12}$$

Comparing with a PCFG, the only difference is the way that $R(B_1, B_2)$ is recursively calculated by α , which here takes into account the impact of the trigram and PLSA models.

The desired probability $p(S \rightarrow w_1, \dots, w_n, \dots)$ can thus be calculated exactly in the same recursive way as in the PCFG case, which is described in [25].

5 Experimental Evaluation

5.1 Experimental data sets and performance measure

The corpus used to train our model was taken from the WSJ portion of the NAB corpus, which was composed of about 150,000 documents spanning the years 1987 to 1989, comprising approximately 42 millions words. The vocabulary was constructed by taking the 20,000 most frequent words of the training data. The PCFG production rules we use are extracted from the sections 2-21 of the WSJ treebank corpus. We split another separate set of data consisting of 325,000 words taken from the year 1989 into half, one half used as development data by random selection, another half for testing.

	NO. OF ARTICLES	NO. OF SENTENCES	NO. OF WORDS
TRAIN	150,981	1,611,571	41,780,924
DEV	378	6904	157,312
TEST	379	6638	153,801

Table 1: Data sets statistics

The statistics of the datasets are shown in Table I.

To evaluate a language model we use the standard definitions of perplexity and entropy on held-out test data. That is, given a test corpus $\mathcal{T} = \{d_1, \dots, d_L\}$ and a language model defining $p(s)$ we calculate test perplexity and test entropy as

$$\begin{aligned}
 \text{Perplexity} &= \sqrt[|\mathcal{T}|]{\frac{1}{p(\mathcal{T})}} = \sqrt[|\mathcal{T}|]{\prod_{i=1}^L \frac{1}{p(w_1 \dots w_{|d_i|}, d_i)}} = \sqrt[|\mathcal{T}|]{\prod_{i=1}^L \prod_{l=1}^M \frac{1}{p(d_i, W_l)}} \\
 \text{Entropy} &= \log_2 \text{Perplexity}
 \end{aligned}$$

where $|\mathcal{T}| = \sum_{i=1}^L |d_i| = \sum_{i=1}^L \sum_{l=1}^M |W_l|$ is the length of test corpus. The goal is to obtain small values of these measures. That is, the goal of language modeling is to predict the probability of natural word sequences; or more simply, to put high probability on word sequences that actually occur (and low probability on word sequences that never occur).

5.2 Computation in Testing

Since the representation for a document of the test data is not contained in the original training corpus, we use similar ‘‘fold-in’’ heuristic approach similar to the one used in [22]: the parameters corresponding to the document-semantic arcs, $\theta(d \rightarrow h)$, are re-estimated by the probability of word subsequence currently seen, w_1, \dots, w_k , i.e., the initial subsequence of a sentence generated by the composite language model, while holding the other parameters fixed.

In this case, we use the recursive gradient ascent to update $\theta(d \rightarrow h)$.

$$\theta(d \rightarrow h)^{(k)} = \theta(d \rightarrow h)^{(k-1)} - \frac{\partial \log p_\theta(S \lll 1, k)}{\partial \theta(d \rightarrow h)} \Big|_{\theta^{(k-1)}}$$

Next we describe how to recursively calculate the gradient of log-likelihood of the initial subsequence of a sentence with respect to the parameters of document-semantic arc. Since

$$\frac{\partial \log p_\theta(S \lll 1, k)}{\partial \theta(d \rightarrow h)} = \frac{1}{p_\theta(S \lll 1, k)} \frac{\partial p_\theta(S \lll 1, k)}{\partial \theta(d \rightarrow h)}$$

$p_\theta(S \lll 1, k)$ can be recursively calculated as described in the last section, so we only describe how to calculate $\frac{\partial p_\theta(S \lll 1, k)}{\partial \theta(d \rightarrow h)}$.

By the formula (13), we have

$$\frac{\partial p_\theta(A \lll i, i + n)}{\partial \theta(d \rightarrow h)}$$

$$\begin{aligned}
&= \sum_{B, B_2 \in \Sigma} p_{\theta}^L(S \Rightarrow B_1 B_2) \frac{\partial \left(\sum_{j=1}^n \alpha_{i, i+j-1}(B_1) p_{\theta}(B_2 \ll i+j, i+n) \right)}{\partial \theta(d \rightarrow h)} \\
&= \sum_{B, B_2 \in \Sigma} p_{\theta}^L(S \Rightarrow B_1 B_2) \sum_{j=1}^n \left(\frac{\partial \alpha_{i, i+j-1}(B_1)}{\partial \theta(d \rightarrow h)} p_{\theta}(B_2 \ll i+j, i+n) + \alpha_{i, i+j-1}(B_1) \frac{\partial p_{\theta}(B_2 \ll i+j, i+k-1)}{\partial \theta(d \rightarrow h)} \right)
\end{aligned}$$

