Factorial Hidden Markov Models for Full and Weakly Supervised Supertagging

by

Srivatsan Ramanujam, B.E

Thesis

Presented to the Faculty of the Graduate School of

The University of Texas at Austin

in Partial Fulfillment

of the Requirements

for the Degree of

Master of Arts

The University of Texas at Austin

August 2009

Factorial Hidden Markov Models for Full and Weakly Supervised Supertagging

Approved by
Supervising Committee:

_____

_____

Dedicated to Appa, Amma, Partha and Priya

Acknowledgments

This thesis is the result of my joint work with my advisers Dr. Jason Baldridge and Dr. Ray Mooney. I am extremely grateful to Prof Baldridge who has been a source of encouragement and guidance. I am also very grateful to Prof Mooney for providing me with this opportunity to work on my Masters Thesis and for his timely advice and directions. I would also like to express my thanks to the UT Austin Natural Language Learning Reading Group. Many of our discussions during the meetings increased my appreciation and interest in NLP. I would also like to thank Prof Sharon Goldwater for answering my questions on her work, which this thesis builds upon. I am also fortunate to have a friend like V Balakrishnan (JHU) whose mathematical prowess I can only hope to emulate one day. The numerous discussions I have had with him has increased my understanding of this beautiful area of Computer Science. Finally, I thank my parents, my brother and my sister-in-law, for making all of this possible.

Srivatsan Ramanujam

The University of Texas at Austin

August 2009

Factorial Hidden Markov Models for Full and Weakly Supervised Supertagging

Srivatsan Ramanujam, M.A.
The University of Texas at Austin, 2009

Supervisor: Raymond Mooney

For many sequence prediction tasks in Natural Language Processing, modeling dependencies between individual predictions can be used to improve prediction accuracy of the sequence as a whole. Supertagging, involves assigning lexical entries to words based on lexicalized grammatical theory such as Combinatory Categorial Grammar (CCG).

Previous work has used Bayesian HMMs to learn taggers for both POS tagging and supertagging separately. Modeling them jointly has the potential to produce more robust and accurate supertaggers trained with less supervision and thereby potentially help in the creation of useful models for new languages and domains.

Factorial Hidden Markov Models (FHMM) support joint inference for multi-

ple sequence prediction tasks. Here, I use them to jointly predict part-of-speech tag and supertag sequences with varying levels of supervision. I show that supervised training of FHMM models improves performance compared to standard HMMs, especially when labeled training material is scarce. Secondly, FHMMs trained from tag dictionaries rather than labeled examples also perform better than a standard HMM. Finally, I show that an FHMM and a maximum entropy Markov model can complement each other in a single step co-training setup that improves the performance of both models when there is limited labeled training material available.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

For many sequence prediction tasks in Natural Language Processing, modeling dependencies between individual predictions can be used to improve prediction accuracy of the sequence as a whole. For example, chunking [28] involves dividing words in a sentence into syntactically related non-overlapping, non-recursive phrases (like verb phrases or noun phrases etc). Training data is available in form of a sequence of words in sentences, annotated with their corresponding POS tag and a chunk tag. In these sequences, the POS tags have useful correlations with the chunk tags, so joint inference can be quite useful.

1.1 Supertagging

Supertagging [4], involves assigning lexical entries to words based on lexicalized grammatical theory such as Combinatory Categorial Grammar (CCG) [25, 2]. For example, the English verb *join* has the POS VB and the CCG category $((S_b\backslash NP)/PP)/NP$ in CCGbank [18]. This category indicates that *join* requires a noun phrase to its left, another to its right, and a prepositional phrase to the right of that. Supertags are thus far more detailed and numerous than POS tags. Every lexical item has as

many supertags as the number of different syntactic contexts in which the item can appear. Recently there is increased interest on supertagging as they are being used as features for tasks such as Machine Translation [6, 17].

1.2 Motivation

Chunking and supertagging can both be modeled using a two-stage cascade of Hidden Markov Models (HMMs) [23]. POS tags are first predicted from the observed words in the first stage of the pipeline; then the chunk tags or supertags are predicted from those POS tags in the next stage. Alternatively, both sequences can be jointly labeled, from the words. Unlike cascaded models, where the errors from one pipeline propagate (and in the process get compounded) to the next stage, joint inference circumvents this problem by predicting the labels in both the stages of the pipeline simultaneously. Joint inference also allows greater sharing of information across different levels of prediction [13]. Factorial Hidden Markov Models (FHMMs) [15] have been successfully applied to joint prediction for part-of-speech (POS) tagging and noun-phrase chunking [13] for the CoNLL2000 dataset [28].

Here, I apply FHMMs to supertagging for the categories defined in CCGbank [18] for English, a harder sequence prediction task than chunking task due to its much higher degree of lexical ambiguity. Fully supervised maximum entropy Markov models have been used for cascaded prediction of POS tags followed by supertags [10]. Here, we seek to learn supertaggers given only a POS tag dictionary and supertag dictionary or a small amount of material labeled with both types of information. Previous work has used Bayesian HMMs to learn taggers for both POS tagging [16] and supertagging [3] separately. Modeling them jointly has the potential to produce more robust and accurate supertaggers trained with less supervision and thereby potentially help in the creation of useful models for new languages and domains.

