Computational Intelligence

An International Journal

Computational Intelligence, Volume 29, Number 4, 2013

EXPLOITING SYNTACTIC, SEMANTIC, AND LEXICAL REGULARITIES IN LANGUAGE MODELING VIA DIRECTED MARKOV RANDOM FIELDS

Shaojun Wang,[1] Shaomin Wang,[2] Li Cheng,[3] Russell Greiner,[4] and Dale Schuurmans[4]

[1]*Department of Computer Science and Engineering, Wright State University Dayton, Ohio, USA*

[2]*Visa Inc., San Francisco, California, USA*

[3]*Bioinformatics Institute, Singapore*

[4]*Department of Computing Science, University of Alberta Edmonton, Alberta, Canada*

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. Even though 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 Expectation-Maximization (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 algorithm 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.

Received 16 March 2011; Revised 6 February 2012; Accepted 10 March 2012; Published online 10 July 2012

Key words: language modeling, lexical information, syntactic structure, semantic content, directed Markov random field, generalized inside–outside algorithm.

1. INTRODUCTION

As a subproblem for machine translation and speech recognition under the source–channel paradigm (Jelinek 2009), statistical language modeling is essentially a density estimation problem to accurately compute 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 (Jelinek 1998); however, statistical language models have recently become more widely used in many other application areas such as information retrieval (Croft and Lafferty 2003), machine translation (Brown et al. 1993), optical character recognition, spelling correction, document classification, and bioinformatics (Searls 1992; Durbin et al. 1998; Coin, Bateman, and Durbin 2003).

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 (Shannon 1948). 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 (Chen and Goodman 1999). The resulting smoothed *n*-gram language models have become a key component of state-of-the-art speech

Address correspondence to Shaojun Wang, Department of Computer Science and Engineering, Wright State University, Dayton, OH 45435, USA; e-mail: shaojun.wang@wright.edu

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 (Bengio et al. 2003).

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 (Chelba and Jelinek 2000; Charniak 2001; Roark 2001) effectively exploits syntactic regularities to achieve greater accuracy than the n-gram model, and the semantic language model (Bellegarda 2000; Hofmann 2001) exploits document-level semantic regularities to achieve similar improvements. Unfortunately, each of these language models only targets some specific, distinct linguistic phenomena (Rosenfeld 2000). The key question (Roukos 1995; Pereira 2000; Bellegarda 2001; McAllester and Schapire 2002) 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 linear interpolation (Rosenfeld 1996; Chelba and Jelinek 2000), 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 (Rosenfeld 1996; Khudanpur and Wu 2000). This approach has since become a dominant technique in statistical natural language processing due to its 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. 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 (MRF). In fact, these two problems are exact duals of one another (Berger et al. 1996). The major weakness with ME methods, however, is that they can only model distributions over explicitly observed features, whereas in natural language, we encounter hidden semantic (Bellegarda 2000) and syntactic information (Chelba and Jelinek 2000). Wang et al. (2005a, 2012) proposed the latent ME (LME) principle, which extends standard ME estimation by incorporating hidden dependency structure. In previous work (Wang et al. 2005a), 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 Markov chain Monte Carlo (MCMC) sampling methods (Mark, Miller, and Grenander 1996; Abney 1997) would have to be employed, leading to enormous computational expense, see more explanation and analysis in Appendix A. Griffiths et al. (2004) proposed a generative composite hidden Markov model (HMM)/latent Dirichlet allocation (LDA) model that 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 (POS) tagging and document classification, not language modeling for density estimation of natural language, 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 for language modeling.

In this paper, instead of using an undirected MRF model, we present a unified generative *directed MRF model* framework that combines n-gram models, PCFG, and probabilistic latent semantic analysis (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 that 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 Section 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 (Baker 1979; Lari and Young 1990) 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 (1948) to capture local lexical regularities. The commonly used trigram model, or second-order Markov chain model, is constructed by assuming that 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 (Lari and Young 1990; Mark et al. 1996; Johnson 1999). However, a more complicated approach (Chelba and Jelinek 2000; Roark 2001) 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, 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

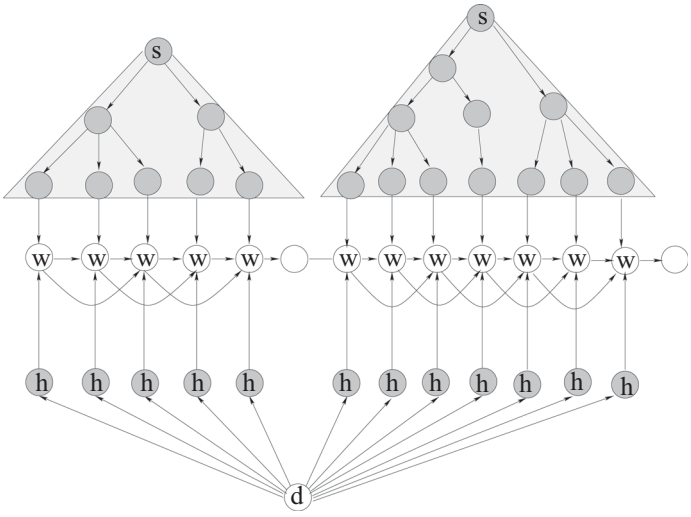

FIGURE 1. The observables in natural language consist of words, sentences, and documents; whereas the hidden data consist of sentence-level syntactic structure and document-level semantic content. The figure illustrates a composite chain/tree/table model incorporating these aspects, where light nodes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

between words and documents can then be easily defined in terms of their proximity in the semantic space by dimensionality reduction techniques (Bellegarda 2000; Hofmann 2001).

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 MRF to be a probability distribution $\mathcal{P}$ which admits a recursive factorization if there exist nonnegative 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

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

Here, $pa(\tau)$ denotes the set of parent states of τ. If the recursive factorization respects to a graph $\mathcal{G}$, then we have a Bayesian network (Lauritzen 1996). However, 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 PCFG to model the syntactic structure and a PLSA (Hofmann 2001) to model its semantic content of natural language. Each of these models can be represented as a directed MRF 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 four-tuple $(\Sigma, \mathcal{V}, \mathcal{R}, S)$ (Hopcroft and Ullman 1979) that consists of: a set of nonterminal symbols Σ whose elements are grammatical phrase markers; a vocabulary of $\mathcal{V} = \{v_1, \ldots, v_M\}$ 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: $A\to\gamma$, where $A\in\Sigma$ and $\gamma\in(\Sigma\cup\mathcal{V})^*$. A PCFG is a CFG with a probability assigned to each rule such that the probabilities of all rules

expanding a given nonterminal sum to 1; specifically, each right-hand side has a conditional probability given the left-hand side of the rule. A PCFG is a branching process (Miller and O'Sullivan 1992; Chi 1999) and its distribution can be represented in a form of (1); thus, it can be treated as a directed MRF model even though the straightforward representation as a complex directed graphical model is problematic (McAllester, Collins, and Pereira 2004), since given observed leaf nodes, there are potentially exponentially many distinct parse tree structures.

A PLSA (Hofmann 2001) is a generative probabilistic model of word-document co-occurrences using the bag-of-words assumption that is described as follows: (1) choose a document d_k with probability $\theta(d_k)$, (2) select a semantic class h with probability $\theta(d_k \rightarrow h)$, and (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, Stephens, and Donnelly (2000) 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 autoregressive HMM (Bilmes 2003), the combined syntactic trigram model is, in fact, an autoregressive PCFG; the combined semantic trigram model is, in fact, an autoregressive PLSA; see Figure 2. 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 autoregressive HMMs (Bilmes 2003).

An alternative approach is to combine these three models based on latent ME principle or under the undirected MRF paradigm (Wang et al. 2012), where the features consist of those discussed above. Appendix A illustrates the computational difficulties faced by this approach.

3. TRAINING ALGORITHM FOR THE COMPOSITE LANGUAGE MODEL

We are interested in learning a composite trigram/syntactic/semantic model from data. We assume that 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:

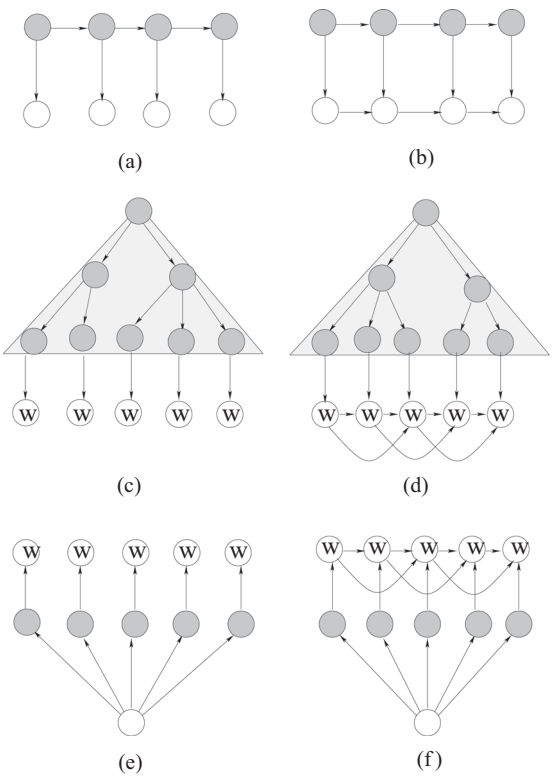

FIGURE 2. Comparing (a) HMM versus (b) autoregressive HMM, and (c) PCFG versus (d) autoregressive PCFG models, and (e) PLSA versus (f) autoregressive PLSA. Our composite trigram/syntactic/semantic model is, in fact, an autoregressive PCFG-PLSA.