By the recursive formula to calculate α , we can calculate $\frac{\partial \alpha_{i, i+j-1}(B_1)}{\partial \theta(d \rightarrow h)}$ in bottom-up procedure by the following recursive formula

$$\frac{\partial \alpha_{i, i+j-1}(B_1)}{\partial \theta(d \rightarrow h)} = \sum_{C_1 C_2} \sum_{i \leq k \leq i+j-1} \theta(B_1 \rightarrow C_1 C_2) \left(\frac{\partial \alpha_{i, k}(C_1)}{\partial \theta(d \rightarrow h)} \alpha_{k+1, i+j-1}(C_2) + \alpha_{i, k}(C_1) \frac{\partial \alpha_{k+1, i+j-1}(C_2)}{\partial \theta(d \rightarrow h)} \right)$$

and

$$\frac{\partial \alpha_{i, i}(B_1)}{\partial \theta(d \rightarrow h)} = \theta(w_{i-2} w_{i-1} B_1 h \rightarrow w_i)$$

Once all the probabilities required for the computation of $\frac{\partial p_{\theta}(S \ll \leq 1, k)}{\partial \theta(d \rightarrow h)}$ are computed, to get the next probability of interest, $\frac{\partial p_{\theta}(S \ll \leq 1, k+1)}{\partial \theta(d \rightarrow h)}$, we need to compute the following quantities:

1. The probabilities $\frac{\partial \alpha_{i, k}(A)}{\partial \theta(d \rightarrow h)}$ for $i = k, k-1, \dots, 1$, in that order.
2. The probabilities $\frac{\partial p_{\theta}(B \ll \leq i, k+1)}{\partial \theta(d \rightarrow h)}$ for $i = k+1, k, \dots, 2$, in that order.
3. The probability $\frac{\partial p_{\theta}(S \ll \leq 1, k+1)}{\partial \theta(d \rightarrow h)}$.

5.3 Experimental design

To serve as a baseline standard of performance, we use a conventional trigram model with Good-Turing back-off and Kneser-Ney smoothing. Implementing these approaches, we obtained perplexity scores of 109 and 103 respectively on development data set.

When we train the PCFG model alone, the perplexity score on development data is 678. Combining the PCFG model with Good-Turing back-off and Kneser-Ney smoothing trigram models by linear interpolation, we obtain the test perplexity score 109 and 102 respectively. Next we train the PLSA model alone where the number of hidden semantic nodes h is set to be $|\mathcal{H}| = 125$, we obtain perplexity score on development data 1487. When this PLSA model is combined with Good-Turing back-off and Kneser-Ney smoothing trigram models by linear interpolation, we find that the test perplexity scores remain unchanged. If we combine these three models together using linear interpolation, we obtain the perplexity scores on development data 108 and 102 respectively.

Next we introduce the composite syntactic/trigram model which is equivalent to the composite syntactic/semantic/trigram language model by setting the semantic node h to be a constant. Using the generalized inside-outside algorithm to train this composite syntactic/trigram model with Good-Turing back-off and Kneser-Ney smoothing trigram models, we achieve a perplexity scores of 94 and 90 on development data of, a 14% and 11% relative reduction in perplexity respectively.

We then introduce the composite semantic/trigram model which is equivalent to the composite syntactic/semantic/trigram language model by setting the syntactic node A to be a constant. We fix the number of possible hidden topics to be $|\mathcal{H}| = 125$ and use the generalized inside-outside algorithm to train the composite semantic/trigram model with Good-Turing back-off and Kneser-Ney smoothing trigram models, here we achieve perplexity scores of 96 and 91 on development data, a 12% and 10% relative reduction in perplexity respectively. Since the representation for a document of the development data is not contained in the original training corpus, during testing we use ‘‘fold-in’’ heuristic approach similar to the one used in [22]: the document-semantic parameters are re-estimated by maximum likelihood estimation while holding semantic-word parameters fixed, where the empirical distribution is given by the current updated document history.

Table 2: Perplexity results for the composite syntactic semantic trigram model on development corpus.

LANGUAGE MODEL	PERPLEXITY	PERPLEXITY
	GOOD-TURING	KNESER-NEY
TRIGRAM (BASELINE)	109	103
PCFG	678	
LINEAR INTERPOLATION OF PCFG & TRIGRAM	109	102
PLSA	1487	
LINEAR INTERPOLATION OF PLSA & TRIGRAM	109	103
LINEAR INTERPOLATION OF PLSA, PCFG & TRIGRAM	108	102
SYNTACTIC TRIGRAM	94	90
SEMANTIC TRIGRAM	96	91
SYNTACTIC, SEMANTIC TRIGRAM	82	79

Finally we use the generalized inside-outside algorithm to train the composite trigram/syntactic/semantic model with Good-Turing back-off and Kneser-Ney smoothing trigram models and we set the number of hidden semantic node h is again set to be $|\mathcal{H}| = 125$. Again since the representation for a document of the development data is not contained in the original training corpus, during testing we use “fold-in” heuristic approach as described in the last subsection: the document-semantic parameters are re-estimated by recursive gradient ascent of maximum likelihood estimation of the initial subsequence of a sentence while holding semantic-word and production rule parameters fixed. This time we achieve perplexity scores of 82 and 79 on development data, a 25% and 21% relative reduction in perplexity respectively.

