

# Stochastic Analysis of Lexical and Semantic Enhanced Structural Language Model

Shaojun Wang<sup>1,2</sup>, Shaomin Wang<sup>3</sup>, Li Cheng<sup>4</sup>,  
Russell Greiner<sup>2</sup>, and Dale Schuurmans<sup>2</sup>

<sup>1</sup> Wright State University, USA

<sup>2</sup> University of Alberta, Canada

<sup>3</sup> Oracle, USA

<sup>4</sup> National ICT, Australia

**Abstract.** In this paper, we present a directed Markov random field model that integrates trigram models, structural language models (SLM) and probabilistic latent semantic analysis (PLSA) for the purpose of statistical language modeling. The SLM is essentially a generalization of shift-reduce probabilistic push-down automata thus more complex and powerful than probabilistic context free grammars (PCFGs). The added context-sensitiveness due to trigrams and PLSAs and violation of tree structure in the topology of the underlying random field model make the inference and parameter estimation problems plausibly intractable, however the analysis of the behavior of the lexical and semantic enhanced structural language model leads to a generalized inside-outside algorithm and thus to rigorous exact EM type re-estimation of the composite language model parameters.

**Keywords:** Language modeling, structural language model, trigram, probabilistic latent semantic analysis.

## 1 Introduction

Natural language perhaps is one of the most intriguing and complex stochastic processes (Chomsky 1956, Jelinek 1998). It was first studied by Shannon (1948) as a Markov chain model when he introduced information theory to illustrate many of its features. Since then various kinds of generative probabilistic language models have been proposed to capture different aspects of natural language regularity. 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, machine translation, optical character recognition, spelling correction, document classification, and bioinformatics. The most recent advance is the work by Wang et al. (2005), where they have proposed a first generative probabilistic model of natural language, a *directed Markov random field model*, that combines trigram models, PCFGs and PLSAs (Hofmann 2001) and simultaneously exploits the relevant lexical, syntactic and

semantic information of natural language with tractable parameter estimation algorithm.

Jelinek and Chelba (Chelba and Jelinek 2000, Jelinek 2004) have developed a structural language model that exploits syntactic structure incrementally while traversing the sentence from left to right, and used it to extract meaningful information from the word history, thus enabling the use of sentence level long range dependencies. SLM is essentially a generalization of shift-reduce probabilistic push-down automata and is non-context free (Jelinek 2004). A thorough comparative study between this model with PCFGs has been presented in (Abney et al. 1999). The probabilistic dependency structure in SLM is more complex than that in a PCFG. When SLM was originally introduced (Chelba and Jelinek 2000), it operated with the help of a set of stacks containing partial parses from which less probable sub-parse trees are discarded. The parameters were trained by a procedure based on n-best final parses. It has been shown that the use of SLM results in lower perplexities as well as lower error rates in speech recognition. When SLM is combined with trigram model with linear interpolation (Chelba and Jelinek 2000) or integrated with trigram model and semantic language model, which carry complementary dependency structure, under maximum entropy estimation paradigm (Khudanpur and Wu 2000) with SLM as a preprocessing tool to extract syntactic structure, almost additive results have been observed in perplexity or word error reductions. Later, Jelinek (2004) studied various stochastic properties of the SLM, in particular he generalized the CKY algorithm (Younger 1967) to obtain a chart which is able to directly compute the sentence probability thus making the stack unnecessary, moreover he derived a generalized inside-outside algorithm which leads to a rigorous EM type re-estimation for the SLM parameters.

Inspired by the works by Jelinek (2004) and Wang et al. (2005), we study the stochastic properties of a composite generative probabilistic language model which integrates trigram model, PLSA models with SLM. Similar as for PCFG (Jelinek et al. 1992) and SLM (Jelinek 2004), among the stochastic properties with which we study are the following ones:

- The probability of the generated sentence based on a generalization of the CKY algorithm.
- The probability of the next word given the sentence prefix.
- The probability of the most probable parse.
- Training algorithm for the statistical parameters of the composite language model.

The added context-sensitiveness due to trigrams and PLSAs and violation of tree structure in the topology of the underlying random field model make the inference and parameter estimation plausibly intractable, in the following we show that exact recursive algorithms do exist with the same order polynomial time complexity as in the SLM for the study of stochastic properties of the lexical and semantic enhanced SLM.

## 2 Jelinek and Chelba’s Simplified Structural Language Model

In this section, we briefly describe the simplified structural language model (SSLM) which was introduced by Jelinek and Chelba (Chelba 1999, Chelba and Jelinek 2000, Jelinek and Chelba 1999, Jelinek 2004). SSLM is completely lexical, that is, phrases are annotated by headwords but not by non-terminals. As the operation of the SSLM proceeds, it generates a string of words,  $w_0, w_1, \dots, w_n, w_{n+1}$ , where  $\forall i = 1, \dots, n, w_i \in \mathcal{V}, w_0 = \langle s \rangle$ , and  $w_{n+1} = \langle /s \rangle$ , where  $\mathcal{V}$  is the set of vocabulary,  $\langle s \rangle, \langle /s \rangle$  are start and stop markers of the sentence, at the meantime, it also generates a parse consisting of a binary tree whose nodes are marked by *headwords* of phrases spanned by the subtree stemming from the leaf. The headword of a phrase can be any word belonging to the span of the phrase and the headword at the apex of the final tree is  $\langle s \rangle$ . The SSLM operates from left to right, builds up the phrase structure in a bottom-up manner and it has two type of operations, *constructor* moves and *predictor* moves.

**1. Constructor moves:** The constructor looks at the pair of right most exposed headwords,  $h_{-2}, h_{-1}$  and takes an action  $a \in \mathcal{A} = \{\text{adjoin right}, \text{adjoin left}, \text{null}\}$  with probability  $\theta(a|h_{-2}h_{-1})$ . The operations of these three actions are defined as the following:

- **adjoin right:** create an apex marked by the identity of  $h_{-1}$  and connect it by a leftward branch to its leftmost exposed headword  $h_{-2}$  and by a rightward branch to the exposed headword  $h_{-1}$ . Increase the indices of the current exposed headwords  $h_{-3}, h_{-4}, \dots$  by 1. These headwords together with  $h_{-1}$  become become the new set of exposed headwords.
- **adjoin left:** create an apex marked by the identity of  $h_{-2}$  and connect it by a leftward branch to its leftmost exposed headword  $h_{-2}$  and by a rightward branch to the exposed headword  $h_{-1}$ . Increase the indices of the new apex and those of the current exposed headwords  $h_{-3}, h_{-4}, \dots$  by 1. These headwords become become the new set of exposed headwords.
- **null:** leave headword indexing and current parse structure unchanged and pass control to the predictor.