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

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 nonterminal symbol A in sentence W_l of document d with parse tree t, and $n(A \to BC; d, W_l, t)$ is the count of nonterminal production rule $A \to BC$ in sentence W_l of document d with parse tree t. The parameters $\theta(d \to h)$, $\theta(uvAh \to w)$, $\theta(A \to BC)$ are normalized so that

$$\sum_{w\in\mathcal{V}}\theta(uvAh\to w) = 1,$$

$$\sum_{BC\in\Sigma}\theta(A\to BC)=1,$$
$$\sum_{h\in\mathcal{H}}\theta(d\to h)=1. \tag{3}$$

Thus, we have a constrained optimization problem, and there will be a Lagrange multiplier for $uvAh$, nonterminal 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 (Hopcroft and Ullman 1979) and potentially has exponential number of loops, which suggests that loopy belief propagation (Pearl 1988; Yedidia, Freeman, and Weiss 2001) and/or variational approximation methods (Wainwright and Jordan 2008) 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 (Hopcroft and Ullman 1979; Durbin et al. 1998); 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.

To make it easier to understand the generalized inside–outside algorithm for the composite trigram/syntactic/semantic model, we first briefly review the classical derivations for PCFGs shown in Lafferty (2000). Considering the PCFG model in Chomsky normal form, the likelihood of the observed data under the PCFG model can be written as below:

$$\mathcal{L}_{\mathrm{PCFG}}(\mathcal{W},\theta)=\prod_{d\in\mathcal{D}}\left(\prod_{l}\left(\sum_{t}\prod_{u,v\in\mathcal{V},A\to w\in\mathcal{R}}(\theta(A\to w))^{n(wA;d,W_l,t)}\prod_{A\to BC\in\mathcal{R}}\theta(A\to BC)^{n(A\to BC;d,W_l,t)}\right)\right), \tag{4}$$

where $n(wA;\, d,\, W_l,\, t)$ is the count of nonterminal symbol A generating w in sentence W_l of document d with parse tree t and $n(A\to BC;d,W_l,t)$ is the count of nonterminal production rule $A\to BC$ in sentence W_l of document d with parse tree t.

To apply the EM algorithm, we consider the auxiliary function

$$Q_{\mathrm{PCFG}}(\theta',\theta)=\sum_{d}\sum_{l}\sum_{t}p_\theta(t|d,W_l)\log\frac{p_{\theta'}(d,W_l,t)}{p_\theta(d,W_l,t)}, \tag{5}$$

where

$$p_\theta(d,W_l,t)=\prod_{A\to w\in\mathcal{R}}\theta(A\to w)^{n(A\to w;d,W_l,t)}\prod_{A\to BC\in\mathcal{R}}\theta(A\to BC)^{n(A\to BC;d,W_l,t)}. \tag{6}$$

Taking the derivative of $Q_{\mathrm{PCFG}}(\theta',\theta)$ with regard to $\theta'(A\to BC)$ and $\theta(A\to w)$, respectively, gives

$$\frac{\partial Q_{\mathrm{PCFG}}(\theta',\theta)}{\partial\theta'(A\to BC)}=\sum_{d\in\mathcal{D}}\sum_{l}\sum_{t}\frac{p_\theta(t|d,W_l)n(A\to BC;d,W_l,t)}{\theta'(A\to BC)},$$
$$\frac{\partial Q_{\mathrm{PCFG}}(\theta',\theta)}{\partial\theta'(A\to w)}=\sum_{d\in\mathcal{D}}\sum_{l}\sum_{t}\frac{p_\theta(t|d,W_l)n(A\to w;d,W_l,t)}{\theta'(A\to w)}.$$

Thus, the reestimated parameters of the PCFG model are then the normalized conditional expected counts:

$$\theta'(A \to BC) = \frac{\sum_{d\in\mathcal{D}}\sum_{l}\sum_{t} p_\theta(t|d, W_l)n(A \to BC; d, W_l, t)}{\sum_{B,C\in\Sigma}\sum_{d\in\mathcal{D}}\sum_{l}\sum_{t} p_\theta(t|d, W_l)n(A \to BC; d, W_l, t)},$$

$$\theta'(A \to w) = \frac{\sum_{d\in\mathcal{D}}\sum_{l}\sum_{t} p_\theta(t|d, W_l)n(A \to w; d, W_l, t)}{\sum_{w\in\mathcal{R}}\sum_{d\in\mathcal{D}}\sum_{l}\sum_{t} p_\theta(t|d, W_l)n(A \to w; d, W_l, t)}. \tag{7}$$

It is easy to check that the following equations hold:

$$\sum_{t} p_\theta(t|d, W_l)n(A \to BC; d, W_l, t) = \frac{\theta(A \to BC)}{p_\theta(d, W_l)}\frac{\partial p_\theta(d, W_l)}{\partial\theta(A \to BC)};$$

$$\sum_{t} p_\theta(t|d, W_l)n(A \to w; d, W_l, t) = \frac{\theta(A \to w)}{p_\theta(d, W_l)}\frac{\partial p_\theta(d, W_l)}{\partial\theta(A \to w)}.$$

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

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

From this, it is not difficult to see that

$$\begin{aligned} \frac{\partial p_\theta(S \to W_l \text{ in } d)}{\partial\theta(A \to BC)} = \sum_{i\le j\le k} &p_\theta(B \Rightarrow w_i \ldots w_j; W_l \text{ in } d)\ p_\theta(C \Rightarrow w_{j+1} \ldots w_k; W_l \text{ in } d) \\ &p_\theta(S \Rightarrow w_1 \ldots w_{i-1}Aw_{k+1} \ldots w_N; W_l \text{ in } d). \end{aligned}$$

Thus, the conditional expected number of times that the rule $A \to BC$ is used in generating the sentence $W_l \in \mathcal{W}$ in document d using the model θ is given by

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

where

$$\alpha_{ij}(B; W_l \text{ in } d) = p_\theta(B \Rightarrow w_i \ldots w_j; W_l \text{ in } d),$$

i.e., the inside probability that the nonterminal B derive the word subsequence $w_i \dots w_j$ in the sentence W_l of document d; and

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

i.e., the outside probability that beginning with the start symbol S, we can derive the sequence $w_1 \dots w_{i-1} A w_{k+1} \dots w_N$ in the sentence W_l of document d.

Similarly, we have

$$\sum_t p_\theta(t|d, W_l) n(A \to w; d, W_l, t) = \frac{\theta(A \to w)}{p_\theta(W_l \text{ in } d)} \sum_{1 \le i \le N} \delta_w(w_i) \theta(d \to h) \beta_{ii}(A; W_l \text{ in } d),$$

where δ is the indicator function.

Following Lafferty's (2000) derivation of the inside–outside formulas for updating the PCFG parameters from a general EM (Dempster, Laird, and Rubin 1977) algorithm, we now formally 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 $\mathcal{L}(\mathcal{W}, \theta') \ge \mathcal{L}(\mathcal{W}, \theta)$. 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 CFG needs to be generalized to take into account the trigram model and PLSA model.