The perplexity results are listed in Table 2 and the perplexity reductions of these results over baseline trigram models with Good-Turing and Kneser-Ney smoothings are shown in Figure 6. It shows that linear interpolation is too blunt to capture subtleties of PCFG and PLSA models, however our approach of integrating syntactic and semantic sources of nonlocal dependency information from PCFG and PLSA models into trigram model results significant perplexity improvement. Basically PCFG and PLSA models carry complementary long-range dependency structure and their gains over trigram model are almost additive. Another observation is that the gains of using Kneser-Ney smoothing over Good-Turing smoothing are almost additive too.

It has been observed that the number of semantic node is critical to the final perplexity results [3, 54], so next we study how the number of semantic nodes influences our composite language model. We first fix the number of hidden semantic topics, we estimate the parameters of the smoothed composite semantic trigram model as well as the smoothed composite syntactic/semantic/trigram model respectively by using the training data. We then use the development data to test the perplexity results. Figure 7 shows how the perplexities of these estimated models with Good-Turing and Kneser-Ney smoothings change as the number of semantic topics is increased and decreased, with the best perplexities achieved in each case when the number of semantic topics equals $\mathcal{H} = 125$.

Once we use the development corpus to tune the optimal number of semantic topic, for the best composite semantic/trigram model and the best composite syntactic/semantic/trigram model, we calculate the perplexity on test corpus. The perplexity scores by the baseline trigram models with Good-Turing and Kneser-Ney smoothings are 103 and 99 respectively. By using the Good-Turing smoothing, we obtain 91, 92 and 80 perplexity scores respectively by the composite syntactic/trigram model, semantic/trigram model, and syntactic/semantic/trigram models. The corresponding perplexity reductions are 12%, 10% and 23%. When we use Kneser-Ney smoothing, we have perplexity scores 89, 90 and 78 respectively by the composite syntactic/trigram model, semantic/trigram model, and syntactic/semantic/trigram models. The corresponding perplexity reductions are 10%, 9% and 21%. Table 3 summarizes the results on test corpus by these models.

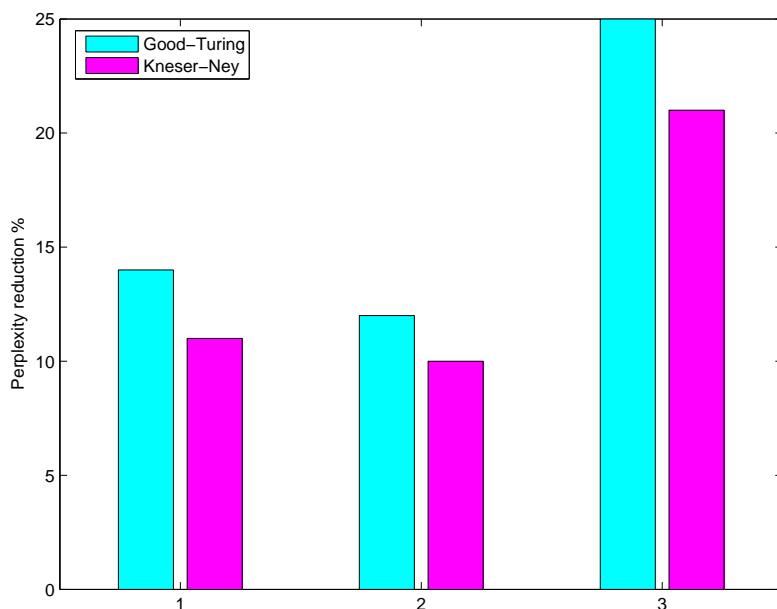


Figure 6: *Relative perplexity reductions over baseline trigram with Good-Turing and Kneser-Ney smoothings by various composite language models: 1. Syntactic-trigram, 2. Semantic-trigram, 3. Syntactic-semantic-trigram.*