If  $a \in \{\text{adjoin left}\}, \text{adjoin right}\}$ , the constructor stays in control and chooses the next action with probability  $\theta(a|h_{-2}h_{-1})$ . If  $a = \text{null}$ , the constructor stops and the control is passed to the predictor. A **null** move ensures that the right-most exposed headword will eventually be connected to the right and an **adjoin** move makes the right-most exposed headword being connected to the left.

**2. Predictor moves:** The predictor generates the next word  $w_i$  with probability  $\theta(w_i|h_{-2}h_{-1}), w_i \in \mathcal{V} \cup \langle s \rangle$ . The indexes of the current headwords  $h_{-1}, h_{-2}, \dots$  are decreased by 1 and the newly generated word becomes the right most exposed headword. Control is then passed to the constructor.

As in (Wang et al. 2005), 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  $\Gamma$  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  $p(x) = \prod_{\tau \in \Gamma} k^\tau(x_\tau, x_{pa(\tau)})$

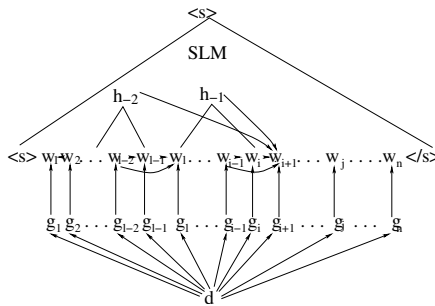
If the recursive factorization respects to a graph  $\mathcal{G}$ , then we have a Bayesian network (Lauritzen 1996). But broadly speaking, the recursive factorization can respect to a more complicated representation other than a graph which has a fixed set of nodes and edges.

All of the weighted grammars and automata can be described as directed Markov random fields, so is the (simplified) structural language model as developed by Jelinek and Chelba (Chelba and Jelinek 2000, Jelinek 2004).

### 3 Simplified Lexical and Semantic Enhanced Structural Language Model

We now describe the composite simplified structural language model enhanced by trigrams and PLSA models, which respectively encode the local lexical information of word co-occurrence and global-spanning semantic content at document level over the entire corpus.

When we combine trigram and PLSA models with SSLM to build a new generative language model, the constructor moves remain unchanged, the predictor however generates the next word  $w_i$  not only depending on the two left-most exposed headwords  $h_{-2}, h_{-1}$  but also the previous two words  $w_{i-2}, w_{i-1}$  as well as the current semantic content  $g_i \in \mathcal{G}$  with probability  $\theta(w_i | w_{i-2} w_{i-1} h_{-2} h_{-1} g_i)$ . Figure 1 illustrates the structure of the simplified lexical and semantic enhanced structural language model.



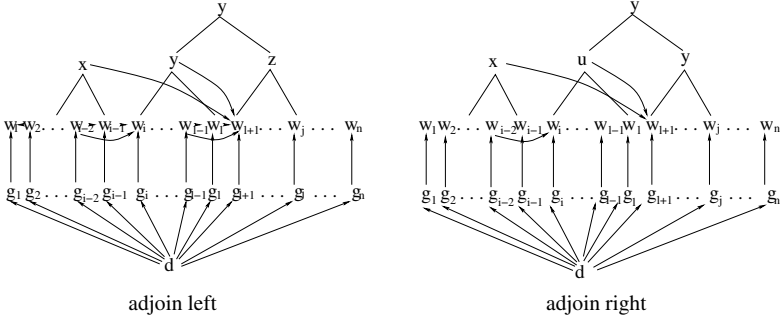
**Fig. 1.** Simplified lexical and semantic enhanced structural language model where the hidden information is the parse tree  $t$  and semantic content  $g$

#### 3.1 Computing the Probability of a Sentence

The inside probability  $p_\theta(d, w_{i+1}^j, y[i, j] | w_i, x)$  of sentence  $W$  in document  $d$  is defined as the probability of the word subsequence  $w_{i+1}^j = w_{i+1}, \dots, w_j$  are generated and  $y$  becomes the headword of the phrase  $w_i, \dots, w_j$  given that  $x$

is the last exposed headword preceding time  $i$  and the word  $w_i$  is generated. Figure 2 illustrates two situations on how the phrase  $w_i, \dots, w_j$  is generated.

The first way as illustrated in Figure 2 to generate  $w_{i+1}, \dots, w_j$  and create a phrase spanning  $\langle i, j \rangle$  whose headword is  $y$ , given that the headword of the preceding phrase is  $x$  and the word  $w_i$  is generated is the following: (i) a string  $w_{i+1}, \dots, w_l$  is generated, (ii) a phrase spanning  $\langle i, l \rangle$  is formed whose headword is  $y$ , (iii) the word  $w_{l+1}$  is generated from its preceding two words  $w_{l-1}, w_l$  and preceding two headwords  $x, y$  by averaging all possible semantic content  $g$  in document  $d$ , (iv) the string  $w_{l+1}, \dots, w_j$  is generated and the span  $\langle l+1, j \rangle$  forms a phrase whose headword is  $z$ , (v) finally, the two phrases are merged into one phrase with headword  $y$  via constructor move, adjoin left.



**Fig. 2.** Diagram illustrating inside probability recursions: adjoin left vs adjoin right

The second way as illustrated in Figure 2 to generate  $w_{i+1}, \dots, w_j$  and create a phrase spanning  $\langle i, j \rangle$  whose headword is  $y$ , given that the headword of the preceding phrase is  $x$  and the word  $w_i$  is conceptually the same as the first situation with modular chance, that is, in (ii) a phrase spanning  $\langle i, l \rangle$  is formed with headword is  $u$ , then in (iii) the word  $w_{l+1}$  is generated from its preceding two words  $w_{l-1}, w_l$  and preceding two headwords  $x, u$  by averaging all possible semantic content  $g$  in document  $d$ , thus (iv) the string  $w_{l+1}, \dots, w_j$  is generated and the span  $\langle l+1, j \rangle$  forms a phrase whose headword is  $y$ , (v) finally, the two phrases are merged into one phrase with headword  $y$  via constructor move, adjoin right.