To apply the EM algorithm, we consider the auxiliary function

$$Q(\theta', \theta) = \sum_d \sum_l \sum_{H_l} \sum_t p_\theta(H_l, t|d, W_l) \log \frac{p_{\theta'}(d, W_l, H_l, t)}{p_\theta(d, W_l, H_l, t)}, \tag{8}$$

where

$$p_\theta(d, W_l, H_l, t) = \prod_{h \in \mathcal{H}} \theta(d \to h)^{n(d, W_l, h)} \prod_{u,v \in \mathcal{V}, A \to w \in \mathcal{R}, h \in \mathcal{H}} \theta(uvAh \to w)^{n(uvAh \to w; d, W_l, t, h)} \prod_{A \to BC \in \mathcal{R}} \theta(A \to BC)^{n(A \to BC; d, W_l, t)}, \tag{9}$$

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

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

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

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

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

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

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

Because of the normalization constraints (3), the reestimated parameters of the composite model are then the normalized conditional expected counts (10):

$$\begin{aligned}
\theta'(A \to BC) &= \frac{\sum_{d \in \mathcal{D}} \sum_{l} \sum_{H_l} \sum_{t} p_\theta(H_l, t|d, W_l) n(A \to BC; d, W_l, t)}{\sum_{B,C \in \Sigma} \sum_{d \in \mathcal{D}} \sum_{l} \sum_{H_l} \sum_{t} p_\theta(H_l, t|d, W_l) n(A \to BC; d, W_l, t)}, \\
\theta'(uvAh \to w) &= \frac{\sum_{d \in \mathcal{D}} \sum_{l} \sum_{H_l} \sum_{t} p_\theta(H_l, t|d, W_l) n(uvAh \to w; d, W_l, t, h)}{\sum_{w \in \mathcal{R}} \sum_{d \in \mathcal{D}} \sum_{l} \sum_{H_l} \sum_{t} p_\theta(H_l, t|d, W_l) n(uvAh \to w; d, W_l, t, h)}, \\
\theta'(d \to h) &= \frac{\sum_{l} \sum_{H_l} \sum_{t} p_\theta(H_l, t|d, W_l) n(d \to h; d, W_l, h)}{\sum_{h \in \mathcal{H}} \sum_{l} \sum_{H_l} \sum_{t} p_\theta(H_l, t|d, W_l) n(d \to h; d, W_l, h)}.
\end{aligned} \tag{10}$$

This looks the same as the PCFG model (Lafferty 2000).

Thus, we need to compute the conditional expected counts:

$$\sum_{d \in \mathcal{D}} \sum_{l} \sum_{H_l} \sum_{t} p_\theta(H_l, t|d, W_l) n(A \to BC; d, W_l, t)$$

$$\sum_{d \in \mathcal{D}} \sum_{l} \sum_{H_l} \sum_{t} p_\theta(H_l, t|d, W_l) n(uvAh \to w; d, W_l, t, h)$$

$$\sum_{l} \sum_{H_l} \sum_{t} p_\theta(H_l, t|d, W_l) n(d \to h; d, W_l, h).$$

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:

$$\sum_{H_l} \sum_{t} p_\theta(H_l, t|d, W_l) n(A \to BC; d, W_l, t) = \frac{\theta(A \to BC)}{p_\theta(d, W_l)} \frac{\partial p_\theta(d, W_l)}{\partial \theta(A \to BC)};$$

$$\sum_{H_l} \sum_{t} p_\theta(H_l, t|d, W_l) n(uvAh \to w; d, W_l, t, h) = \frac{\theta(uvAh \to w)}{p_\theta(d, W_l)} \frac{\partial p_\theta(d, W_l)}{\partial \theta(uvAh \to w)};$$

$$\sum_{H_l} \sum_{t} p_\theta(H_l, t|d, W_l) n(d \to h; d, W_l, h) = \frac{\theta(d \to h)}{p_\theta(d, W_l)} \frac{\partial p_\theta(d, W_l)}{\partial \theta(d \to h)};$$

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

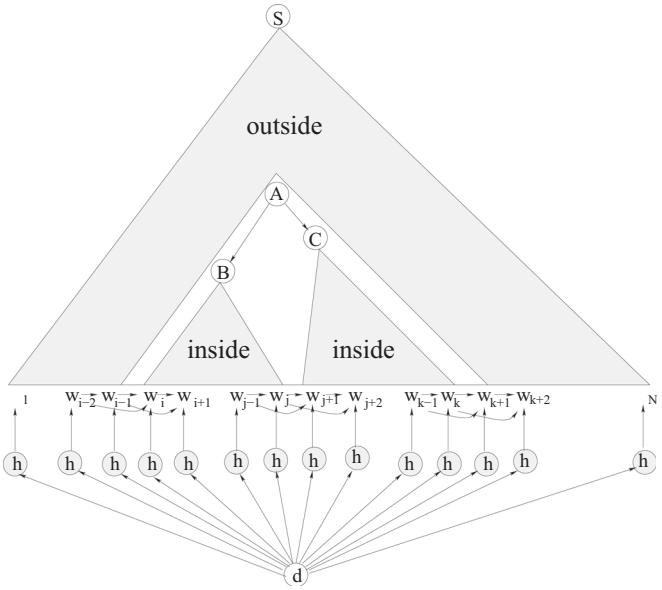

FIGURE 3. Inside and outside probabilities for nonterminal production rule in the composite language model, where each component is influenced by the injected trigram and PLSA models.

Let $A \Rightarrow \gamma$ denote that, beginning with a nonterminal A, we can derive a string γ of words and nonterminals 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 $A \to BC$ within a tree t for sentence $W_l = (w_1, \ldots, w_N)$ in document d can be specified by a triple (i, j, k), $i \le j \le k$. The partial derivative of the probability $p_\theta(S \to W_l \text{ in } d) = p_\theta(d, W_l)$ with respect to the parameter $\theta(A \to BC)$ only involves those parse trees which use the rule $A \to BC$. Consider the event "$S \to W_l$ in d using $A \to BC$ 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:

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

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 that 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 nonternimal 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.

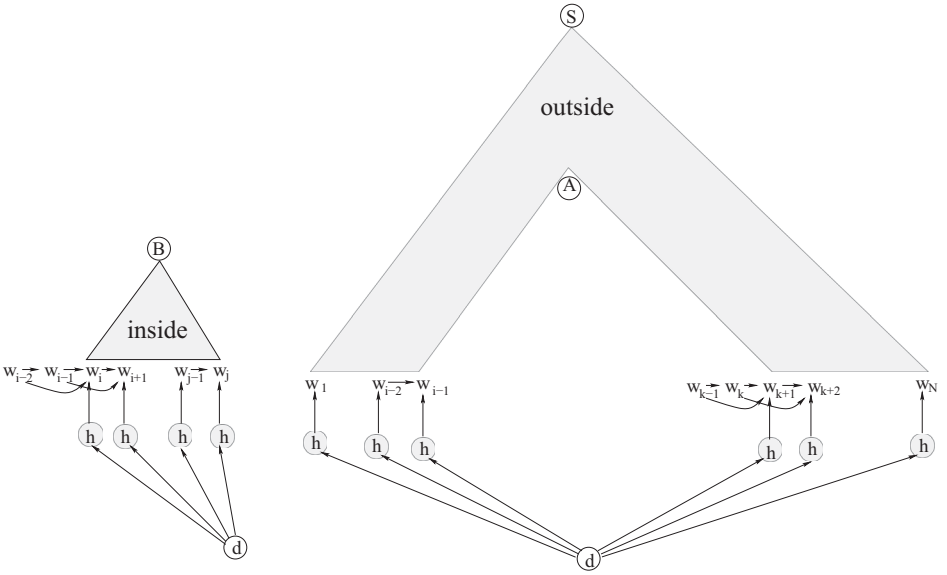

FIGURE 4. Inside probability α_{ij} and outside probability β_{ik} in the composite language model.

From this, it is not difficult to see that

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

Thus, the conditional expected number of times that the rule $A \to BC$ is used in generating the sentence $W_l \in \mathcal{W}$ in document d using the model θ is given by

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

where

$$\alpha_{ij}(B; W_l \text{ in } d) = p_\theta(B \Rightarrow w_i \dots w_j; W_l \text{ in } d),$$

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

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

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 $w_1 \dots w_{i-1} A w_{k+1} \dots w_N$ in the sentence W_l of document d. See Figure 4 for illustration.

Similarly, consider the event "$S \to W_l$ using $uvAh \to w$ in d in position (i)." Because of the Markov property of the directed MRF model, the probability of this event can be

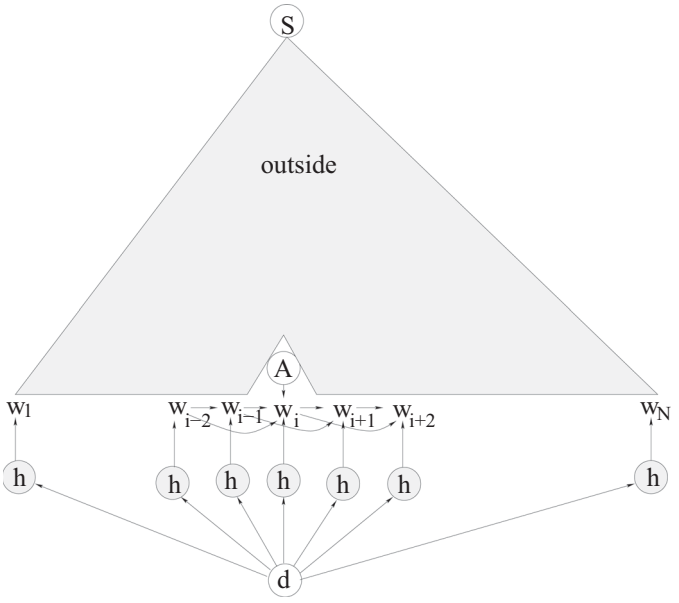

FIGURE 5. Inside and outside probabilities for terminal production rule in the composite language model, where each component is influenced by the injected trigram and PLSA models.

written as a product of four terms, again *the factorization property*, as follows:

$$\begin{aligned} &p_\theta(S \to W_l \text{ in } d;\ \text{using } uvAh \to w \text{ in position } (i)) \\ &\quad = \delta_{uvw}(w_{i-2}w_{i-1}w_i)(\theta(d \to h)\theta(uvAh \to w)) \\ &\quad p_\theta(S \Rightarrow w_1 \dots w_{i-1}Aw_{i+1} \dots w_N; W_l \text{ in } d). \end{aligned}$$

See Figure 5for 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 $uvAh \to w$. Again, the factorization property is the crucial constituent for the success to derive an efficient and exact recursive algorithm.