1.3 Related Work

This thesis closely follows the work of [13] [3] and [16]. [13] uses joint inference to address the problem of NP chunking in a fully supervised setting. He uses FHMMs for jointly labelling POS and NP chunk tags and reported accuracy results for POS tags in the high 90s and for chunk tags, in the low 90s. However, he does not consider the entire set of 22 chunk types and does not report F1 scores on the same. For text chunking, similar POS accuracies were achieved in this thesis, without using smoothing for emission probabilities and also with the entire set of 22 chunk types. This thesis primarily addresses supertagging and especially in the weakly supervised setting.

[3] uses a bi-tag HMM for weakly supervised training using EM with and without using grammar informed initialization of the CCG transition probabilities. We only consider the prior probability of occurrence of categories based on their complexity and do not take into account the rules of CCG formalism in deciding the transition probabilities. Our FHMMB model outperforms that of [3] by over 14%, while considering the entire CCG set (without applying any frequency cut-off for filtering out insignificant tags).

[16] uses a Bayesian tritag HMM (BHMM) for POS tagging and considers three different scenarios. A weakly supervised setting with fixed hyperparameters α and β, hyper parameter inference (learning the optimal values for α and β) and hyper parameter inference with varying corpus size and dictionary knowledge. Our bitag HMM achieved results close to what was reported by her BHMM on a random 24000 word subset of the WSJ. In all our experiments, we have kept the test set separate from the training set from which the dictionary was built, however this distinction is not made by her. By choosing the test set as a subset of the training set using which the tag dictionary was built, one circumvents the out of vocabulary words problem. We believe that is not a realistic scenario. We were able to achieve

$\sim 5\%$ improvement in CCG accuracy on creating the tag dictionaries from the entire Penn Treebank dataset [22].

[21] have also used a factorial model for performing joint labeling of the POS and chunk tags but by using Dynamic Conditional Random Fields (DCRF). The advantage of using an FHMM over DCRF is the ability to perform weakly supervised training using a generative model like the FHMM. Even in the supervised training scenario, FHMM has the advantage of lower training time when compared to discriminative training models like DCRF.

To the best of my knowledge, this is the first work on joint inference in the Bayesian framework for supertagging.

1.4 Contributions

I make the following key contributions in this thesis. First, through the Bayesian FHMMs, I show that joint inference improves supervised supertag prediction (compared to HMMs), especially when labelled training data is scarce. Second, I show how FHMMs trained on tag dictionaries also outperform HMMs. Finally, I show how, when training data is limited, the generative FHMMs and a maximum entropy Markov model (discriminative model) can bootstrap each other, in a single round co-training setup, to complement each other for supertagging.

1.5 Outline

The rest of this thesis is organized as follows. In chapter 2, I review some computational preliminaries, and provide the basis for the Bayesian approach followed in this thesis. I also discuss the Dirichlet distribution and its significance in acting as the priors in the models. In chapter 3, I describe the datasets and the sequence prediction problems in greater detail. I review the different models explored and

also describe the Gibbs sampling algorithm used for Inference in all the models.

In chapter 5, I describe the supervised training experiments for both chunking and supertagging. In this chapter, I also discuss the results from the single round co-training of the FHMM and C&C, a maximum entropy Markov model [10]. In chapter 6, I describe the weakly supervised supertagging experiments, where the models are trained on tag dictionaries. I also discuss the latent middle layer experiments, where instead of using the POS layer from the tag dictionary for the middle layer, it is treated as latent and learned in a fully unsupervised manner. Finally I present conclusions in chapter 7.

Chapter 2

Foundations

In this chapter, I review some computational preliminaries on inference methods in graphical models, and provide the basis for the Bayesian approach followed in this thesis. I then discuss the Dirichlet distribution and its significance in acting as the priors in the models.

2.1 Estimation Methods

In this section I review some of the mathematical background that is helpful in understanding the problem of inference in graphical models.

Most probablistic models used in NLP employ the Bayes Rule, which defines the probability of a hypothesis h (for example any linguistic annotation of a sequence of words), given observed data d (for example, a collection of sentences) as:

$$P(h|d) = P(d|h)P(h)/P(d)$$

Where *posterior* $P(h|d)$, is expressed as being proportional to the product of the *likelihood* $P(d|h)$ and the *prior* $P(h)$. The denominator $P(d)$ acts as a normalizing term and is independent of the hypothesis in consideration, so we generally

ignore it. The prior signifies the innate learning bias in our model (how well does h conform to our idea of a good hypothesis) and it is independent of the observed data. The likelihood gives a measure of the *closeness* of the fit, of the hypothesis on the data.

There are three common approaches to selecting a hypothesis given observated data, for a given model, viz. Maximum Likelihood, Maximum-a-posteriori and Bayesian methods (Model Averaging).