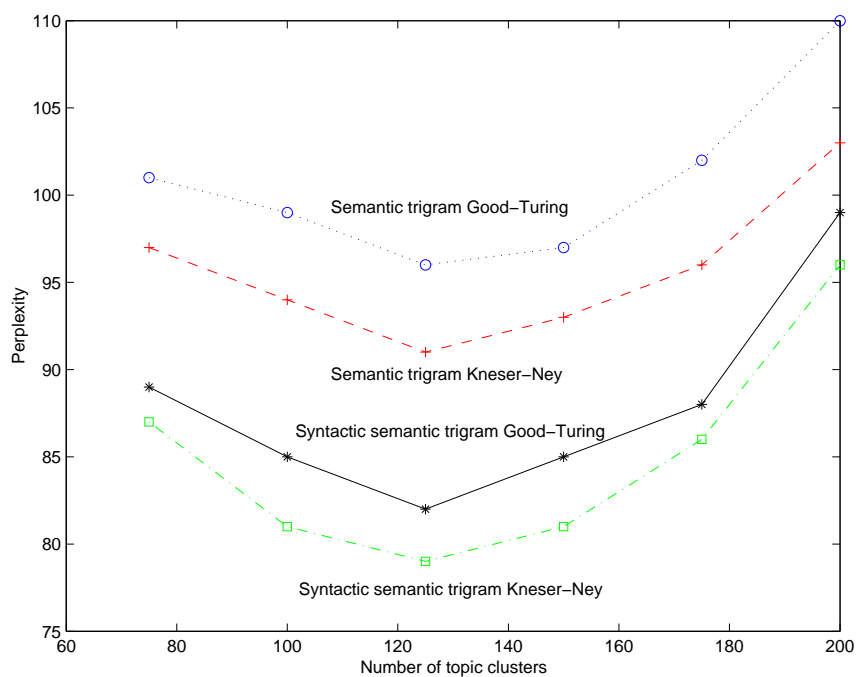


Figure 7: *Perplexities versus number of topics for the semantic/trigram and the syntactic/semantic/trigram language models.*

Table 3: Perplexity results for the syntactic/trigram, semantic/trigram as well as syntactic/semantic/trigram language models on test corpus.

LANGUAGE MODEL	PERPLEXITY	REDUCTION	PERPLEXITY	REDUCTION
	GOOD-TURING		KNESER-NEY	
TRIGRAM (BASELINE)	103		99	
SYNTACTIC TRIGRAM	91	12%	89	10%
SEMANTIC TRIGRAM	92	10%	90	9%
SYNTACTIC, SEMANTIC TRIGRAM	80	23%	78	21%

6 Conclusion and Discussion

We present an original approach that combines n -gram, PCFG and PLSA to build a sophisticated mixed chain/tree/table directed MRF model for statistical language modeling, where various aspects of natural language—such as local word interaction, syntactic structure, and semantic document information—can be modeled by mixtures of exponential families with a rich expressive power that can take their interactions into account simultaneously and automatically. The composite directed MRF model we build becomes context sensitive grammar, and problems induced seem to be NP hard. However for this particular model, we show that we can generalize the well-known inside-outside algorithm to estimate its parameters in cubic time. To alleviate the sparseness of n -gram counts, we also generalize various smoothing techniques to handle cases where there exist hidden variables. The experiments we have carried out show improvement in perplexity over current state-of-the-art technique. The composite language model trained in an unsupervised setting could be used as a parser, namely by selecting the most likely parse by the Viterbi algorithm, where the lexical information and semantic content should help to improve the performance.

Griffiths et al, (2004) recently proposed a generative composite HMM/LDA (latent Dirichlet allocation) model which takes into account of both short-range syntactic dependencies and long-range semantic dependencies between words and can be used to simultaneously find syntactic classes and semantic topics for purposes of part-of-speech tagging and document classification, but they have used MCMC to estimate the parameters for a much simpler model. However we propose an exact estimation algorithm for a much more complicated model.

We should note that there remain several avenues to improving the quality of the language models we are able to estimate from data. One way is to use semantic smoothing [3, 54], which has been shown to be effective in improving the perplexity results. Basically we can introduce an additional node between each topic node and word node to capture semantic similarity and subtle variation between words or introduce additional node S between the topic nodes and the document node to take into account of semantic similarity and sub-topic variation within each document and among documents.

Blei et al. [9] state that PLSA is not a well-defined generative model of documents, and there is no natural way to represent a document not seen in the original training corpus, this is why the “fold-in” heuristic procedure has to be used during testing to reestimate the semantic content. Blei et al. proposed LDA model to overcome this problem. It would be interesting to integrate LDA model into our composite language model and in this case variational method may have to be used.

Figure 8 illustrates the Chomsky’s hierarchy of grammars in terms of computational complexity [19, 23]. It seems that for a subclass of probabilistic context sensitive grammars, the problems there are indeed tractable. For example, instead of PCFG, we may combine the structural language model [12, 26] which is essentially a probabilistic push-down automata with trigram and PLSA models, the resulting composite model seems to be tractable. An interesting and outstanding question is then how to characterize this tractable subclass of probabilistic context sensitive grammars.