Thus the inside probability  $p_\theta(w_{i+1}^j, y[i, j] | w_i, x), \forall j > i, i = 0, 1, \dots, n$  can be recursively computed by the following formula,

$$\begin{aligned}
 p_\theta(d, w_{i+1}^j, y[i, j] | w_i, x) &= \sum_{l=i}^{j-1} \sum_z \left( \sum_{g_{l+1} \in \mathcal{G}} \theta(w_{l+1} | x, y, w_{l-1}, w_l, g_{l+1}) \theta(\mathbf{null} | x, y) \right. & (1) \\
 &\theta(g_{l+1} | d) \left. \right) p_\theta(d, w_{i+1}^l, y[i, j] | w_i, x) p_\theta(d, w_{l+2}^j, z[l+1, j] | w_{l+1}, y) \theta(\mathbf{left} | y, z) \\
 &+ \sum_{l=i}^{j-1} \sum_z \left( \sum_{g_{l+1} \in \mathcal{G}} \theta(w_{l+1} | x, u, w_{l-1}, w_l, g_{l+1}) \theta(\mathbf{null} | x, u) \right. \\
 &\left. \theta(g_{l+1} | d) \right) p_\theta(d, w_{i+1}^l, u[i, j] | w_i, x) p_\theta(d, w_{l+2}^j, y[l+1, j] | w_{l+1}, u) \theta(\mathbf{right} | u, y)
 \end{aligned}$$

The boundary conditions for the above recursion are given as

$$p_\theta(d, w_{i+1}^i, y[i, i]|w_i, x) = p_\theta(d, h(w_i) = y|w, h_{-1}(T^{i-1}) = x) = 1 \quad \forall x \in W^{(i-1)}, y = w_i$$

and the probability of a sentence  $W$  in document  $d$  is given by

$$p_\theta(d, W) = p_\theta(d, w_2^{n+1}, </s> [1, n+1]|w_1, <s>) \left( \sum_{g_1 \in \mathcal{G}} \theta(w_1 | <s> g_1) \theta(g_1 | d) \right) \quad (2)$$

### 3.2 Computing the Probability of Next Word Given the Sentence Prefix

In order to compute the left-to-right probability of a word given the sentence prefix, define  $p_\theta(d, w_0^{i+1}, x)$  to be the probability that the sequence  $w_0, w_1, \dots, w_i, w_{i+1}$  in document  $d$  is generated such that the last exposed headword of the parse tree for the string  $w_0, w_1, \dots, w_i$  is  $x$  and define the set of words  $W^{(i)} = \{w_0, w_1, \dots, w_i\}$ . Then  $\forall x \in W^{(i)}$ , we have the following recursive formula,

$$p_\theta(d, w_0^{l+1}, x) = \sum_{i=1}^l \sum_{y \in W^{(i-1)}} p_\theta(d, w_0^i, y) p_\theta(w_{i+1}^l, x[i, l]|w_i, y) \left( \sum_{g_{l+1} \in \mathcal{G}} \theta(w_{l+1}|y, x, w_l, w_{l-1}, g_{l+1}) \theta(\text{null}|y, x) \theta(g_{l+1}|d) \right) \quad (3)$$

with the initial conditions

$$p_\theta(d, w_0^1, x) = \begin{cases} \sum_{g_1 \in \mathcal{G}} \theta(w_1 | <s> g_1) \theta(g_1 | d) & \text{if } x = <s> \\ 0 & \text{otherwise} \end{cases}$$

Thus we have

$$p_\theta(d, w_0, w_1, \dots, w_i, w_{i+1}) = \sum_{x \in W^{(i+1)}} p_\theta(d, w_0^{i+1}, x)$$

and

$$p_\theta(d, w_{i+1}|w_0, w_1, \dots, w_i) = \frac{\sum_{x \in W^{(i+1)}} p_\theta(d, w_0^{i+1}, x)}{\sum_{x \in W^{(i+1)}} p_\theta(d, w_0^i, x)} \quad (4)$$

### 3.3 Finding the Most Probable Parse

Denote  $\hat{p}_\theta(w_{i+1}^j, y[i, j]|w_i, x)$  as the probability of the most probable sequence of moves that generate the words  $w_{i+1}, \dots, w_j$  with  $y$  being the headword of the phrase  $w_i, w_{i+1}, \dots, w_j$ . Then finding the most probable parse can be recursively obtained by changing the sum sign in the inside probability computation into a max sign (Kschischang et al. 2001).

$$\begin{aligned}
 & \hat{p}_\theta(d, w_{i+1}^j, y[i, j]|w_i, x) \\
 = & \max \left\{ \max_{l \in \{i, j-1\}, z} \left[ \left( \sum_{g_{l+1} \in \mathcal{G}} \theta(w_{l+1}|x, y, w_{l-1}, w_l, g_{l+1}) \right. \right. \right. \\
 & \left. \left. \left. \theta(\mathbf{null}|x, y)\theta(g_{l+1}|d) \right) \hat{p}_\theta(d, w_{i+1}^l, y[i, j]|w_i, x) \hat{p}_\theta(d, w_{l+2}^j, z[l+1, j]|w_{l+1}, y)\theta(\mathbf{left}|y, z) \right], \right. \\
 & \left. \max_{l \in \{i, j-1\}, z} \left[ \left( \sum_{g_{l+1} \in \mathcal{G}} \theta(w_{l+1}|x, u, w_{l-1}, w_l, g_{l+1}) \right. \right. \right. \\
 & \left. \left. \left. \theta(\mathbf{null}|x, u)\theta(g_{l+1}|d) \right) \hat{p}_\theta(d, w_{i+1}^l, u[i, j]|w_i, x) \hat{p}_\theta(d, w_{l+2}^j, y[l+1, j]|w_{l+1}, u)\theta(\mathbf{right}|u, y) \right] \right\}
 \end{aligned} \tag{5}$$

The boundary conditions for the above recursion are given as

$$\hat{p}_\theta(d, w_{i+1}^i, y[i, i]|w_i, x) = p_\theta(d, h(w_i) = y|w, h_{-1}(T^{i-1}) = x) = 1 \quad \forall x \in W^{(i-1)}, y = w_i$$

then the probability of most probable parse in sentence  $W$  in document  $d$  is given by

$$p_\theta(d, W, \hat{T}) = \hat{p}_\theta(d, w_2^{n+1}, < /s > [1, n+1]|w_1, < s >) \left( \sum_{g_1 \in \mathcal{G}} \theta(w_1 | < s > g_1) \theta(g_1 | d) \right) \tag{6}$$