2.1.1 Maximum Likelihood

The objective of maximum likelihood estimation (MLE) is to select hypothesis $\hat{h}$ for which the likelihood $P(d|h)$ is maximum:

$$\hat{h} = \arg\max_{h} P(d|h)$$

This approach does not favour any hypothesis *a priori* and hence the prior can be considered as being uniform over the hypothesis space. In many applications in the supervised learning setting, the maximum likelihood estimate can be directly be obtained based on the frequency counts from the training data. In the unsupervised learning setting, a general class of algorithms known as Expectation Maximization [11], can be employed to obtain maximum likelihood estimates of the latent variables in the model.

The common pitfalls of the MLE approach is its susceptibility to getting stuck at local maxima and the inability to encode *apriori* knowledge into the model. Further, MLE approach tends to either overfit or underfit the data (depending on the learning bias). For example, in case of linear regression if we constrain the model to be linear in the coefficients as well, we might obtain a line which may not match the data well enough (underfitting). On the other hand, if we relax the constraint on the coefficients (allowing them to be of any arbitrary degree) we will end up with

a curve which exactly fits the training data but which may perform poorly on a independent test set (overfitting). While there are ways and means of discarding poor hypotheses (ex. through cross validation of the model on an independent development set) such methods are usually time consuming as it entails having to enumerate the entire hypothesis space.

2.1.2 Maximum Aposteriori

In the *maximum a posteriori* (MAP) approach, we make use of informative priors which express our preference for certain hypotheses over others. Our objective then, is to select the hypothesis $\hat{h}$ that maximizes the posterior $P(h|d)$:

$$\hat{h} = \arg\max_{h} P(d|h)P(h)$$

The additional term in the form of the prior $P(h)$ can be viewed as a *regularization* term. The priors act in penalizing models which have a more complex structure. We prefer simpler models to explain the data, over more sophisticated ones, all other things being equal.(*Occam's Razor*). While the MAP approach overcomes some of the pitfalls of the MLE approach, such as being less vulnerable to underfitting/overfitting or the ability to encode apriori knowledge, it is however, sensitive to the choice of the prior distribution. In the absence of a well-grounded understanding of the priors, MAP estimation might perform poorly as they are point estimates.(The mode of the posterior need not be a representative of the posterior distribution). MAP estimation is thus not considered to be truly Bayesian although we make use of Bayesian rule and informative priors in them.

2.1.3 Bayesian Methods

Both MLE and MAP are point estimates of the best hypothesis. When training data is limited or when the model is less constrained, point estimates tend to perform

poorly. The key principle in the Bayesian approach is not just the usage of a prior, but to average over all variables that are unknown. In this approach we integrate over the probability of all hypotheses to directly obtain an estimate of the posterior. Thus Bayesian model averaging generally tends to be more robust against overfitting/underfitting.

$$P(h|d) = \int_h P(d|h)P(h)dh$$

In most real applications, it will be difficult to obtain direct samples from the posterior. We will hence be making use of a class of sampling algorithms called Markov Chain Monte Carlo methods, to sample from the posterior distribution. For a detailed introduction to MCMC methods, please refer [1].

2.2 Dirichlet Prior

Priors have a significant influence in determining the hidden tags/categories for given sequence of words, especially in the weakly supervised learning setting. Linguistic structure typically is characterized by a sparsity in its distribution. The multinomials describing the probability of occurrence of each tag/category corresponding to the words in a collection of sentences, is typically sparse. This sparsity can be captured by the use of Dirichlet distribution for the priors.

Dirichlet distribution is the generalization of the two dimensional Beta distribution, to the K-dimensional case. The probability density function of a K-dimensional dirichlet, with hyperparameters α_1, α_2,...,α_k is given by:

$f(x_1, \ldots, x_{K-1}; \alpha_1, \ldots, \alpha_K) = \frac{1}{\mathrm{B}(\alpha)} \prod_{i=1}^{K} x_i^{\alpha_i - 1}$.

Which holds, such that $x_1, x_2, \ldots, x_{k-1} > 0$ and $x_k = 1 - x_1 - x_2 - \cdots - x_{k-1}$. Here $\mathrm{B}(\alpha)$, acts as a normalizing constant. Dirichlet priors are the conjugate of the multinomial distribution and this property helps us obtain a closed form solution of

the posterior.

2.2.1 Polya's Urn

Intuitively, the Dirichlet distribution can be described by the Polya's Urn process. Consider an urn filled with balls of K different colors, with α_1, α_2,...,α_k balls of each color respectively. We begin by picking a ball at random from the urn, and place it back into the urn while including another ball of the same color. This process is repeated. As the number of draws approaches infinity, the distribution of the balls in the urn approaches the K-dimensional Dirichlet distribution.

It can be seen that as the number of draws increases, the effect of adding a new ball of the same color as the one picked, has a diminishing effect on the distribution of balls in the Urn. This diminishing return becomes more evident when we start with larger values of α_1, α_2,...,α_k etc. The hyperparameters thus determine the distribution of the multinomial sampled from the Dirichlet distribution.

2.2.2 Symmetric Vs Asymmetric

If the hyperparameters are all the same, they are referred to as Symmetric Dirichlet priors and they are called Asymmetric otherwise. When all hyperparameters are equal to 1, the resulting Dirichlet distribution favours multinomials that correspond to the Uniform distribution. When one or more of the hyperparameters are closer to 0, the resulting Dirichlet distribution favours multinomials that are sparse. We will be tapping into this property of the Dirichlet distribution, in making an informed decision on the priors, for our models.