In language modeling research, there exists a body of ad hoc tricks which are known to make significant improvements. For example, Bellegarda [3] proposes an ad hoc combination of tri-gram and latent semantic analysis (LSA) models, by using the perplexity formula

$$p(w_\ell | w_1 \dots w_{\ell-1}) = \frac{p(w_\ell | w_{\ell-2} w_{\ell-1}) p_{PLSA}(d_\ell | w_\ell)}{\sum_{w_i} p(w_i | w_{\ell-2} w_{\ell-1}) p_{PLSA}(d_\ell | w_i)} = \frac{p(w_\ell | w_{\ell-2} w_{\ell-1}) \frac{p_{PLSA}(w_\ell | d_\ell)}{p_{PLSA}(w_\ell)}}{\sum_{w_i} p(w_i | w_{\ell-2} w_{\ell-1}) \frac{p_{PLSA}(w_i | d_\ell)}{p_{PLSA}(w_i)}} \quad (13)$$

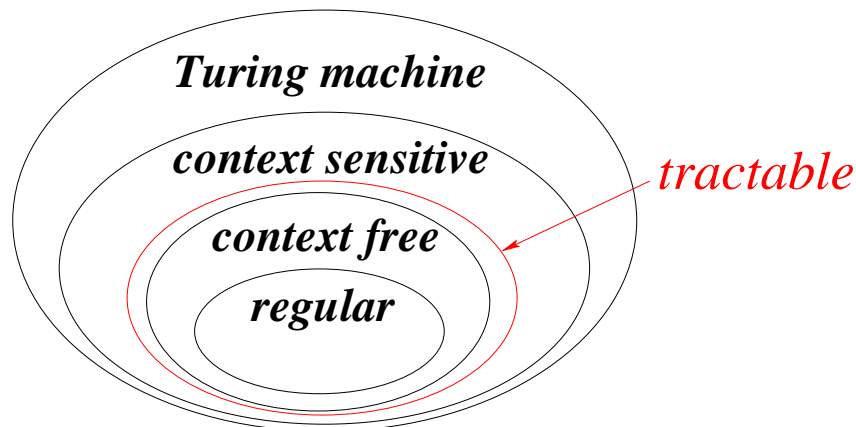


Figure 8: In the Chomsky's hierarchy of grammars nested according to the increasing restrictions placed on the production rules in the grammar, there is a subclass of probabilistic context sensitive grammars which is tractable.

Here $d_\ell = (w_1 \dots w_{\ell-1})$ is the current document d 's history; $p_{LSA}(d_\ell|w_\ell)$ is the probability of the current document history given the current word w_ℓ ; $p_{LSA}(w_\ell|d_\ell)$ is the probability of the current word w_ℓ history given the current document d_ℓ ; and $p_{LSA}(w_\ell)$ is the probability of current word w_ℓ . Each of these components is obtained by the latent semantic analysis. We calculated the perplexity of Bellegarda's model using the same training data and develop data considered above. The perplexity obtained by Bellegarda's model in this case is 99, which is a 9% reduction in perplexity compared to the baseline tri-gram model.

However, if one incorporates another ad hoc improvement from speech recognition research, a substantial further reduction can be obtained. The idea is to strengthen the influence of the LSA portion of the model by raising its contribution to some power (here 7) and renormalizing

$$p(w_\ell|w_1 \dots w_{\ell-1}) = \frac{p(w_\ell|w_{\ell-2}w_{\ell-1})(p_{LSA}(d_\ell|w_\ell))^7}{\sum_{w_i} p(w_i|w_{\ell-2}w_{\ell-1})(p_{LSA}(d_\ell|w_i))^7} \quad (14)$$

Doing so for Bellegarda's model results in a surprising and dramatic perplexity value of 86; almost equal to the best of our results using the composite syntactic/semantic/trigram model. Although we have yet to investigate such techniques in our composite language model, given the potentially dramatic improvements they potentially offer, it remains important future research to understand how such improvements might be incorporated in a principled manner.

References

- [1] S. Abney. 1997. Stochastic attribute-value grammars. *Computational Linguistics*, 23(4):597-618.
- [2] J. Baker. 1979. Trainable grammars for speech recognition. *Proceedings of the 97th Meeting of the Acoustical Society of America*, 547-550.
- [3] J. Bellegarda. 2000. Exploiting latent semantic information in statistical language modeling. *Proceedings of IEEE*, 88(8):1279-1296.
- [4] J. Bellegarda. 2001. Robustness in statistical language modeling: review and perspectives. In *Robustness in Languages and Speech Technology*, J. Junqua and G. van Noods (Editors), Kluwer Academic Publishers.
- [5] J. Bened and J. Snchez. 2005. Estimation of stochastic context-free grammars and their use as language models. *Computer Speech and Language*, 19(3): 249-274.
- [6] Y. Bengio, R. Ducharme, P. Vincent and C. Jauvin. 2003. A neural probabilistic language model. *Journal of Machine Learning Research*, 3:1137-1155.