Thus, we have

$$\begin{aligned} &\sum_{H_l} \sum_{t} p_\theta(H_l, t|d, W_l) n(uvAh \to w; d, W_l, t) \\ &\quad = \frac{\theta(uvAh \to w)}{p_\theta(W_l \text{ in } d)} \sum_{1 \le i \le N} \delta_{uvw}(w_{i-2}w_{i-1}w_i)\theta(d \to h)\beta_{ii}(A; W_l \text{ in } d), \end{aligned}$$

where δ is the indicator function.

Similarly, consider the event "$d \to W_l$ in d using $d \to h$ in position (i)." Because of the Markov property of the directed MRF model, the probability of this event can be written as

a product of three terms as follows:

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

Thus, we have

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

Just as in the PCFG case, there is an efficient recursive method for computing the αs and βs using the CKY chart-parsing algorithm (Younger 1967). The method for doing this is almost the same as for PCFG and is implicit in the following recursive formulas:

$$\alpha_{ij}(A; W_l \text{ in } d) = \sum_{BC} \sum_{i \le k \le j} (\theta(A \to BC) \alpha_{ik}(A; W_l \text{ in } d) \alpha_{k+1 j}(C; W_l \text{ in } d)), \tag{11}$$

$$\alpha_{ii}(A; W_l \text{ in } d) = \sum_{h} (\theta(d \to h) \theta(w_{i-2} w_{i-1} Ah \to w_i)), \tag{12}$$

$$\begin{aligned} \beta_{ij}(A; W_l \text{ in } d) = \Bigg(& \sum_{B,C} \sum_{k<i} \theta(B \to CA) \alpha_{ki-1}(C; W_l \text{ in } d) \beta_{kj}(B; W_l \text{ in } d) \\ & + \sum_{B,C} \sum_{k>j} \theta(B \to AC) \alpha_{j+1k}(C; W_l \text{ in } d) \beta_{ik}(B; W_l \text{ in } d) \Bigg), \end{aligned} \tag{13}$$

$$\beta_{1N}(A; W_l \text{ in } d) = \delta_S(A; W_l \text{ in } d). \tag{14}$$

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 remaining αs and βs via the above recursive formulas. The time complexity of the algorithm is cubic in terms of the length of a sentence, as is apparent from the recursive loops over three sequence position indices, i, j, k.

Chi and Geman (1998) and Chi (1999) proved that the maximum likelihood estimate of production rule probabilities for a PCFG yields a *proper distribution*, i.e., there is no probability mass lost to infinitely large trees (Booth and Thompson 1973). Similarly, we can show that the maximum likelihood estimate of production rule probabilities for this composite trigram/syntactic/semantic model always yields 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 B. ■

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 $\theta(A \to BC)\alpha_{ik}(B)\alpha_{k+1,j}(C)$. When we reach the topmost nonterminal S, we can then "trace back" 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 (Chen and Goodman 1999) 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 $\theta(uvAh \to w)$. 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 (Good 1953) 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

$$n^* = (n+1)\frac{r_{n+1}}{r_n}, \tag{15}$$

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 vuw with n counts, we take

$$P_{GT}(uvw) = \frac{n^*}{N}, \tag{16}$$

where $N = \sum_{n=0}^{\infty} r_n n^*$. In practice, the Good–Turing estimate is not used by itself (Katz 1987); 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 (1987) 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 are given as

$$\rho_n = \frac{\frac{n^*}{n} - \frac{(k+1)r_{k+1}}{r_1}}{1 - \frac{(k+1)r_{k+1}}{r_1}}. \tag{17}$$

When we use (10) to estimate $\theta(uvAh \to w)$, we use the expected count of $n(uvAh \to w)$, where A and h are hidden. However, when the trigram uvw has count $n(uv \to w) > 0$, if we discount the expected count of $n(uvAh \to w)$ by the ratio $\rho_n(uvw)$, then we discount the trigrams by the same ratio $\rho_n(uvw)$ since $\sum_{A\in\Sigma, h\in\mathcal{H}} \rho_n(uvw)n(uvAh \to w) = \rho_n(uvw)n(uv \to w)$. Therefore, instead of using iterative parameter estimation of (10), we

use smoothed iterative parameter estimation as equation (18),

$$\theta'_s(uvAh \to w) = \frac{\rho_n(uvw)\sum_{d\in\mathcal{D}}\sum_{l}\sum_{H_l}\sum_{t} p_\theta(H_l, t|d, W_l)n(uvAh \to w; d, W_l, t, h)}{\sum_{w\in\mathcal{R}}\rho_n(uvw)\sum_{d\in\mathcal{D}}\sum_{l}\sum_{H_l}\sum_{t} p_\theta(H_l, t|d, W_l)n(uvAh \to w; d, W_l, t, h)}. \quad (18)$$

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

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

and

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

Similarly, we can use Kneser–Ney smoothing (Ney, Essen, and Kneser 1995; Ney, Martin, and Wessel 1997), and the smoothed iterative parameter estimation as equation (19),

$$\theta'_s(uvAh \to w) = \frac{n_{1+}(\cdot vw)\sum_{d\in\mathcal{D}}\sum_{l}\sum_{H_l}\sum_{t} p_\theta(H_l, t|d, W_l)n(uvAh \to w; d, W_l, t, h)}{\sum_{w\in\mathcal{R}} n_{1+}(\cdot vw)\sum_{d\in\mathcal{D}}\sum_{l}\sum_{H_l}\sum_{t} p_\theta(H_l, t|d, W_l)n(uvAh \to w; d, W_l, t, h)}, \quad (19)$$

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 $p_\theta(S \to w_1 w_2 \ldots w_k \ldots)$; that is, the probability that an arbitrary word sequence $w_1 w_2 \ldots w_k$ is the initial subsequence of a sentence generated by the composite trigram, syntactic, and semantic language model. We derive the *generalized left-to-right inside* algorithm to perform this computation by following the work of (Jelinek and Lafferty 1991), which assumes that a PCFG model is used.

In the following, we basically use the same notation as in Jelinek and Lafferty (1991). 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 $w_i \ldots w_j$. Thus,

$$\begin{aligned} p_\theta(A \ll i, j) = \alpha_{ij}(A) &+ \sum_{x_1 \in \mathcal{V}} p_\theta(A \to w_i \ldots w_j x_1) \\ &+ \sum_{x_1 x_2 \in \mathcal{V}^2} p_\theta(A \to w_i \ldots w_j x_1 x_2) + \cdots \\ &+ \sum_{x_1 \ldots x_n \in \mathcal{V}^n} p_\theta(A \to w_i \ldots w_j x_1 \ldots x_n) + \cdots \end{aligned} \tag{20}$$

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

Let $p_\theta^L(A \to B) = \sum_{B_2 \in \Sigma} p_\theta(A \to B_1 B_2)$ be the sum of the probabilities of all the rules $A \to B_1 B_2$ whose first left-hand 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

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

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

$$p_\theta^L(A \Rightarrow B_1 B_2) = p_\theta(A \to B_1 B_2) + \sum_{C \in \Sigma} p_\theta^L(A \Rightarrow C) p_\theta(C \to B_1 B_2). \tag{21}$$

Obviously,

$$\begin{aligned} p_\theta(A \ll i, i{+}n) = \sum_{B_1, B_2 \in \Sigma} & p_\theta(A \to B_1 B_2) \, (\alpha_{i,i}(B_1) p_\theta(B_2 \ll i{+}1, i{+}n) \\ & + \alpha_{i,i+1}(B_1) p_\theta(B_2 \ll i{+}2, i{+}n) + \cdots \\ & + \alpha_{i,i+n-1}(B_1 \Rightarrow w_i \ldots w_{i+n-1}) p_\theta(B_2 \ll i{+}n, i{+}n) \\ & + p_\theta(B_1 \ll i, i{+}n)), \end{aligned}$$

since to generate the initial subsequence $w_i w_{i+1} \ldots w_{i+n}$, some rule $A \to B_1 B_2$ 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