2.2.3 Effect of varying hyperparameters

Figures 2.1,2.2, 2.3, and 2.4 shows the probability distribution of the multinomial in three variables, sampled from a Dirichlet distribution, for different values of the

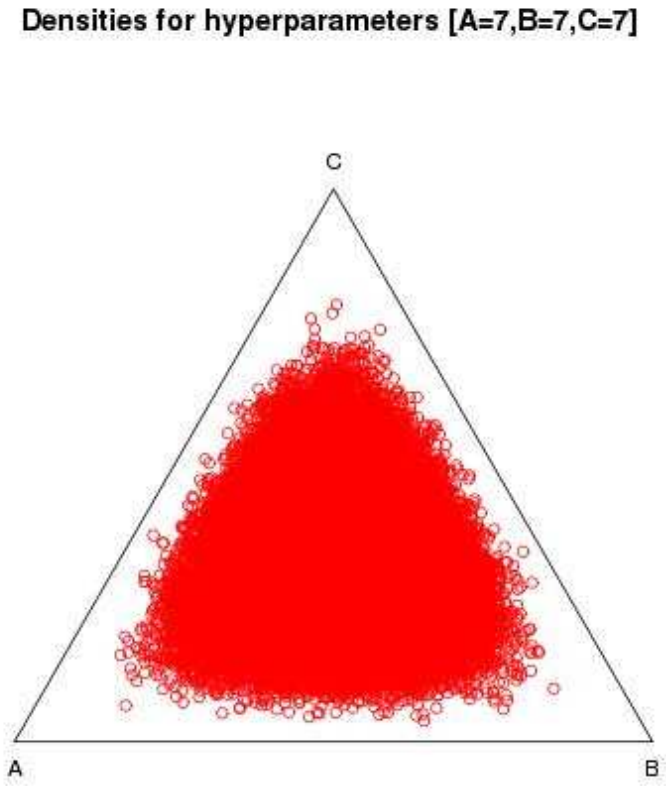

Figure 2.1: Multinomial Densities for Hyperparameters [A=7,B=7,C=7]

hyperparameters of the distribution. We can think of the probability of a component as a force acting from the corresponding vertex of the simplex, that attempts to pull away any point on the simplex towards it.

As seen in Figure 2.1, high values for the hyper-parameters for each of the K-components, leads to a Dirichlet distribution where multinomials closer to uniform have a higher probability of occurrence.

If we look at it from the Polya's Urn analogy, starting with considerably large number of balls of each color would lead to little influence of the sampling and

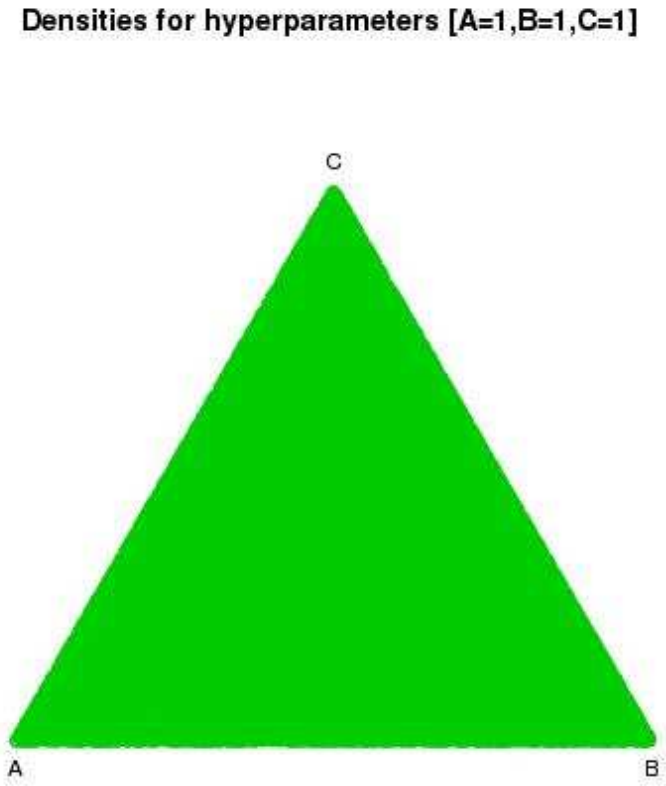

Figure 2.2: Multinomial Densities for Hyperparameters [A=1,B=1,C=1]

replacement procedure, on the distribution of the balls in the Urn.

As seen in Figure 2.2, setting all the hyper-parameters to 1, results in a Dirichlet distribution that favours all multinomials equally. This achieves the same effect as *add-one smoothing*, i.e. apriori knowledge does not favour one hypothesis over another.