### 3.4 Training Algorithm for Simplified Lexical and Semantic Enhanced Structural Language Model

Without writing down explicit formula of likelihood function, Jelinek (2004) has derived an EM-type parameter estimation algorithm for SLM whose structure is considerably more complex than that of a probabilistic context free grammar, and it is a generalization of the inside-outside algorithm (Baker 1979). The derivation is conceptually based on relevant frequency counting for discrete data which is a common practice for estimating PCFGs (Lari and Young 1990). In this section, we derive parameter estimation algorithm for the composite simplified lexical and semantic enhanced structural language model from the general EM algorithm (Dempster et al. 1977). This leads to a further generalization of inside-outside algorithm proposed by Jelinek (2004).

Similar as in the composite trigram/PCFG/PLSA model (Wang et al. 2005), the likelihood of the observed data  $\mathcal{W}$  under this composite language model can be written as below:

$$\mathcal{L}(\mathcal{W}, \theta) = \prod_{d \in \mathcal{D}} \left( \prod_l \left( \sum_{G_l} \left( \sum_T p_\theta(d, W_l, G_l, T) \right) \right) \right) \tag{7}$$

where

$$\begin{aligned}
 p_\theta(d, W_l, G_l, T) = & \prod_{d \in \mathcal{D}} \left( \prod_l \left( \prod_{g \in \mathcal{G}} \theta(g|d)^{c(d, W_l, g)} \prod_{u, v, h_{-2}, h_{-1} \in \mathcal{V}, g \in \mathcal{G}} \right. \right. \\
 & \left. \left. \theta(w|uvh_{-2}h_{-1}g)^{c(uvh_{-2}h_{-1}g; d, W_l, T)} \prod_{h_2, h_1 \in \mathcal{V}, a \in \mathcal{A}} \theta(a|h_{-2}h_{-1})^{c(h_{-2}h_{-1}a; d, W_l, T)} \right) \right)
 \end{aligned}$$

where  $p_\theta(d, W_l, G_l, T)$  is the probability of generating sentence  $W_l$  in document  $d$  with parse tree  $T$  and semantic content sequence  $G_l$ ,  $c(d, W_l, g)$  is the count of semantic content  $g$  in document  $d$ ,  $c(uvwh_{-2}h_{-1}g; d, W_l, T)$  is the count of trigrams  $uvw$  with  $w$ 's two left most exposed headwords  $h_{-2}h_{-1}$ , parse tree  $T$  and semantic content  $g$  in sentence  $W_l$  of document  $d$ , and  $c(h_{-2}h_{-1}a; d, W_l, T)$  is the count of constructor move  $a$  conditioning on  $h_{-2}h_{-1}$  in sentence  $W_l$  of document  $d$  with parse tree  $T$ . The parameters  $\theta(g|d)$ ,  $\theta(w|uvh_{-2}h_{-1}g)$ ,  $\theta(a|h_{-2}h_{-1})$  are normalized so that  $\sum_{w \in \mathcal{V}} \theta(w|uvh_{-2}h_{-1}g) = 1$ ,  $\sum_{a \in \mathcal{A}} \theta(a|h_{-2}h_{-1}) = 1$  and  $\sum_{g \in \mathcal{G}} \theta(g|d) = 1$ .

Following Lafferty's (2000) derivation of the inside-outside formulas for updating the parameters of PCFGs and Wang et al.'s (2005) derivation of the generalized inside-outside formulas for updating the parameters of the composite trigram/PCFG/PLSA models from a general EM (Dempster et al. 1977) algorithm, we derive the generalized inside-outside algorithm for the simplified lexical and semantic enhanced structural language model.

To apply the EM algorithm, we consider the auxiliary function

$$Q(\theta', \theta) = \sum_d \sum_l \sum_{G_l} \sum_T p_\theta(G_l, T|d, W_l) \log \frac{p_{\theta'}(d, W_l, G_l, T)}{p_\theta(d, W_l, G_l, T)}$$

Because of the normalization constraints, the re-estimated parameters of the composite model are then the normalized conditional expected counts:

$$\begin{aligned} \theta'(a|h_{-2}h_{-1}) &= \frac{\sum_{d \in \mathcal{D}} \sum_l \sum_{G_l} \sum_T p_\theta(G_l, T|d, W_l) c(h_{-2}h_{-1}a; d, W_l, T)}{\text{normalization over } a} \\ \theta'(w|uvh_{-2}h_{-1}g) &= \frac{\sum_{d \in \mathcal{D}} \sum_l \sum_{G_l} \sum_T p_\theta(G_l, T|d, W_l) c(uvwh_{-2}h_{-1}g; d, W_l, T, g)}{\text{normalization over } w} \quad (8) \\ \theta'(g|d) &= \frac{\sum_l \sum_{G_l} \sum_T p_\theta(G_l, T|d, W_l) c(d, W_l, g)}{\text{normalization over } g} \end{aligned}$$

Thus we need to compute the conditional expected counts, the numerators of (8).

In general, the sum requires summing over an exponential number of parse trees due to combinatorial explosion of possible parse trees. However, just as with standard PCFGs (Lafferty 2000) and composite trigram/PCFG/PLSA model (Wang et al. 2005), it is easy to check that the following equations still hold

$$\begin{aligned} \sum_{G_l} \sum_T p_\theta(G_l, T|d, W_l) c(h_{-2}h_{-1}a; d, W_l, T) &= \frac{\theta(a|h_{-2}h_{-1})}{p_\theta(d, W_l)} \frac{\partial p_\theta(d, W_l)}{\partial \theta(a|h_{-2}h_{-1})}, \quad (9) \\ \sum_{G_l} \sum_T p_\theta(G_l, T|d, W_l) c(uvwh_{-2}h_{-1}g; d, W_l, T, g) &= \frac{\theta(w|uvh_{-2}h_{-1}g)}{p_\theta(d, W_l)} \frac{\partial p_\theta(d, W_l)}{\partial \theta(w|uvh_{-2}h_{-1}g)}, \\ \sum_{G_l} \sum_T p_\theta(G_l, T|d, W_l) c(d, W_l, g) &= \frac{\theta(g|d)}{p_\theta(d, W_l)} \frac{\partial p_\theta(d, W_l)}{\partial \theta(g|d)} \end{aligned}$$