- [7] A. Berger, S. Della Pietra and V. Della Pietra. 1996. A maximum entropy approach to natural language processing. *Computational Linguistics*, 22(1):39-71.
- [8] J. Bilmes. 2003. Graphical models and automatic speech recognition. In *Mathematical Foundations of Speech and Language Processing*, M. Johnson, S. Khudanpur, M. Ostendorf, R. Rosenfeld (Editors), Springer-Verlag.
- [9] D. Blei, A. Ng and M. Jordan. 2003. Latent Dirichlet allocation. *Journal of Machine Learning Research*, 3:993-1022.
- [10] T. Booth and R. Thompson. 1973. Applying probability measures to abstract languages. *IEEE Transactions on Computers*, 22:442-450.
- [11] P. Brown, S. Della Pietra, V. Della Pietra and R. Mercer. 1993. The mathematics of statistical machine translation: parameter estimation. *Computational Linguistics*, 19(2):263-311.
- [12] C. Chelba and F. Jelinek. 2000. Structured language modeling. *Computer Speech and Language*, 14(4):283-332.
- [13] S. Chen and J. Goodman. 1999. An empirical study of smoothing techniques for language modeling. *Computer Speech and Language*, 13(4): 319-358.
- [14] Z. Chi and S. Geman. 1998. Estimation of probabilistic context-free grammars. *Computational Linguistics*, 24(2):299-305.
- [15] Z. Chi. 1999. Statistical properties of probabilistic context-free grammars. *Computational Linguistics*, 25(1):131-160.
- [16] L. Coin, A. Bateman and R. Durbin. 2003. Enhanced protein domain discovery by using language modeling techniques from speech recognition. *Proceedings of the National Academy Sciences*, 100(8):4516-4520.
- [17] W. Croft and J. Lafferty. 2003. *Language Modeling for Information Retrieval*. Kluwer International Series on Information Retrieval, 13.
- [18] A. Dempster, N. Laird and D. Rubin. 1977. Maximum likelihood estimation from incomplete data via the EM algorithm. *Journal of Royal Statistical Society*, 39:1-38.
- [19] R. Durbin, S. Eddy, A. Krogh, and G. Mitchison. 1998. *Biological Sequence Analysis: Probabilistic Models of Proteins and Nucleic Acids*. Cambridge University Press.
- [20] I. Good. 1953. The population frequencies of species and the estimation of population parameters. *Biometrika*, 40:237-264.
- [21] T. Griffiths, M. Steyvers, D. Blei and J. Tenenbaum. 2004. Integrating topics and syntax. *Advances in Neural Information Processing Systems* 17.
- [22] T. Hofmann. 2001. Unsupervised learning by probabilistic latent semantic analysis. *Machine Learning*, 42(1):177-196.
- [23] J. Hopcroft and J. Ullman. 1979. *Introduction to Automata Theory, Languages and Computation*. Addison-Wesley.
- [24] F. Jelinek. 1998. *Statistical Methods for Speech Recognition*. MIT Press.
- [25] F. Jelinek and J. Lafferty. 1991. Computation of the probability of initial substring generation by stochastic context-free grammars. *Computational Linguistics*, 17(3):347-360.
- [26] F. Jelinek. 2004. Stochastic analysis of structured language modeling. *Mathematical Foundations of Speech and Language Processing*, M. Johnson, S. Khudanpur, M. Ostendorf and R. Rosenfeld (Editors), 37-72, Springer-Verlag.

- [27] M. Johnson. 1999. PCFG models of linguistic tree representations. *Computational Linguistics*, 24(4):613-632.
- [28] S. Katz. 1987. Estimation of probabilities from sparse data for the language model component of a speech recognizer. *IEEE Transactions on Acoustics, Speech and Signal Processing*, 35(3):400-401.
- [29] S. Khudanpur and J. Wu. 2000. Maximum entropy techniques for exploiting syntactic, semantic and collocational dependencies in language modeling. *Computer Speech and Language*, 14(4):355-372.
- [30] J. Lafferty. 2000. A derivation of the inside-outside algorithm from the EM algorithm. *IBM Research Report* 21636.
- [31] K. Lari and S. Young. 1990. The estimation of stochastic context-free grammars using the inside-outside algorithm. *Computer Speech and Language*, 4:35-56.
- [32] S. Lauritzen. 1996. *Graphical Models*. Oxford Press.
- [33] K. Mark, M. Miller and U. Grenander. 1996. Constrained stochastic language models, In *Image Models and Their Speech Model Cousins*, S. Levinson and L. Shepp (Editors), Springer-Verlag.
- [34] D. McAllester and R. Schapire. 2002. Learning theory and language modeling. In *Exploring Artificial Intelligence in the New Millennium*, G. Lakemeyer and B. Nebel, (Editors), Morgan Kaufmann.
- [35] D. McAllester, M. Collins and F. Pereira. 2004. Case-factor diagrams for structured probabilistic modeling. *Proceedings of 20th Conference on Uncertainty in Artificial Intelligence*.
- [36] M. Miller and J. O'Sullivan. 1992. Entropies and combinatorics of random branching processes and context-free languages. *IEEE Transactions on Information Theory*, 38(4):1292-1310.
- [37] Y. Miyao and J. Tsujii. 2002. Maximum Entropy Estimation for Feature Forests. *Proceedings of Human Language Technology Conference (HLT)*, 292-297.
- [38] H. Ney, U. Essen and R. Kneser. 1995. On the estimation of small probabilities by leaving-one-out. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(12):1202-1212, 1995.
- [39] H. Ney, S. Martin, F. Wessel. 1997. Statistical language modeling using leaving-one-out. In *Corpus Based Methods in Language and Speech Processing*, S. Young, G. Bloothoft (Editor.), pp. 174-207, Kluwer Academic Publishers.
- [40] F. Pereira. 2000. Formal grammar and information theory: together again? *Philosophical Transactions of the Royal Society: Mathematical, Physical and Engineering Sciences*, 358(1769):1239-1253, April.
- [41] J. Pearl. 1988. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann.
- [42] J. Pritchard, M. Stephens and P. Donnelly. 2000. Inference of population structure using multilocus genotype data. *Genetics*, 155:945-959.
- [43] B. Roark. 2001. Probabilistic top-down parsing and language modeling. *Computational Linguistics*, 27(2):249-276.
- [44] S. Roukos. 1995. Language representation. *Survey of the State of the Art in Human Language Technology*, R. Cole editor. 35-41, Cambridge University Press.
- [45] R. Rosenfeld. 1996. A maximum entropy approach to adaptive statistical language modeling. *Computer Speech and Language*, 10(2):187-228.
- [46] R. Rosenfeld. 2000. Incorporating linguistic structure into statistical language models. *Philosophical Transactions of the Royal Society: Mathematical, Physical and Engineering Sciences*, 358(1769):1311-1324.