As the hyper-parameters are set to values closer to zero, the resulting Dirichlet distribution favours multinomials that are sparse (one or more variables have a much higher/lower probability of occurrence than the rest). This is typical of many

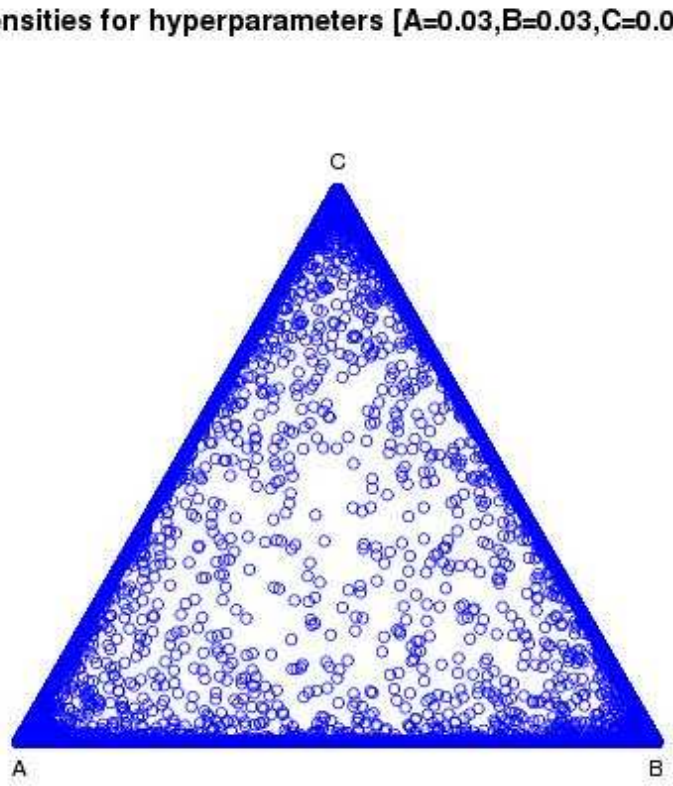

Figure 2.3: Multinomial Densities for Hyperparameters [A=0.03,B=0.03,C=0.03]

NLP applications. The multinomials might usually represent for example, the tag transition distribution (in case of POS tagging) or the word emission distribution.

Typically each tag might be able to combine with only a limited number of other tags and similarly each tag might be associated only with a limited number of words. Our apriori knowledge of such sparsity in the transition and emission distributions can be encoded by the choice of such smaller values for the hyperparameters in the Dirichlet distribution.

Asymmetric hyperparameters of the Dirichlet distribution more closely re-

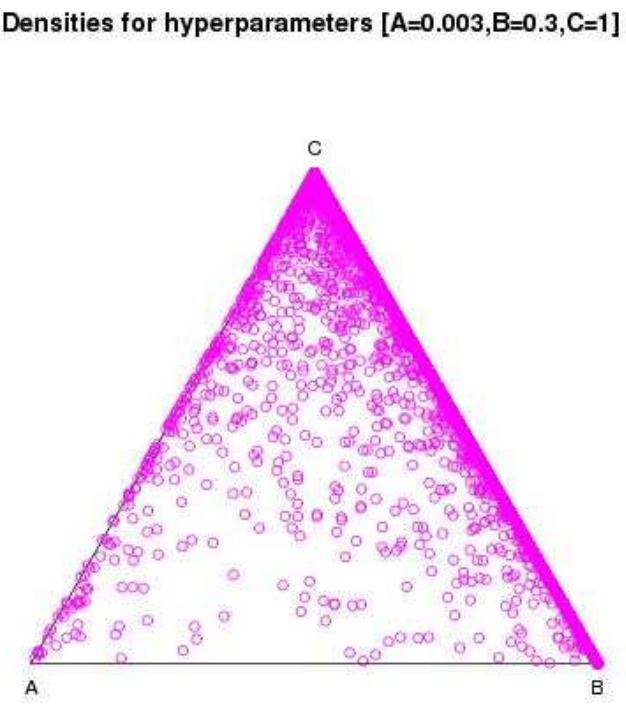

Figure 2.4: Multinomial Densities for Hyperparameters [A=0.03,B=0.3,C=1.0]

semble the preference for certain categories over others. In case of POS tags, this apriori knowledge is unavailable (as it is usually corpora dependent). With CCG tags however, the rules of Categorial Grammar can be employed, to choose values for each component of the asymmetric prior.

[3] made use of the *complexity* of a category (inversely proportional to the number of recursive categories contained within it) along with its ability to combine with all other categories, for the smoothing in the CCG transition matrix in his HMM model. We can use this grammar informed initialization in two ways. One approach would be to can directly plug the approach of [3] in the Bayesian FHMM models discussed in this paper or as it will be explained later in the experiments,

we could use these priors to obtain the initial random sample of tag sequences to boostrap our models.

Another significance of Dirichlet priors being popular choice in many NLP application, is its property of conjugacy with the multinomial distribution. Conjugacy is the property of a distribution D, which when used as the prior $P(\theta)$, combines with the likelihood $P(x|\theta)$ which belongs to a distribution L, resulting in a posterior $P(\theta|x)$, that is a similar distribution to D (with possibly different values for its parameters). This property of algebraic convenience, greatly helps in obtaining closed form expression for the posterior.(For another example of conjugacy, the Gaussian family is self-conjugate, i.e using a Gaussian prior, with a Gaussian likelihood will give a posterior that is also Gaussian).