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

Now the central problem then becomes to recursively represent the probability of a sentence,  $W$  in a document,  $p_\theta(d, W)$ , in terms of its parameters. Following Jelinek’s derivation for structural language model (Jelinek 2004), we first derive formulas for computing  $p_\theta(d, W, x, y[i, j])$ , the probability that  $W$  is produced by some tree  $T$  that has a phrase spanning  $\langle i, j \rangle$  whose headword is  $y$  and its immediately preceding exposed headword is  $x$ . More formally,

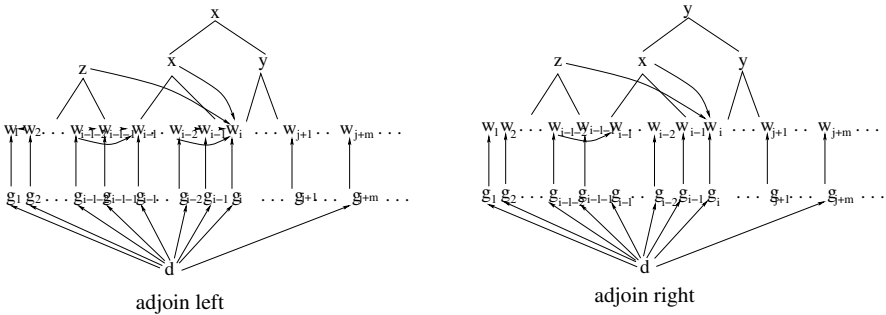
$$\begin{aligned}
 p_\theta(d, W, x, y[i, j]) &\doteq p_\theta(d, w_0, w_1, \dots, w_{n+1}, h_{-1}(w_0, \dots, w_{i-1}) = x, h(w_i, \dots, w_j) = y) \\
 &= p_\theta(d, w_0, w_1, \dots, w_{n+1}, h_{-1}(w_0, \dots, w_{i-1}) = x) \\
 &\quad p_\theta(d, w_{i+1}, \dots, w_j, h(w_i, \dots, w_j) = y | w_i, h_{-1}(w_0, \dots, w_{i-1}) = x) \\
 &\quad p_\theta(d, w_{j+1}, \dots, w_{n+1} | h_{-1}(w_0, \dots, w_{i-1}) = x, h(w_i, \dots, w_j) = y)
 \end{aligned}$$

The middle term is an inside probability and can be recursively calculated. We need a way to compute the “outside probability” which is the product of the outer terms of the above equation,

$$\begin{aligned}
 p_\theta(d, w_0^i, w_{j+1}^{n+1}, x[i-1]; y[i, j]) &\doteq p_\theta(d, w_0, w_1, \dots, w_{n+1}, h_{-1}(w_0, \dots, w_{i-1}) = x) \quad (10) \\
 p_\theta(d, w_{j+1}, \dots, w_{n+1} | h_{-1}(w_0, \dots, w_{i-1}) = x, h(w_i, \dots, w_j) = y)
 \end{aligned}$$

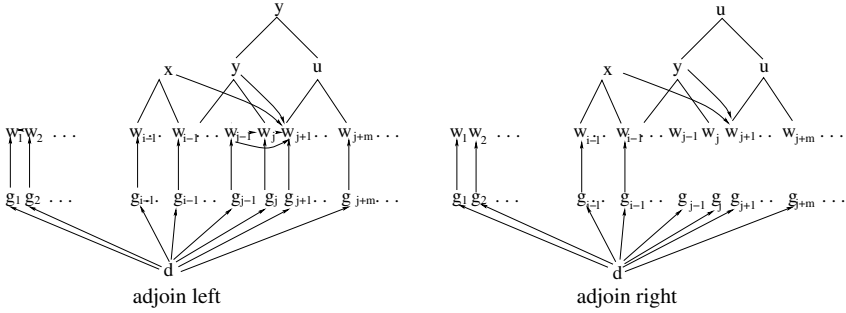
We thus have

$$\begin{aligned}
 p_\theta(d, W) &= \sum_{i,j} p_\theta(d, W, x, y[i, j]) \\
 &= \sum_{i,j} p_\theta(d, w_0^i, w_{j+1}^{n+1}, x[i-1]; y[i, j]) p_\theta(d, w_{i+1}^j, y[i, j] | w_i, x) \quad (11)
 \end{aligned}$$



**Fig. 3.** Diagram illustrating outside probability recursions: first term in (11) is recursively computed with adjoin left and adjoin right being used respectively

Figures 3- 4 illustrate four cases that the outside probability  $p_\theta(d, w_0^i, w_{j+1}^{n+1}, x[i-1]; y[i, j])$  can be recursively obtained by the inside and outside probabilities. In the first two cases, the first term in the definition of outside probability,  $p_\theta(d, w_0, w_1, \dots, w_{n+1}, h_{-1}(w_0, \dots, w_{i-1}) = x)$  remains unchanged and it is the second term that is recursively represented by inside and outside probabilities,



**Fig. 4.** Diagram illustrating outside probability recursions: second term in (11) is recursively computed with adjoin left and adjoin right being used respectively

and the difference between these two cases is on whether constructor move, adjoin left, or adjoin right is used. Similarly in the last two cases, the second term in the definition of outside probability,  $p_\theta(d, w_{j+1}, \dots, w_{n+1} | h_{-1}(w_0, \dots, w_{i-1}) = x, h(w_i, \dots, w_j) = y)$  remains unchanged and it is the first term that is recursively represented by inside and outside probabilities, and again the difference between these two cases is on whether constructor move, adjoin left, or adjoin right is used. This leads to the four double sums on the right-hand side of the following recursive formulas,