- [47] L. Saul and F. Pereira. 1997. Aggregate and mixed-order Markov models for statistical language processing. *Proceedings of the Second Conference on Empirical Methods in Natural Language Processing*, 81-89. ACM Press.
- [48] D. Searls. 1992. The linguistics of DNA. *American Scientist*, 80(6):579-591.
- [49] C. Shannon. 1948. A mathematical theory of communication. *Bell System Technical Journal*, 27(2):379-423.
- [50] S. Shieber. 1985. Evidence against the context-freeness of natural language. *Linguistics and Philosophy*, 8:333-343.
- [51] D. Van Uytsel and D. Compernelle. 2005. Language modeling with probabilistic left corner parsing. *Computer Speech and Language*, 19(2):171-204.
- [52] M. Wainwright and M. Jordan. 2003. Graphical models, exponential families, and variational inference. *Technical Report 649, Department of Statistics, University of California, Berkeley*.
- [53] S. Wang, D. Schuurmans and Y. Zhao. 2003. The latent maximum entropy principle. *Manuscript*.
- [54] S. Wang, D. Schuurmans, F. Peng and Y. Zhao. 2005. Combining statistical language models via the latent maximum entropy principle. *Machine Learning Journal: Special Issue on Learning in Speech and Language Technologies*, 60:229-250.
- [55] S. Yedidia, W. Freeman and Y. Weiss. 2001. Generalized belief propagation. *Advances in Neural Information Processing Systems*, 13:689-695.
- [56] D. Younger. 1967. Recognition and parsing of context free languages in time N^3 . *Information and Control*, 10:198-208.

Proof: The proof of Theorem 1 is almost identical to the one given by Chi et al. [14, 15] which considers PCFG case. Let $q_A = \hat{p}$ (derivation tree rooted in A fails to terminate). We will show that $q_S = 0$ (i.e., derivation trees rooted in S always terminate). Consider Chomsky normal form, for each $A \in \Sigma$, let $n(A, t)$ be the count of instances of A in derivation tree t and $\hat{n}(A, t)$ be the count of non-root and non-terminal instances of A in derivation tree t . For any $A \in \Sigma$