Chapter 3

Data

Both text chunking and supertagging involve the prediction of two state sequences, corresponding to an observed word sequence. The generative process can be thought of as two parallel hidden state sequences (POS tags and chunk/supertags) generating an observed word sequence and the task is to invert this process and infer the original hidden states.

We use two datasets for this problem, viz. a) The CoNLL2000 dataset [28] b) The CCGbank dataset [18].

The Figure .3.1 and Figure 3.2 summarize some key attributes of the CoNLL2000 and the CCGbank datasets respectively. It can be seen that the percentage of out-of-vocabulary words (OOV) in the CoNLL2000 dataset is quite large compared to

Feature	Count
Training Sentences	8936
Testing Sentences	2012
Unique Words	19122
Unique POS Tags	44
Unique Chunk Tags	22
OOV Words (in Test)	3302
%OOV Words (in Test)	6.96

Figure 3.1: CoNLL2000 Dataset

Feature	Count
Training Sentences	38015
Testing Sentences	5435
Unique Words	43062
Unique POS Tags	48
Unique CCG Tags	1241
OOV Words (in Test)	3627
%OOV Words (in Test)	2.86

Figure 3.2: CCGbank Dataset

No.Sentences	38015
Avg. POS type ambiguity	1.16
Avg. POS token ambiguity	2.19
Avg. CCG type ambiguity	1.71
Avg. CCG token ambiguity	18.71
Avg. Pair token ambiguity	20.19

Figure 3.3: POS and CCG ambiguity in training set

the CCGbank dataset. The number of training sentences in the CCGbank dataset far exceeds those in the CoNLL2000 dataset, however the number of unique tags (1241) in the former is much larger than that of the latter.

Figure. 3.3 summarizes the ambiguities of the POS and CCG types and tokens in the CCGbank dataset. It can be seen that the larger size of the CCG tag set also translates to a greater per token ambiguity in the prediction of CCG tags, making supertagging a harder problem.

3.1 Text Chunking

In text chunking, the task is to predict the POS tags and the phrase chunks for an observed sequence of words. It is an intermediate step towards complete parsing. The sentences in this dataset are extracted from the Penn Treebank, Wall Street Journal corpus [22]. A sample sentence and its annotations, from the CoNLL2000 dataset, is shown below.

He	PRP	B-NP
reckons	VBZ	B-VP
the	DT	B-NP
current	JJ	I-NP
account	NN	I-NP
deficit	NN	I-NP
will	MD	B-VP
narrow	VB	I-VP
to	TO	B-PP
only	RB	B-NP
#	#	I-NP
1.8	CD	I-NP
billion	CD	I-NP
in	IN	B-PP
September	NNP	B-NP
.	.	O

In the example above, the second column identifies the POS tag of the word and third identifies the chunk tag, which in this case indicates if the word forms either the beginning (B-) or is in the middle (I-) of a verb phrase (VP) or a noun phrase (NP) or is outside any phrase (O).

3.2 Supertagging

In supertagging, the task is to predict the POS and the CCG tag sequence, corresponding to an observed word sequence. The categories of CCG are an inductively defined set containing elements that are either atomic elements or (curried) functions specifying the canonical linear direction in which they seek their arguments. For example, following is a collection of entries from CCGbank.

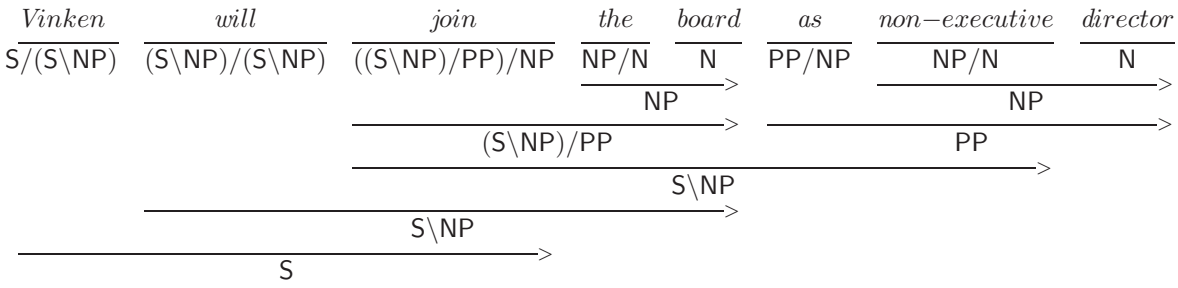

Figure 3.4: Normal form CCG derivation, using only application rules.

$$
\begin{aligned}
&\textit{the} := \mathsf{NP_{nb}/N}\\
&\textit{of} := \mathsf{(NP\backslash NP)/NP}\\
&\textit{of} := \mathsf{((S\backslash NP)\backslash(S\backslash NP)/NP}\\
&\textit{were} := \mathsf{(S_{dcl}\backslash NP)/(S_{pss}\backslash NP)}\\
&\textit{buy} := \mathsf{(S_{dcl}\backslash NP)/NP}\\
&\textit{buy} := \mathsf{((((S_b\backslash NP)/PP)/PP)/(S_{adj}\backslash NP))/NP}
\end{aligned}
$$

Here *buy* is associated with two different Supertags, based on the syntactic context in which it can occur in. CCG analyses of sentences are built up from lexical categories combining to form derived categories, until an entire sentence is reduced to a single derived category with corresponding dependencies an example of which is shown in Figure 3.4.