$$\begin{aligned}
 & p_\theta(d, w_0^i, w_{j+1}^{n+1}, x[i-1]; y[i, j]) \tag{12} \\
 = & \sum_{k=1}^{i-1} \sum_{z \in W^{i-k-1}} \left[ p_\theta(d, w_0^{i-1}, w_{j+1}^{n+1}, z[i-k-1]; x[i-k, j]) p_\theta(d, w_{i-k-1}^{i-1}, x[i-k, i-1] | w_{i-1}, z) \right. \\
 & \left. \left( \sum_{g_i \in \mathcal{G}} \theta(w_i | z, x, w_{i-2}, w_{i-1}, g_i) \theta(\mathbf{null} | z, x) \theta(g_i | d) \right) \theta(\mathbf{left} | x, y) \right] \\
 + & \sum_{k=1}^{i-1} \sum_{z \in W^{i-k-1}} \left[ p_\theta(d, w_0^{i-1}, w_{j+1}^{n+1}, z[i-k-1]; y[i-k, j]) p_\theta(d, w_{i-k-1}^{i-1}, x[i-k, i-1] | w_{i-1}, z) \right. \\
 & \left. \left( \sum_{g_i \in \mathcal{G}} \theta(w_i | z, x, w_{i-2}, w_{i-1}, g_i) \theta(\mathbf{null} | z, x) \theta(g_i | d) \right) \theta(\mathbf{right} | x, y) \right] \\
 + & \sum_{m=i}^{n-j+1} \sum_{z \in W_{j+1}^{j+m}} \left[ p_\theta(d, w_0^i, w_{j+m+1}^{n+1}, x[i-1]; y[i, j+m]) p_\theta(d, w_{j+2}^{j+m}, u[j+1, j+m] | w_{j+1}, y) \right. \\
 & \left. \left( \sum_{g_{j+1} \in \mathcal{G}} \theta(w_{j+1} | x, y, w_{j-1}, w_j, g_{j+1}) \theta(\mathbf{null} | x, y) \theta(g_{j+1} | d) \right) \theta(\mathbf{left} | y, u) \right] \\
 + & \sum_{m=i}^{n-j+1} \sum_{z \in \mathcal{W}_{j+1}^{j+m}} \left[ p_\theta(d, w_0^i, w_{j+m+1}^{n+1}, x[i-1]; u[i, j+m]) p_\theta(d, w_{j+2}^{j+m}, u[j+1, j+m] | w_{j+1}, y) \right. \\
 & \left. \left( \sum_{g_{j+1} \in \mathcal{G}} \theta(w_{j+1} | x, y, w_{j-1}, w_j, g_{j+1}) \theta(\mathbf{null} | x, y) \theta(g_{j+1} | d) \right) \theta(\mathbf{right} | y, u) \right]
 \end{aligned}$$

with the boundary condition

$$p_{\theta}(d, w_0^1, w_{n+1}^{n+2}, x[0]; y[1, n+1]) = \begin{cases} \sum_{g_1 \in \mathcal{G}} \theta(w_1 | \langle s \rangle g_1) \theta(g_1 | d) \mathbf{f} \ x = \langle s \rangle, y = \langle /s \rangle \\ 0 & \text{otherwise} \end{cases}$$

By (12), the outside probability can be recursively represented by a full set of model parameters. Thus by (11), calculating the derivative of the probability of a sentence  $W_l$  can be done recursively via calculating the derivative of outside probability. Then by (9), we have

$$\begin{aligned} \sum_{G_l} \sum_T p_{\theta}(G_l, T | d, W_l) c(h_{-2} h_{-1} a; d, W_l, T) &= \sum_i \sum_j c_l(d, x, y, i, j, \mathbf{a}) \quad \forall a \in \mathcal{A}, \\ \sum_{G_l} \sum_T p_{\theta}(G_l, T | d, W_l) c(uvw h_{-2} h_{-1} g; d, W_l, t, g) &= \sum_i \sum_j c_l(d, x, y, i, j, \mathbf{null}) \\ &\quad \delta(u = w_{j-1}, v = w_j, w = w_{j+1}), \\ \sum_{G_l} \sum_T p_{\theta}(G_l, T | d, W_l) c(d, W_l, g) &= \sum_i \sum_j c_l(d, g, i, j) \end{aligned}$$

where the quantities  $c_l(d, x, y, i, j, \mathbf{a})$  and  $c_l(d, g, i, j)$  are calculated by the following recursions,

$$\begin{aligned} c_l(d, x, y, i, j, a = \mathbf{left}) &= \frac{1}{p_{\theta}(d, W_l)} p_{\theta}(d, w_{i+1}^j, y[i, j] | w_i, x) \\ &\quad \sum_z \sum_{k=1}^{i-1} \left[ p_{\theta}(d, w_0^{i-1}, w_{j+1}^{n+1}, z[i-k-1]; x[i-k, j]) p_{\theta}(d, w_{i-k-1}^{i-1}, x[i-k, i-1] | w_{i-1}, z) \right. \\ &\quad \left. \left( \sum_{g_i \in \mathcal{G}} \theta(w_i | z, x, w_{i-2}, w_{i-1}, g_i) \theta(\mathbf{null} | z, x) \theta(g_i | d) \right) \theta(\mathbf{left} | x, y) \right], \\ c_l(d, x, y, i, j, a = \mathbf{right}) &= \frac{1}{p_{\theta}(d, W_l)} p_{\theta}(d, w_{i+1}^j, y[i, j] | w_i, x) \\ &\quad \sum_z \sum_{k=1}^{i-1} \left[ p_{\theta}(d, w_0^{i-1}, w_{j+1}^{n+1}, z[i-k-1]; y[i-k, j]) p_{\theta}(d, w_{i-k-1}^{i-1}, x[i-k, i-1] | w_{i-1}, z) \right. \\ &\quad \left. \left( \sum_{g_i \in \mathcal{G}} \theta(w_i | z, x, w_{i-2}, w_{i-1}, g_i) \theta(\mathbf{null} | z, x) \theta(g_i | d) \right) \theta(\mathbf{right} | x, y) \right], \\ c_l(d, x, y, i, j, a = \mathbf{null}) &= \frac{1}{p_{\theta}(d, W_l)} p_{\theta}(d, w_{i+1}^j, y[i, j] | w_i, x) \\ &\quad \sum_u \sum_{m=i}^{n-j+1} \left[ p_{\theta}(d, w_0^i, w_{j+m+1}^{n+1}, x[i-1]; y[i, j+m]) p_{\theta}(d, w_{j+2}^{j+m}, u[j+1, j+m] | w_{j+1}, y) \right. \\ &\quad \left. \left( \sum_{g_{j+1} \in \mathcal{G}} \theta(w_{j+1} | x, y, w_{j-1}, w_j, g_{j+1}) \theta(\mathbf{null} | x, y) \theta(g_{j+1} | d) \right) \theta(\mathbf{left} | y, u) \right] \\ &\quad + \sum_u \sum_{m=i}^{n-j+1} \left[ p_{\theta}(d, w_0^i, w_{j+m+1}^{n+1}, x[i-1]; u[i, j+m]) p_{\theta}(d, w_{j+2}^{j+m}, u[j+1, j+m] | w_{j+1}, y) \right. \\ &\quad \left. \left( \sum_{g_{j+1} \in \mathcal{G}} \theta(w_{j+1} | x, y, w_{j-1}, w_j, g_{j+1}) \theta(\mathbf{null} | x, y) \theta(g_{j+1} | d) \right) \theta(\mathbf{right} | y, u) \right], \\ c_l(d, g, i, j) &= \frac{1}{p_{\theta}(d, W_l)} p_{\theta}(d, w_{i+1}^j, y[i, j] | w_i, x) \end{aligned}$$