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

Then, following the same recursive derivation as in Jelinek and Lafferty (1991), we have

$$\begin{aligned} p_\theta(A \ll i, i+n) &= \sum_{B_1, B_2 \in \Sigma} p_\theta(A \to B_1 B_2) R(B_1, B_2) \\ &\quad + \sum_{B_1 \in \Sigma} p_\theta^L(A \to B_1) p(B_1 \ll i, i+n) \\ &= \sum_{B_1, B_2 \in \Sigma} \left[p_\theta(A \to B_1 B_2) + \sum_{C_1 \in \Sigma} p_\theta^L(A \to C_1) p_\theta(C_1 \to B_1 B_2) \right] R(B_1, B_2) \\ &\quad + \sum_{C_1, C_2 \in \Sigma} p_\theta^L(A \to C_1) p_\theta^L(C_1 \to C_2) p(C_2 \ll i, i+n) \\ &= \cdots\cdots \\ &= \sum_{B_1, B_2 \in \Sigma} p_\theta^L(A \Rightarrow B_1 B_2) R(B_1, B_2) + \sum_{C_1, \ldots, C_k \in \Sigma, k \to \infty} \left(p_\theta^L(A \to C_1) \right. \\ &\quad \left. \prod_{l=2}^{k} p_\theta^L(C_{l-1} \to C_l) p(C_k \ll i, i+n) \right). \end{aligned} \tag{22}$$

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 (21) and successive resubstitutions, we get the final formula

$$\begin{aligned} p_\theta(A \ll i, i+n) &= \sum_{B, B_2 \in \Sigma} p_\theta^L(A \Rightarrow B_1 B_2) R(B_1, B_2) \\ &= \sum_{B, B_2 \in \Sigma} p_\theta^L(A \Rightarrow B_1 B_2) \left(\sum_{j=1}^{n} \alpha_{i, i+j-1}(B_1) p_\theta(B_2 \ll i+j, i+n) \right). \end{aligned} \tag{23}$$

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. Similar to the observation for PCFG model in Jelinek and Lafferty (1991, p. 320), this algorithm is very similar to that of inside probability (11), and thus the time complexity is cubic order of n.

The desired probability $p(S \to w_1, \ldots, w_n, \ldots)$ can thus be calculated exactly in the same recursive way as in the PCFG case, which is described in Jelinek and Lafferty (1991).

5. EXPERIMENTAL EVALUATION

5.1. Experimental Data Sets and Performance Measure

The corpus used to train our model was taken from the *Wall Street Journal* (WSJ) portion of the North American business (NAB) corpus, which was composed of about 150,000 documents spanning the years 1987–1989, comprising approximately 42 million words. The vocabulary was constructed by taking the 20,000 most frequent words of the

TABLE 1. Data Sets Statistics.

	No. of articles	No. of sentences	No. of words
Train	150,981	1,611,571	41,780,924
Dev	378	6,904	157,312
Test	379	6,638	153,801

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.

The statistics of the data sets are shown in Table 1.

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 $\mathcal{T} = \{d_1, \ldots, d_L\}$ and a language model defining $p(s)$, we calculate test perplexity and test entropy as

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

where $|\mathcal{T}| = \sum_{i=1}^{L} |d_i|$ 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 Hofmann (2001). The parameters corresponding to the document-semantic arcs, $\theta(d \to h)$, are reestimated by the probability of word subsequence currently seen, $w_1, \ldots, w_k$, i.e., the initial subsequence of a sentence generated by the composite language model, while holding the other parameters fixed. The reason we use prefix at test time is because when the language model is used in speech recognition or machine translation (particularly phrase-based), the word string in a sentence is transcribed or translated sequentially one-by-one. When we decode the next word w_{k+1}, assume that we treat the history as known. Given a new test sentence W_l, the recursive gradient update for $\theta(d \to h)$ shown below is done first, and then using the recursive formula in Section 4, we calculate the likelihood of W_l given the model. Gildea and Hofmann (1999) used an online EM algorithm to reestimate this parameter. In fact, Bellegarda (2000) encountered a similar situation when he used, LSA uncovers the salient semantic relationships between words. He had to treat the document history to be the current document so far, then map it to semantic feature space through singular value decomposition (SVD) and compute LSA probability. This is a procedure analogous to what we are doing here.

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

$$\theta(d \to h)^{(k)} = \theta(d \to h)^{(k-1)} + \epsilon \frac{\partial \log p_\theta(S \ll 1, k)}{\partial \theta(d \to h)}\bigg|_{\theta(d\to h)^{(k-1)}},$$

where ϵ is the learning rate and we find that empirically setting $\epsilon = 0.2$ gives the best perplexity reduction.

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

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

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

By the formula (23), we have

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

Again, this algorithm is very similar to that of inside probability, and thus the time complexity is cubic order of n.

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

$$\begin{aligned}
&\frac{\partial \alpha_{i,i+j-1}(B_1)}{\partial \theta(d \to h)} \\
&= \sum_{C_1C_2} \sum_{i\le k\le i+j-1} \theta(B_1 \to C_1 C_2) \left(\frac{\partial \alpha_{i,k}(C_1)}{\partial \theta(d \to h)} \alpha_{k+1,i+j-1}(C_2) + \alpha_{i,k}(C_1) \frac{\partial \alpha_{k+1,i+j-1}(C_2)}{\partial \theta(d \to h)} \right)
\end{aligned}$$

and

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

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

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

TABLE 2. Perplexity Results for the Composite Syntactic Semantic Trigram Model on Development Corpus.

Language model	Perplexity Good–Turing	Perplexity Kneser–Ney
Trigram (Baseline)	109	103
4-gram	105	99
5-gram	106	101
PCFG	678	
Linear interpolation of PCFG & trigram	109	102
PLSA	1,487	
Linear interpolation of PLSA & trigram	109	103
Linear interpolation of PLSA, PCFG, & trigram	108	102
Syntactic and semantic	598	596
Syntactic trigram	94	90
Semantic trigram	96	91
Syntactic, semantic trigram	82	79

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 obtain 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 at sentence level, 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 1,487. When this PLSA model is combined with Good–Turing back-off and Kneser–Ney smoothing trigram models by linear interpolation at sentence level, we find that the test perplexity scores remain unchanged. If we combine these three models together using linear interpolation at sentence level, we obtain the perplexity scores on development data 108 and 102, respectively.

Next, we introduce the composite syntactic/trigram model that 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 that is equivalent to the composite syntactic/semantic/trigram language model by setting the syntatic 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 Hofmann (2001): the document-semantic parameters are reestimated by MLE while holding semantic word parameters fixed, where the empirical distribution is given by the current updated document history.

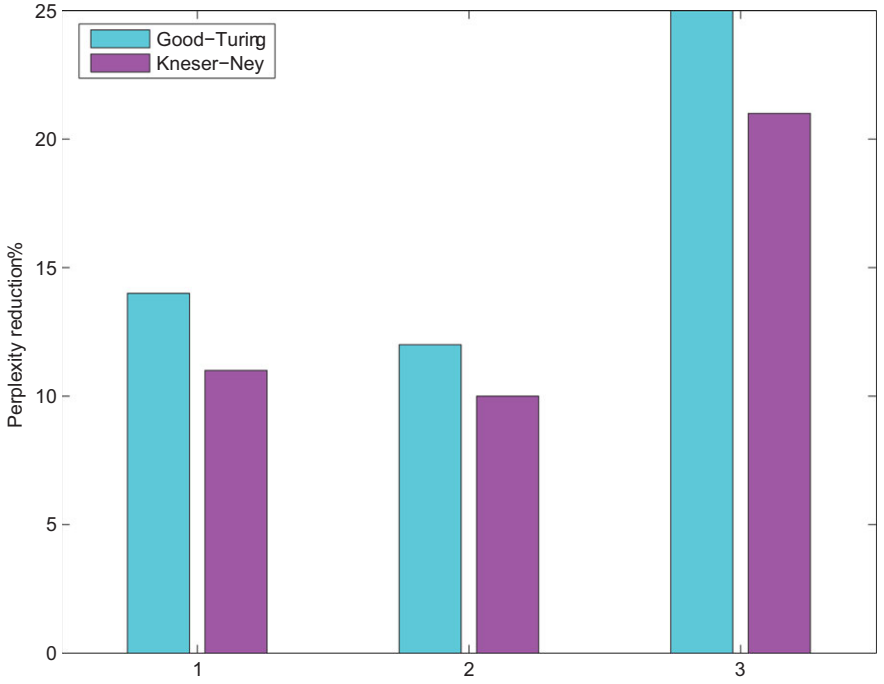

FIGURE 6. Relative perplexity reductions over baseline trigram with Good–Turing and Kneser–Ney smoothings by various composite language models: 1) Syntactic-trigram, 2) Semantic-trigram, and 3) Syntactic-semantic trigram.

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 which 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 reestimated by recursive gradient ascent of MLE 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.

When we perform the generalized inside–outside algorithm to train the composite trigram/PCFG/PLSA model, five iterations are sufficient to convergence, and it takes approximately 10 hours for each iteration. During testing, we use recursive gradient ascent to update the document-semantic parameters and the generalized left-to-right inside algorithm to compute the perplexity, it takes approximately 25 hours.