Unlike text chunking, supertagging conveys detailed syntactic-subcategorization information that its disambiguation is called *almost parsing* [4]. CCG is a lexicalized grammar formalism, in which syntactic rules are applied based only on the syntactic type of input [26]. The CCGbank [18, 10] is a translation of phrase structure analyses of the Penn Treebank into CCG analyses. We only consider the lexical category annotations and not the derivations, in this thesis.

Accurately assigning lexical categories to words is the key to fast parsing for CCG. [10] use a very fast maximum entropy markov model to predict lexical

categories before fully parsing a sentence based on those categories. In another light, supertagging for CCG can also be seen as a way of generalizing a lexicon by identifying categories for unseen words or unobserved word/category pairs. The performance of the C&C supertagger relies on the existence of CCGbank, which required considerable manual effort to create. It is thus of interest to build accurate supertaggers, and also to do so with a little supervision as possible in order to support the creation of grammars and annotated resources for other domains and languages.

Chapter 4

The Models

Factorial HMMs generalize HMMs to handle multiple sequences and their interactions [15]. Many real world observations are caused by the interactions of multiple causes which are typically independent. It has been applied extensively in modeling loosely coupled sequences. We consider three different models in this paper: a HMM model and two FHMM models which we call FHMMA and FHMMB.

4.1 Hidden Markov Model

The Standard HMM model shown in Figure. 4.1 models a generative process in which a POS tag (or a chunk/CCG tag) emits a word in each time instance and also its successor.

4.2 Factorial Hidden Markov Model A

Our FHMMA model is shown in Figure 4.2. We consider this as a generative process which produces a sequence of POS tags and for each POS tag in the sequence, it generates a chunk/CCG tag and each of the chunk/CCG tags, in turn emits a word.

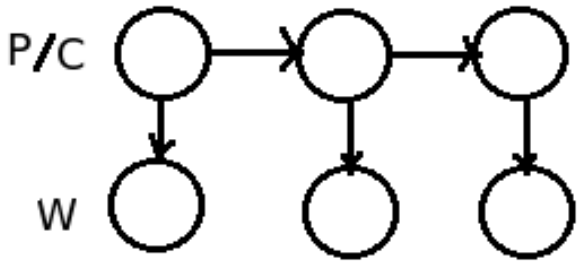

Figure 4.1: Standard HMM

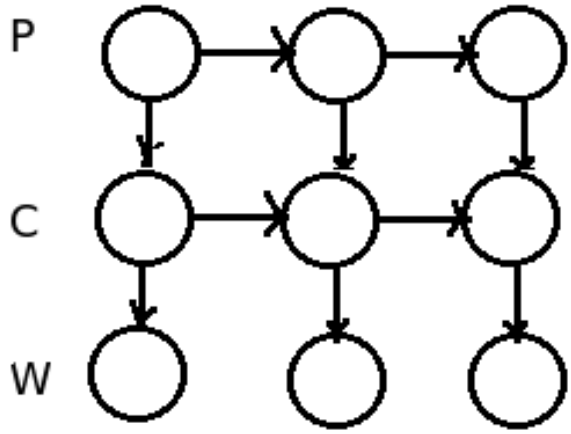

Figure 4.2: Factorial HMM Model A

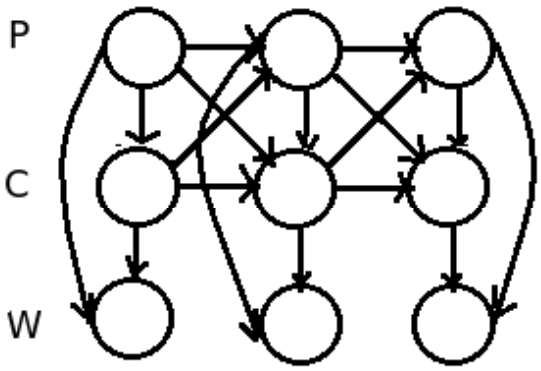

Figure 4.3: Factorial HMM Model B

4.3 Factorial Hidden Markov Model B

The second model we consider is FHMMB shown in Figure 4.3. This model has a greater interlinking of POS and chunk/CCG tags in adjacent time slices. The generative process in this case can be explained as a POS and chunk/CCG tag pair from the previous time slice emitting the POS tag in the current time slice, which along with the POS, chunk/CCG pair from the previous time slice, emits the chunk/CCG tag of the current time slice. The POS and chunk/CCG tag of the current time slice then, together emit a word.

In all three models, our objective is to invert this generative process and infer the POS and chunk/CCG tag sequence, from a given word sequence. The probability of the hidden state given the observed word for each of the models HMM, FHMMA and FHMMB is given in Figures 4.1, 4.2 and 4.3 respectively.