$$\begin{aligned}
& \sum_u \sum_{m=i}^{n-j+1} \left[ p_\theta(d, w_0^i, w_{j+m+1}^{n+1}, x[i-1]; y[i, j+m]) p_\theta(w_{j+2}^{j+m}, u[j+1, j+m] | w_{j+1}, y) \right. \\
& \quad \left. \left( \theta(w_{j+1} | x, y, w_{j-1}, w_j, g) \theta(\mathbf{null} | x, y) \theta(g|d) \right) \theta(\mathbf{left} | y, u) \right] \\
& + \sum_u \sum_{m=i}^{n-j+1} \left[ p_\theta(d, w_0^i, w_{j+m+1}^{n+1}, x[i-1]; u[i, j+m]) p_\theta(d, w_{j+2}^{j+m}, u[j+1, j+m] | w_{j+1}, y) \right. \\
& \quad \left. \left( \theta(w_{j+1} | x, y, w_{j-1}, w_j, g) \theta(\mathbf{null} | x, y) \theta(g|d) \right) \theta(\mathbf{right} | y, u) \right],
\end{aligned}$$

In order to be consistent to the recursive formula for SSLM derived in (Jelinek 2004), we consider  $w_{j+1}$  to derive the formula for  $c_l(d, g, i, j)$ . An alternative formula exists if we consider  $w_i$  instead.

$$\begin{aligned}
c_l(d, g, i, j) &= \frac{1}{p_\theta(d, W_i)} p_\theta(d, w_{i+1}^j, y[i, j] | w_i, x) \\
& \sum_z \sum_{k=1}^{i-1} \left[ p_\theta(d, w_0^{i-1}, w_{j+1}^{n+1}, z[i-k-1]; x[i-k, j]) p_\theta(d, w_{i-k-1}^{i-1}, x[i-k, i-1] | w_{i-1}, z) \right. \\
& \quad \left. \left( \theta(w_i | z, x, w_{i-2}, w_{i-1}, g) \theta(\mathbf{null} | z, x) \theta(g|d) \right) \theta(\mathbf{left} | x, y) \right] \\
& + \sum_z \sum_{k=1}^{i-1} \left[ p_\theta(d, w_0^{i-1}, w_{j+1}^{n+1}, z[i-k-1]; y[i-k, j]) p_\theta(d, w_{i-k-1}^{i-1}, x[i-k, i-1] | w_{i-1}, z) \right. \\
& \quad \left. \left( \theta(w_i | z, x, w_{i-2}, w_{i-1}, g) \theta(\mathbf{null} | z, x) \theta(g|d) \right) \theta(\mathbf{right} | x, y) \right]
\end{aligned}$$

Then by (9), we get the re-estimates

$$\begin{aligned}
\theta'(a | h_{-2} = x, h_{-1} = y) &= \frac{\sum_d \sum_l \sum_i \sum_j c_l(d, x, y, i, j, a)}{\sum_{a' \in \mathcal{A}} \sum_d \sum_l \sum_i \sum_j c_l(d, x, y, i, j, a')} \\
\theta'(w | uvh_{-2} = x, h_{-1} = y) &= \\
& \frac{\sum_d \sum_l \sum_i \sum_j c_l(d, x, y, i, j, \mathbf{null}) \delta(u = w_{j-1}, v = w_j, w = w_{j+1})}{\sum_{w' \in \mathcal{V}} \sum_d \sum_l \sum_i \sum_j c_l(d, x, y, i, j, \mathbf{null}) \delta(u = w_{j-1}, v = w_j, w = w')} \\
\theta'(g|d) &= \frac{\sum_l \sum_i \sum_j c_l(d, g, i, j)}{\sum_{g' \in \mathcal{G}} \sum_l \sum_i \sum_j c_l(d, g', i, j)}
\end{aligned}$$

## 4 Extension of Training to Complete Lexical and Semantic Enhanced Structural Language Models

We now extend our results to the complete structural language model which has more complex constructor than SSLM and an additional module, the **tagger**.

Each headword  $h$  is replaced by heads  $\mathbf{h} = (h^1, h^2)$  where  $h^1$  is a headword and  $h^2$  is a tag or a non-terminal. The operation of complete lexical and semantic enhanced structural language model,

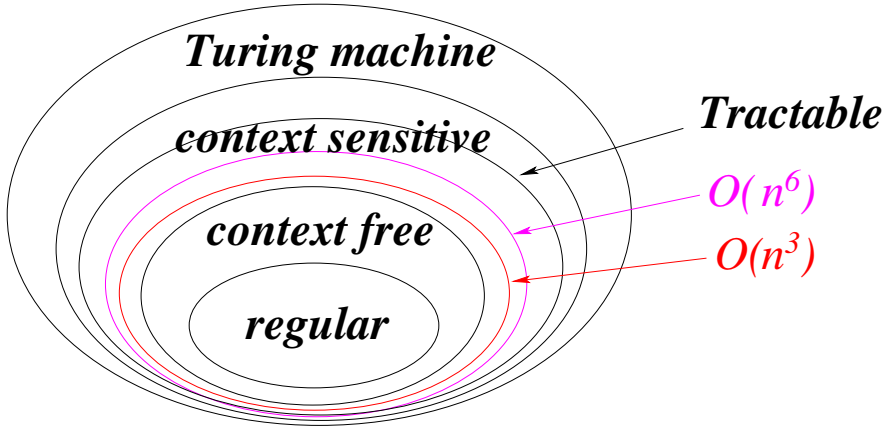
- Depending on the last two exposed heads  $\mathbf{h}_{-2}, \mathbf{h}_{-1}$ , the two preceding words  $w_{i-2}, w_{i-1}$  as well as current semantic node  $g_i$ , the predictor generates the next word  $w_i$  with probability  $\theta(w_i|w_{i-2}w_{i-1}\mathbf{h}_{-2}\mathbf{h}_{-1}g_i)$ .
- Depending on the last exposed head  $\mathbf{h}_{-1}$  and current word  $w_i$ , the tagger tags  $w_i$  by a part of speech  $f \in \mathcal{F}$  with probability  $\theta(f|w_i\mathbf{h}_{-1})$ , shifts heads left one position i.e.,  $\mathbf{h}'_{-i-1} = \mathbf{h}_{-i}, i = 1, \dots$  and generates a new last exposed head,  $\mathbf{h}'_{-1} = (h^1_{-1}, h^2_{-2}) = (w_i, f)$ .
- The constructor operates with a probability  $\theta(\mathbf{a}|\mathbf{h}_{-2}\mathbf{h}_{-1})$  where  $\mathbf{a} \in \mathcal{A} = \{\mathbf{right}|\gamma, \mathbf{left}|\gamma, \mathbf{up}|\gamma, \mathbf{null}\}$  where  $\gamma \in \Gamma$ , the set of non-terminal symbols.