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.

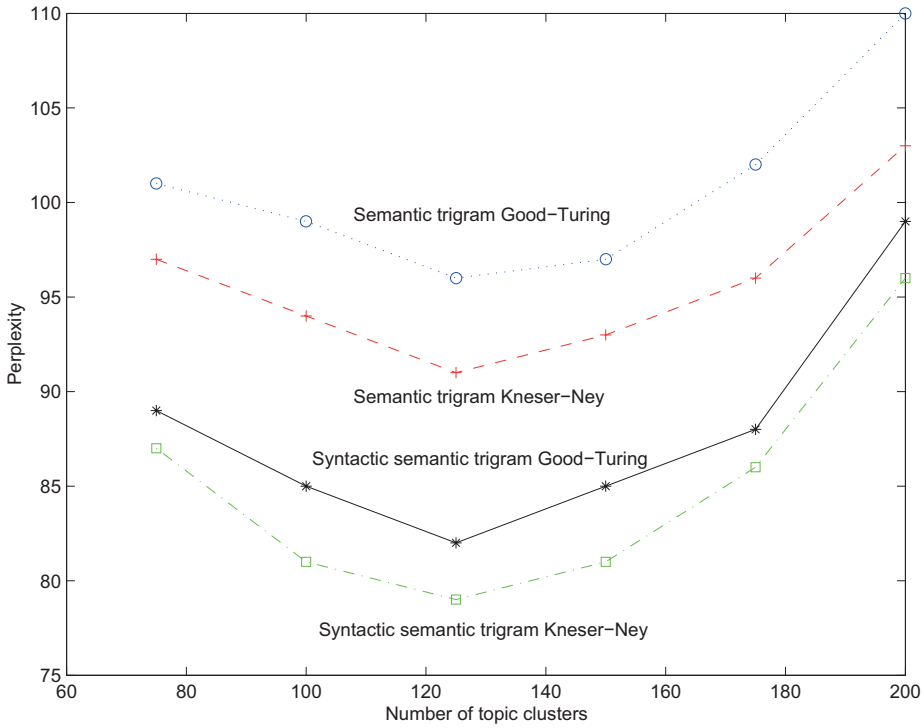

FIGURE 7. Perplexities versus number of topics for the semantic/trigram and the syntactic/semantic/trigram language models.

It is well known that HMM is a special case of PCFG (Durbin et al. 1998). So, we take POS tags from the Penn Treebank as the hidden states and train a composite trigram/HMM language model. However, we find that the composite trigram/HMM language model gives worse perplexity result than trigram. In fact, this experiment has been done by IBM researcher in early 1990s (Jelinek 2009).

It has been observed that the number of semantic node is critical to the final perplexity results (Bellegarda 2000; Wang et al. 2005), so next, we study how the number of semantic nodes influences our composite language model. We first fix the number of hidden semantic topics, and 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,

TABLE 3. Perplexity Results for the Syntactic/Trigram, Semantic/Trigram as Well as Syntactic/Semantic/Trigram Language Models on Test Corpus.

Language model	Perplexity Good–Turing	Reduction	Perplexity Kneser–Ney	Reduction
Trigram (Baseline)	103		99	
Syntactic trigram	91	12%	89	10%
Semantic trigram	92	10%	90	9%
Syntactic, semantic trigram	80	23%	78	21%

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.

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 trigrams with current state-of-the-art smoothing techniques. The composite language model trained in an unsupervised setting could be used as a parser, namely, by selecting the most likely parse by the Viberbi algorithm, where the lexical information and semantic content should help to improve the performance.

Griffiths et al. (2004) proposed a generative composite HMM/LDA model that 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 POS 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 (Bellegarda 2000; Wang et al. 2005), 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 subtopic variation within each document and among documents.

Blei, Ng, and Jordan (2003) 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.

Integrating LDA with bigrams has been investigated by Wallach (2006, 2008), where the author used MCMC in the inference step and very small training corpora up to 100K tokens. However, the problem becomes much complicated, harder, and challenging when adding PCFG into n-gram/LDA since the number of parse trees grows exponentially fast with sentence length. If we use the same variational approach as in Blei et al. (2003) for LDA to approximate the log-likelihood of the observed data and choose variational distribution to be a product of a Dirichlet and multinomials, what we found is that in the variational lower bound, there is a term that is an expected conditional marginal likelihood over *hidden* variables, the parse trees in PCFG model. This is completely different from previous work using variational methods in Jordan et al. (1999). Unfortunately, these hidden variables make the computation of this term intractable. How to handle this term becomes a new challenge of using a variational method for inference and parameter estimation in complex probabilistic models. An alternative approach to overcome this difficulty is to reconstruct a tractable composite trigram/PCFG/PLSA model, and let the multinomial variational distribution play the role of $\theta(d \rightarrow h)$ in the composite trigram/PCFG/PLSA language model. This enables us to expand the intractable term, i.e., the expected conditional marginal likelihood over *hidden* variables, into two terms, and the objective we are optimizing will become a kind of penalized log-likelihood consisting of the log-likelihood of the reconstructed composite trigram/PCFG/PLSA model, and a regularization term that measures the divergence between the variational distribution and the true distribution of the semantic information. It can be further shown that maximizing this penalized log-likelihood with respect to parameters of variational distributions is equivalent to minimizing the difference of two Kullback–Leibler (KL) divergences; one is the KL divergence between the semantic probability and the posterior semantic probability given the sentences, and the other is between the variational posterior probability and the true posterior probability.

In this work, we have used a simple vanilla PCFG, further research is to use very rich PCFG models used in (Charniak 2001; Roark 2001; Petrov et al. 2006) where the nonterminals are annotated with lexical and nonlexical contextual information.

We have used a rather ad hoc approach to extend smoothing methods to handle hidden variables in the composite trigram/PCFG/PLSA language model. How to smooth fractional counts due to latent variables in Kneser–Ney's sense in a principled way is a long-standing open problem. Teh (2006) described a hierarchical Bayesian model consisting of Pitman–Yor processes as a language model and derived estimation formulas for trigrams based on this model, which are generalizations of one of the most successful smoothing techniques, interpolated Kneser–Ney smoothing (Ney et al. 1995). It can be shown that nonparametric Bayesian smoothing for the composite trigram/PCFG/PLSA language model is a hierarchical extension of the Pitman–Yor process that obeys power law distribution, the hierarchy of nonparametric priors is a lattice. By exploring the particular structures of the hierarchical Pitman–Yor mixture composite syntactic, semantic, and lexical language model and resorting to MCMC sampling method and variational methods such as structured mean-field variational EM, etc., the seemingly complicated estimation and inference problems can be decomposed into easier subproblems, where the generalized inside–outside algorithms developed in this paper can be carried out as an internal building block. Since the model obeys power law distribution, a long-standing open problem, smoothing fractional counts due to latent variables in Kneser–Ney's sense in a principled way, might be solved.

REFERENCES

ABNEY, S. 1997. Stochastic attribute-value grammars. Computational Linguistics, **23**(4):597–618.

BAKER, J. 1979. Trainable grammars for speech recognition. *In* Proceedings of the 97th Meeting of the Acoustical Society of America, pp. 547–550.

BELLEGARDA, J. 2000. Exploiting latent semantic information in statistical language modeling. Proceedings of IEEE, **88**(8):1279–1296.

BELLEGARDA, J. 2001. Robustness in statistical language modeling: Review and perspectives. *In* Robustness in Languages and Space Technology. *Edited by* J. Junqua and G. van Noods. Kluwer Academic Publishers: Norwell, MA.

BENGIO, Y., R. DUCHARME, P. VINCENT, and C. JAUVIN. 2003. A neural probabilistic language model. Journal of Machine Learning Research, **3**:1137–1155.

BERGER, A., S. D. PIETRA, and V. D. PIETRA. 1996. A maximum entropy approach to natural language processing. Computational Linguistics, **22**(1):39–71.

BILMES, J. 2003. Graphical models and automatic speech recognition. *In* Mathematical Foundations of Speech and Language Processing. *Edited by* M. Johnson, S. Khudanpur, M. Ostendorf, and R. Rosenfeld. Springer-Verlag: New York.

BLEI, D., A. NG, and M. JORDAN. 2003. Latent Dirichlet allocation. Journal of Machine Learning Research, **3**:993–1022.

BOOTH, T., and R. THOMPSON. 1973. Applying probability measures to abstract languages. IEEE Transactions on Computers, **22**:442–450.

BROWN, P., S. D. PIETRA, V. D. PIETRA, and R. MERCER. 1993. The mathematics of statistical machine translation: Parameter estimation. Computational Linguistics, **19**(2):263–311.

CHARNIAK, E. 2001. Immediate-head parsing for language models. *In* Proceedings of the 39th Annual Conference on Association of Computational Linguistics, pp. 124–131.