$$\begin{aligned}
q_A &= \hat{p}\left(\bigcup_{B \in \Sigma} \bigcup_{B_1 B_2 \text{ s.t. } B \in \{B_1, B_2\}, (A \rightarrow B_1 B_2) \in \mathcal{R}} \bigcup_{B=B_1} \{B_1 \text{ fails to terminate}\} \cup_{B=B_2} \{B_2 \text{ fails to terminate}\}\right) \\
&\leq \sum_{B \in \Sigma} \hat{p}\left(\bigcup_{B_1 B_2 \text{ s.t. } B \in \{B_1, B_2\}, (A \rightarrow B_1 B_2) \in \mathcal{R}} \bigcup_{B=B_1} \{B_1 \text{ fails to terminate}\} \cup_{B=B_2} \{B_2 \text{ fails to terminate}\}\right) \\
&= \sum_{B \in \Sigma} \sum_{B_1 B_2 \text{ s.t. } B \in \{B_1, B_2\}, (A \rightarrow B_1 B_2) \in \mathcal{R}} \hat{p}\left(A \rightarrow B_1 B_2\right) \\
&\quad \hat{p}\left(\bigcup_{B=B_1} \{B_1 \text{ fails to terminate}\} \cup_{B=B_2} \{B_2 \text{ fails to terminate}\} | A \rightarrow B_1 B_2\right) \\
&\leq \sum_{B \in \Sigma} \sum_{B_1 B_2 \text{ s.t. } B \in \{B_1, B_2\}, (A \rightarrow B_1 B_2) \in \mathcal{R}} \hat{p}\left(A \rightarrow B_1 B_2\right) q_B \\
&= \sum_{B \in \Sigma} q_B \left(\frac{\sum_{B_1 B_2 \text{ s.t. } B \in \{B_1, B_2\}, (A \rightarrow B_1 B_2) \in \mathcal{R}} \sum_i E_{\hat{p}} \left[n\left(A \rightarrow B_1 B_2; t_i\right) | t_i \in \Omega_{W_i}, W_i \in \mathcal{D} \right]}{\sum_{B_1 B_2 \text{ s.t. } (A \rightarrow B_1 B_2) \in \mathcal{R}} \sum_i E_{\hat{p}} \left[n\left(A \rightarrow B_1 B_2; t_i\right) | t_i \in \Omega_{W_i}, W_i \in \mathcal{D} \right]} \right) \\
&= \sum_{B \in \Sigma} q_B \left(\frac{\sum_i \sum_{B_1 B_2 \text{ s.t. } B \in \{B_1, B_2\}, (A \rightarrow B_1 B_2) \in \mathcal{R}} E_{\hat{p}} \left[n\left(A \rightarrow B_1 B_2; t_i\right) | t_i \in \Omega_{W_i}, W_i \in \mathcal{D} \right]}{\sum_i \sum_{B_1 B_2 \text{ s.t. } (A \rightarrow B_1 B_2) \in \mathcal{R}} E_{\hat{p}} \left[n\left(A \rightarrow B_1 B_2; t_i\right) | t_i \in \Omega_{W_i}, W_i \in \mathcal{D} \right]} \right) \\
&= \sum_{B \in \Sigma} q_B \left(\frac{\sum_i \sum_{B_1 B_2 \text{ s.t. } B \in \{B_1, B_2\}, (A \rightarrow B_1 B_2) \in \mathcal{R}} E_{\hat{p}} \left[n\left(A \rightarrow B_1 B_2; t_i\right) | t_i \in \Omega_{W_i}, W_i \in \mathcal{D} \right]}{\sum_i E_{\hat{p}} \left[n\left(A; t_i\right) | t_i \in \Omega_{W_i}, W_i \in \mathcal{D} \right]} \right)
\end{aligned}$$

where $E_{\hat{p}}$ is expectation under \hat{p} and where “ \mathcal{D} ” means “conditioned on $t \in \Omega(W_l), W_l \in \mathcal{D}$ and $\Omega(W_l)$ is the set of parse trees for sentence W_l . Thus we have

$$\begin{aligned} & q_A \sum_i E_{\hat{p}} \left[n(A; t_i) \mid t_i \in \Omega_{W_l}, W_l \in \mathcal{D} \right] \\ \leq & \sum_{B \in \Sigma} q_B \sum_i \sum_{B_1 B_2 \text{ s.t. } B \in \{B_1, B_2\}, (A \rightarrow B_1 B_2) \in \mathcal{R}} E_{\hat{p}} \left[n(A \rightarrow B_1 B_2; t_i) \mid t_i \in \Omega_{W_l}, W_l \in \mathcal{D} \right] \end{aligned}$$

Sum over $A \in \Sigma$:

$$\begin{aligned} & \sum_{A \in \Sigma} q_A \sum_i E_{\hat{p}} \left[n(A; t_i) \mid t_i \in \Omega_{W_l}, W_l \in \mathcal{D} \right] \\ \leq & \sum_{B \in \Sigma} q_B \sum_i \sum_{A \in \Sigma} \sum_{B_1 B_2 \text{ s.t. } B \in \{B_1, B_2\}, (A \rightarrow B_1 B_2) \in \mathcal{R}} E_{\hat{p}} \left[n(A \rightarrow B_1 B_2; t_i) \mid t_i \in \Omega_{W_l}, W_l \in \mathcal{D} \right] \\ = & \sum_{B \in \Sigma} q_B \sum_i E_{\hat{p}} \left[n(B; t_i) \mid t_i \in \Omega_{W_l}, W_l \in \mathcal{D} \right] \end{aligned}$$

i.e.,

$$\sum_{A \in \Sigma} q_A \sum_i E_{\hat{p}} \left[\left(\hat{n}(A; t_i) - n(A; t_i) \right) \mid t_i \in \Omega_{W_l}, W_l \in \mathcal{D} \right] \geq 0$$

Clearly, for every i , $\hat{n}(A; t_i) = n(A; t_i)$ whenever $A \neq S$ and $\hat{n}(A; t_i) < n(A; t_i)$. Hence we conclude $q_S = 0$.

■