The increased complexity of the complete lexical and semantic enhanced SLM mainly arises from the enlargement of headword vocabulary. The recursive formulas however can be updated with simple modular modifications.

## 5 Conclusions and Further Directions

We have shown how to integrate trigrams and PLSAs with the structural language model to build a composite generative probabilistic language model. The resulting composite language model has even more complex dependency structure but with more expressive power than the original SLM. We have studied its various stochastic properties and extended various recursive formulas with conceptually simple modular modifications, i.e., replacing  $\theta(w_i|h_{-2}h_{-1})$  with  $\sum_{g_i \in \mathcal{G}} \theta(w_i|w_{i-2}w_{i-1}h_{-2}h_{-1}g_i)\theta(g_i|d)$ , while remaining the same order computational complexity. Even though the added context-sensitiveness due to trigrams and PLSAs and violation of tree structure in the topology of the underlying random field model make the inference and parameter estimation problems plausibly intractable, these recursive formulas are nevertheless *exact* to solve these problems for the lexical and semantic enhanced SLM. The main reason rendering this being true is that the computation of the probability of a sentence can be factorized into two parts where each part can be recursively calculated and no overlapping features exist when performing the computation recursively.

Statistical natural language processing is an empirical field, nevertheless some famous published papers in NLP only described the algorithms without any experimental justification for the usefulness. For example, James Baker's 4 pages paper (Baker 1979) showed the nowadays well known inside-outside algorithm with no empirical results. Similarly in Jelinek's paper (Jelinek 2004), there is no any experimental results too, mainly due to its  $O(n^6)$  complexity where  $n$  is the length of a sentence. Our paper is in the same flavour of theirs, emphasizing algorithmic aspect. Similar as analyzed in (Jelinek 2004) for SLM, the complexity of the generalized inside-outside algorithm for the lexical and semantic enhanced SLM is in the same order as in SLM and is proportional to  $n^6$ .



**Fig. 5.** 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

Figure 5 illustrates the Chomsky's hierarchy of grammars in terms of computational complexity (Hopcroft and Ullman 1979) where there exist a tractable subclass of probabilistic context sensitive grammars with  $n^6$  time complexity as well as a subclass of probabilistic context sensitive grammars with cubic time complexity. The  $n^6$  order complexity makes all the algorithms developed in this work and in (Jelinek 2004) impractical. However various schemes as suggested in (Jelinek 2004) can be used to prune substantial fraction of entries from the charts by thresholding in the computation of inside and outside probabilities and limiting non-terminal productions. We plan to report experimental results by using these various techniques to approximately perform parameter estimations for the lexical and semantic enhanced SLM in the future.

## References

1. S. Abney, D. McAllester, and F. Pereira. (1999). Relating probabilistic grammars and automata. *Proceedings of ACL*, 542-549.
2. J. Baker. (1979). Trainable grammars for speech recognition. *Proceedings of the 97th Meeting of the Acoustical Society of America*, 547-550.
3. C. Chelba. (1999). Exploiting syntactic structure for natural language modeling. *Ph.D. Dissertation*, Johns Hopkins University.
4. C. Chelba and F. Jelinek. (2000). Structured language modeling. *Computer Speech and Language*, 14(4):283-332.
5. N. Chomsky. (1956). Three models for the description of language. *IRE Transactions on Information Theory*, 2(3):113-24.
6. 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.

7. T. Hofmann. (2001). Unsupervised learning by probabilistic latent semantic analysis. *Machine Learning*, 42(1):177-196
8. J. Hopcroft and J. Ullman. (1979). *Introduction to Automata Theory, Languages and Computation*. Addison-Wesley.
9. F. Jelinek, J. Lafferty and R. Mercer. (1992). Basic methods of probabilistic context-free grammars. *Speech Recognition and Understanding: Recent Advances, Trends, and Applications*, P. Laface and R. De Mori, (Editors), 347-360, Springer-Verlag.
10. F. Jelinek. (1998). *Statistical Methods for Speech Recognition*. MIT Press.
11. F. Jelinek and C. Chelba. (1999). Putting language into language modeling. *Proceedings of the 6th EuroSpeech Communication and Technology*, 1-6.
12. 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.
13. 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.
14. F. Kschischang, B. Frey and H. Loeliger. (2001). Factor graphs and the sum-product algorithm *IEEE Transactions on Information Theory*, 47(2):498-519.
15. J. Lafferty. (2000). A derivation of the inside-outside algorithm from the EM algorithm. *IBM Research Report* 21636.
16. 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.
17. S. Lauritzen. (1996). *Graphical Models*. Oxford Press.
18. C. Shannon. (1948). A mathematical theory of communication. *Bell System Technical Journal*, 27(2):379-423.
19. S. Wang, S. Wang, R. Greiner, D. Schuurmans and L. Cheng. (2005). Exploiting syntactic, semantic and lexical regularities in language modeling via directed Markov random fields. *The 22nd International Conference on Machine Learning*, 953-960.
20. D. Younger. (1967). Recognition and parsing of context free languages in time  $N^3$ . *Information and Control*, 10:198-208.