CHELBA, C., and F. JELINEK. 2000. Structured language modeling. Computer Speech and Language, **14**(4):283–332.

CHEN, S., and J. GOODMAN. 1999. An empirical study of smoothing techniques for language modeling. Computer Speech and Language, **13**(4):319–358.

CHI, Z., and S. GEMAN. 1998. Estimation of probabilistic context-free grammars. Computational Linguistics, **24**(2):299–305.

CHI, Z. 1999. Statistical properties of probabilistic context-free grammars. Computational Linguistics, **25**(1):131–160.

COIN, L., A. BATEMAN, and R. DURBIN. 2003. Enhanced protein domain discovery by using language modeling techniques from speech recognition. *In* Proceedings of the National Academy Sciences, **100**(8):4516–4520.

CROFT, W., and J. LAFFERTY. 2003. Language Modeling for Information Retrieval. Kluwer International Series on Information Retrieval, **13**: Norwell, MA.

DEMPSTER, A., N. LAIRD, and D. RUBIN. 1977. Maximum likelihood estimation from incomplete data via the EM algorithm. Journal of Royal Statistical Society, **39**:1–38.

DURBIN, R., S. EDDY, A. KROGH, and G. MITCHISON. 1998. Biological Sequence Analysis: Probabilistic Models of Proteins and Nucleic Acids. Cambridge University Press: Cambridge, UK.

GILDEA, D., and T. HOFMANN. 1999. Topic-based language models using EM. *In* Proceedings of the 6th European Conference on Speech Communication and Technology, pp. 2167–2170.

GOOD, I. 1953. The population frequencies of species and the estimation of population parameters. Biometrika, **40**:237–264.

GRIFFITHS, T., M. STEYVERS, D. BLEI, and J. TENENBAUM. 2004. Integrating topics and syntax. *In* Advances in Neural Information Processing Systems, **17**. MIT Press: Cambridge, MA.

HOFMANN, T. 2001. Unsupervised learning by probabilistic latent semantic analysis. Machine Learning, **42**(1):177–196.

HOPCROFT, J., and J. ULLMAN. 1979. Introduction to Automata Theory, Languages and Computation. Addison-Wesley: Boston, MA.

JELINEK, F. 1998. Statistical Methods for Speech Recognition. MIT Press: Cambridge, MA.

JELINEK, F. 2009. The dawn of statistical ASR and MT. Computational Linguistics, **35**(4):483–494.

JELINEK, F., and J. LAFFERTY. 1991. Computation of the probability of initial substring generation by stochastic context-free grammars. Computational Linguistics, **17**(3):347–360.

JOHNSON, M. 1999. PCFG models of linguistic tree representations. Computational Linguistics, **24**(4):613–632.

KATZ, S. 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.

KHUDANPUR, S., 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.

LAFFERTY, J. 2000. A derivation of the inside outside algorithm from the EM algorithm. IBM Research Report 21636.

LARI, K., and S. YOUNG. 1990. The estimation of stochastic context-free grammars using the inside outside algorithm. Computer Speech and Language, **4**:35–56.

LAURITZEN, S. 1996. Graphical Models. Oxford University Press: Oxford, UK.

MARK, K., M. MILLER, and U. GRENANDER. 1996. Constrained stochastic language models. *In* Image Models and Their Speech Model Cousins. *Edited by* S. Levinson and L. Shepp. Springer-Verlag: New York.

MCALLESTER, D., M. COLLINS, and F. PEREIRA. 2004. Case-factor diagrams for structured probabilistic modeling. *In* Proceedings of the 20th Conference on Uncertainty in Artificial Intelligence, pp. 382–391.

MCALLESTER, D., and R. SCHAPIRE. 2002. Learning theory and language modeling. *In* Exploring Artificial Intelligence in the New Millenium. *Edited by* G. Lakemeyer and B. Nebel. Morgan Kaufmann: Waltham, MA.

MILLER, M., 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.

NEY, H., 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.

NEY, H., S. MARTIN, and F. WESSEL. 1997. Statistical language modeling using leaving-one-out. *In* Corpus Based Methods in Language and Speech Processing. *Edited by* S. Young and G. Bloothoft. Kluwer Academic Publishers: Norwell, MA, pp. 174–207.

PEARL, J. 1988. Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference. Morgan Kaufmann: Waltham, MA.

PEREIRA, F. 2000. Formal grammar and information theory: Together again? Philosophical Transactions of the Royal Society: Mathematical, Physical and Engineering Sciences, **358**(1769): 1239–1253.

PETROV, S., L. BARRETT, R. THIBAUX, and D. KLEIN. 2006. Learning accurate, compact, and interpretable tree annotation. *In* Proceedings of the 21st International Conference on Computational Linguistics and the 44th Annual Meeting of the Association for Computational Linguistics (COLING-/ACL), pp. 443–440.

PRITCHARD, J., M. STEPHENS, and P. DONNELLY. 2000. Inference of population structure using multilocus genotype data. Genetics, **155**:945–959.

ROARK, B. 2001. Probabilistic top-down parsing and language modeling. Computational Linguistics, **27**(2):249–276.

ROUKOS, S. 1995. Language representation. *In* Survey of the State of the Art in Human Language Technology. *Edited by* R. Cole. Cambridge University Press: New York, pp. 35–41.

ROSENFELD, R. 1996. A maximum entropy approach to adaptive statistical language modeling. Computer Speech and Language, **10**(2):187–228.

ROSENFELD, R. 2000. Incorporating linguistic structure into statistical language models. Philosophical Transactions of the Royal Society: Mathematical, Physical and Engineering Sciences, **358**(1769): 1311–1324.

ROSENFELD, R., S. CHEN, and X. ZHU. 2001. Whole-sentence exponential language models: A vehicle for linguistic-statistical integration. Computer Speech and Language, **15**(1):55–73.

SEARLS, D. 1992. The linguistics of DNA. American Scientist, **80**(6):579–591.

SHANNON, C. 1948. A mathematical theory of communication. Bell System Technical Journal, **27**(2): 379–423.

TEH, Y. 2006. A hierarchical Bayesian language model based on Pitman-Yor processes. *In* Proceedings of the 44th Annual Conference on Association of Computational Linguistics, pp. 985–992.

WAINWRIGHT, M., and M. JORDAN. 2008. Graphical models, exponential families, and variational inference. Foundations and Trends in Machine Learning, **1**:1–305.

WALLACH, H. 2006. Topic modeling: Beyond bag-of-words. *In* Proceedings of the 23rd International Conference on Machine Learning (ICML), pp. 977–984.

WALLACH, H. 2008. Structured topic models for language. Ph.D. Dissertation, University of Cambridge: Cambridge, UK.

WANG, S., D. SCHUURMANS, F. PENG, and Y. ZHAO. 2005a. Combining statistical language models via the latent maximum entropy principle. Machine Learning Journal: Special Issue on Learning in Speech and Language Technologies, **59**:1–22.

WANG, S., D. SCHUURMANS, and Y. ZHAO. 2012. The latent maximum entropy principle. ACM Transactions on Knowledge Discovery from Data (TKDD), forthcoming.

WANG, S., S. WANG, R. GREINER, D. SCHUURMANS, and L. CHENG. 2005b. Exploiting syntactic, semantic and lexical regularities in language modeling via directed Markov random fields. *In* Proceedings of the 22nd International Conference on Machine Learning (ICML), pp. 953–960.

YEDIDIA, S., W. FREEMAN, and Y. WEISS. 2001. Generalized belief propagation. Advances in Neural Information Processing Systems, **13**:689–695.

YOUNGER, D. 1967. Recognition and parsing of context free languages in time N^3. Information and Control, **10**:198–208.

APPENDIX A

Combine n-gram, m-SLM, and PLSA by latent ME principle or under undirected MRF paradigm.

ME approach has been a key method for language modeling since 1990s (Rosenfeld 1996; Jelinek 1998). In this section, we illustrate how to combine n-gram, PCFG, and PLSA using ME philosophy and what computational difficulties there are. Since the parse tree structure and semantic string of a sentence are not observable, we apply the latent ME principle (Wang et al. 2012). Let W, H, t, and d denote a sentence, its parse tree structure, and semantic string of a document, respectively. We choose the features as $f(uvAhw)$, $f(ABC)$, and $f(gd)$ in the composite language model formed under the directed MRF paradigm. We denote the set of features as vectors $\underline{f}(uvAhw)$, $\underline{f}(ABC)$, and $\underline{f}(gd)$, respectively. We look at the count of each feature over W, H, t, and d, and denote the counts of the whole set of features over W, H, t, and d as a vector $\#\underline{f}(d, W, H, t)$. Under the latent ME principle (Wang et al. 2003), we choose $P(d, W, H, t)$ which maximizes the joint entropy

$$\max_{p(d,W,H,t)} -\sum_{d,W,H,t} p(d, W, H, t) \log p(d, W, H, t) \tag{A.1}$$

subject to the following nonlinear constraints

$$\begin{aligned}
\sum_{d,W,H,t} p(d, W, H, t)\underline{f}(uvAhw) &= \sum_{d,W} \tilde{p}(d, W) \sum_{H,t} p(H, t|d, W)\underline{f}(uvAhw),\\
\sum_{d,W,H,t} p(d, W, H, t)\underline{f}(ABC) &= \sum_{d,W} \tilde{p}(d, W) \sum_{H,t} p(H, t|d, W)\underline{f}(ABC),\\
\sum_{d,W,H,t} P(d, W, H, t)\underline{f}(gd) &= \sum_{W,D} \tilde{p}(d, W) \sum_{H,t} P(H, t|d, W)\underline{f}(gd),
\end{aligned} \tag{A.2}$$

where $\tilde{P}(W, D)$ denotes the empirical distribution of a sentence in a document over training corpus. This is a nonconvex optimization problem due to the nonlinear constraints and there is no closed-form solution.

Following what we have proposed in Wang et al. (2012), we restrict our solution to have an exponential form as below, and satisfy the same constraints of (A.2)

$$p_{\underline{\lambda}}(d, W, H, t) = \frac{1}{Z_{\underline{\lambda}}} e^{<\underline{\lambda}, \# \underline{f}(d, W, H, t)>}, \tag{A.3}$$

where $Z_{\underline{\lambda}} = \sum_{d,W,H,t} e^{<\underline{\lambda}, \# \underline{f}(d, W, H, t)>}$ is the normalization factor to ensure a proper distribution. This is reminiscent of the whole sentence model in Rosenfeld, Chen, and Zhu (2001) if we choose $\underline{f}(uvAhw)$, $\underline{f}(ABC)$, and $\underline{f}(gd)$ as the features for the whole sentence model. Thus, $p_{\underline{\lambda}}(d, W, H, t)$ is a composite language model that is formed by integrating n-gram, PCFG, and PLSA under *undirected* MRF paradigm. If we compare the sentence likelihood of composite language under *undirected* MRF paradigm, $p_{\underline{\lambda}}(d, W, H, t)$ in (A.3), with that under *directed* MRF paradigm, $p_\theta(d, W, H, t)$ in (9), the key difference between (A.3) and (9) is on normalization where $p_{\underline{\lambda}}(d, W, H, t)$ has only one global normalization factor and $p_\theta(d, W, H, t)$ has many local normalization factors.

To estimate unknown parameters $\underline{\lambda}$, we maximize the following log-likelihood

$$\max_{\underline{\lambda}} \sum_{d,W} \tilde{P}(d, W) \sum_{H,t} \log p_{\underline{\lambda}}(d, W, H, t). \tag{A.4}$$

Taking derivative with respect to $\underline{\lambda}$ and setting to 0, we obtain the same set of constraints shown in (A.2). Thus, the feasible solutions of ME are the stationary points of maximum likelihood, the difference between ME and maximum likelihood is that, among a set of feasible solutions (stationary points), ME chooses the one having the highest entropy and maximum likelihood chooses the one having the highest likelihood (Wang et al. 2012). Unfortunately, the training is intractable due to the following reasons: (i) The normalization factor is intractable since the number of all possible sentences with fixed length L is $|\mathcal{V}|^L$ and L can be arbitrary. (ii) The feature expectation on the right-hand side of equations (A.2) is tractable where we can use generalized inside–outside algorithm analogous to the one in Section 3 or N-best list approximation algorithm proposed in this paper; however, the feature expectation on the left-hand side of equations (A.2) is intractable since the number of all possible sentences with fixed length L is $|\mathcal{V}|^L$ and L can be arbitrary.

APPENDIX B

The proof of Theorem 1 is almost identical to the one given by Chi and Geman (1998) and Chi (1999), 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 Chormsky normal form, for each $A \in \Sigma$, let $n(A, t)$ be the count of instances of A in derivation tree t and $1\,\hat{n}(A, t)$ be the count of nonroot and nonterminal instances of A in derivation tree t. For any $A \in \Sigma$

$$
\begin{aligned}
q_A &= \hat{p}(\cup_{B\in\Sigma} \cup_{B_1B_2 \text{ s.t. } B\in\{B_1,B_2\},(A\to B_1B_2)\in\mathcal{R}} \cup_{B=B_1} \{B_1 \text{ fails to terminate}\} \\
&\quad \cup_{B=B_2}\{B_2 \text{ fails to terminate}\} \\
&\le \sum_{B\in\Sigma} \hat{p}(\cup_{B_1B_2 \text{ s.t. } B\in\{B_1,B_2\},(A\to B_1B_2)\in\mathcal{R}} \cup_{B=B_1}\{B_1 \text{ fails to terminate}\} \\
&\quad \cup_{B=B_2}\{B_2 \text{ fails to terminate}\}) \\
&= \sum_{B\in\Sigma} \sum_{B_1B_2 \text{ s.t. } B\in\{B_1,B_2\},(A\to B_1B_2)\in\mathcal{R}} \hat{p}(A \to B_1B_2) \\
&\quad \hat{p}(\cup_{B=B_1}\{B_1 \text{ fails to terminate}\} \cup_{B=B_2} \{B_2 \text{ fails to terminate}\} | A \to B_1B_2) \\
&\le \sum_{B\in\Sigma} \sum_{B_1B_2 \text{ s.t. } B\in\{B_1,B_2\},(A\to B_1B_2)\in\mathcal{R}} \hat{p}(A \to B_1B_2) q_B \\
&= \sum_{B\in\Sigma} q_B \left(\frac{\sum_{B_1B_2 \text{ s.t. } B\in\{B_1,B_2\},(A\to B_1B_2)\in\mathcal{R}} \sum_i E_{\hat{p}}[n(A \to B_1B_2; t_i) | t_i \in \Omega_{W_l}, W_l \in \mathcal{D}]}{\sum_{B_1B_2 \text{ s.t. } (A\to B_1B_2)\in\mathcal{R}} \sum_i E_{\hat{p}}[n(A \to B_1B_2; t_i) | t_i \in \Omega_{W_l}, W_l \in \mathcal{D}]} \right) \\
&= \sum_{B\in\Sigma} q_B \left(\frac{\sum_i \sum_{B_1B_2 \text{ s.t. } B\in\{B_1,B_2\},(A\to B_1B_2)\in\mathcal{R}} E_{\hat{p}}[n(A \to B_1B_2; t_i) | t_i \in \Omega_{W_l}, W_l \in \mathcal{D}]}{\sum_i \sum_{B_1B_2 \text{ s.t. } (A\to B_1B_2)\in\mathcal{R}} E_{\hat{p}}[n(A \to B_1B_2; t_i) | t_i \in \Omega_{W_l}, W_l \in \mathcal{D}]} \right) \\
&= \sum_{B\in\Sigma} q_B \left(\frac{\sum_i \sum_{B_1B_2 \text{ s.t. } B\in\{B_1,B_2\},(A\to B_1B_2)\in\mathcal{R}} E_{\hat{p}}[n(A \to B_1B_2; t_i) | t_i \in \Omega_{W_l}, W_l \in \mathcal{D}]}{\sum_i E_{\hat{p}}[n(A; t_i) | t_i \in \Omega_{W_l}, W_l \in \mathcal{D}]} \right),
\end{aligned}
$$

where $E_{\hat{p}}$ is expectation under $\hat{p}$ and "$|\mathcal{D}$" 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

$$
\begin{aligned}
& q_A \sum_i E_{\hat{p}}[n(A; t_i) | t_i \in \Omega_{W_l}, W_l \in \mathcal{D}] \\
& \quad \le \sum_{B\in\Sigma} q_B \sum_i \sum_{B_1B_2 \text{ s.t. } B\in\{B_1,B_2\},(A\to B_1B_2)\in\mathcal{R}} E_{\hat{p}}[n(A \to B_1B_2; t_i) | t_i \in \Omega_{W_l}, W_l \in \mathcal{D}].
\end{aligned}
$$

Sum over $A \in \Sigma$:

$$\sum_{A\in\Sigma} q_A \sum_i E_{\hat{p}}[n(A;t_i)|t_i \in \Omega_{W_l}, W_l \in \mathcal{D}]$$

$$\leq \sum_{B\in\Sigma} q_B \sum_i \sum_{A\in\Sigma} \sum_{B_1 B_2 \text{ s.t. } B\in\{B_1,B_2\},(A\to B_1B_2)\in\mathcal{R}} E_{\hat{p}}[n(A \to B_1B_2;t_i)|t_i \in \Omega_{W_l}, W_l \in \mathcal{D}]$$

$$= \sum_{B\in\Sigma} q_B \sum_i E_{\hat{p}}[n(B;t_i)|t_i \in \Omega_{W_l}, W_l \in \mathcal{D}],$$

i.e.,

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

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 $q_S = 0$.