4.4 Being Bayesian

A model is defined with parameters θ (for example, the transition/emission probabilities), few observed variables w (in our case, the word sequence), and latent vari-

ables t which govern the hidden structure (in our case, the POS and chunk/CCG sequence). Bayesian methods in computational linguistics has drawn much attention recently [16, 20]. They have been shown to empirically outperform maximum likelihood based models.

The Bayesian approach utilizes Bayes theorem to factorize the posterior distribution $P(\theta|d)$ into the product of the likelihood $P(d|\theta)$ and the prior $P(\theta)$.

$$P(\theta|d) = P(d|\theta)P(\theta)$$

In this paper, we follow the Bayesian approach to inference, which unlike point-estimates like the Maximum-Likelihood (MLE) or Maximum Aposteori (MAP) estimates, attempts to find a distribution over the hidden variables directly, without fixing particular values for the model parameters. The distribution over latent variables given the observed data is obtained by integrating over all possible values of θ:

$$P(t|w) = \int P(t|w,\theta)P(\theta|w)d\theta$$

One of the primary advantages of the Bayesian approach is that by integrating over all values of θ, we are not relying on point-estimates which are sensitive to the variation in θ (avoids overfitting). Also, it is often the case that linguistic structures have a sparse distribution and we can choose appropriate values for the priors to suit our problem. For example in POS tagging, only a few tags have a high probability of following a given tag and most words are emitted by only a few tags [16]. We sample our priors from a symmetric Dirichlet distribution, which is the conjugate of the multinomial distribution which is what characterizes the distribution of tags in our sequence prediction tasks.

We add symmetric Dirichlet priors α and β over the transition and emission distributions of each of the three models. We then also consider tag specific priors for the emission probabilities, β.

Our bi-tag HMM (for POS tagging) can be formulated as

$$
\begin{aligned}
t_i|t_{i-1} = t, \tau^{(t,t')} &\sim Mult(t) \\
w_i|t_i = t, \omega^{(t)} &\sim Mult(\omega^{(t)}) \\
\tau^{(t,t')}|\alpha, &\sim Dirichlet(\alpha) \\
\omega^{(t)}|\beta, &\sim Dirichlet(\beta)
\end{aligned}
$$

Here we use t_i and w_i to refer to the i'th tag and word and τ refers to the state transition distribution and ω refers to the word emission distribution.

4.5 Inference

I describe the Gibbs sampling algorithm that will be used for performing inference, in all my experiments.

4.5.1 Motivation

The Forward-Backward algorithm works well for HMM models, however owing to the larger state space of the FHMM models, it becomes intractable to apply with FHMM models.The Gibbs sampler can produce a state sequence corresponding to an observation sequence of length T in time $O(KT)$ where K is the number of states. In contrast the EM algorithm or Viterbi decoding (in case of supervised learning) require $O(k^2T)$ to achieve the same. Hence Gibbs sampling is much faster than EM and because K quite large, we can run many more iterations of the Gibbs sampler

in the same time that is required to run the EM in a weakly supervised learning setting. We hence use Gibbs sampling for inference in our models.

4.5.2 Approach

The Gibbs sampler is a specific case of Markov Chain Monte Carlo (MCMC) method that is suitable for sampling from highdimensional spaces. To sample from a distribution $P(z)$, where $z = (z_1, ..., z_n)$ the Gibbs sampler proceeds by sampling and updating each z_i one at a time from $P(z_i|z - i)$, where $zi = (z_1, ..., z_{i1}, z_{i+1}, ..., z_n)$, i.e., all of the z except z_i .

This is called the point-wise collapsed Gibbs sampler. There are other variations of the Gibbs sampler which can be used, namely the explicit Gibbs sampler, where in the parameters of the graphical model are also sampled after each iteration and also the blocked version of the Gibbs sampler which uses a dynamic programming algorithm (Forward-Backward) to sample a set of parameters at the same time. This is done to satisfy the ergodicity of the Markov chain while also maintaining the detailed balance property. For example, we might want to sample both a POS tag and a CCG tag for a given word, simultaneously. This is because some combinations of POS tags and CCG tags may not be permitted. In such cases the blocked Gibbs sampler can be used. For a more detailed discussion of these versions of the Gibbs Samplers, refer [5] and [14].

The equations for sampling a POS, chunk/CCG pair through Gibbs sampling, in each of the three models HMM, FHMMA and FHMMB are summarized in Figure 4.1, Figure 4.2, and Figure 4.3 respectively (where MB(.) denotes the Markov Blanket of a node in the graphical model). For the HMM model, the POS and CCG tags are sampled independent of each other. In case of the FHMM models, the interlinks between the POS, CCG nodes in the model, decides the interdependency during the joint inference of the POS, CCG tag sequence.

$$P(t_i|MB(t_i)) \sim P(t_i|t_{i-1})P(t_{i+1}|t_i)P(w_i|t_i)$$
$$P(c_i|MB(c_i)) \sim P(c_i|c_{i-1})P(c_{i+1}|c_i)P(w_i|c_i) \tag{4.1}$$

$$P(t_i, c_i|MB(t_i, c_i)) \sim P(t_i|t_{i-1})P(c_i|t_i, c_{i-1})P(t_{i+1}|t_i)P(c_i + 1|t_{i+1}, c_i)P(w_i|c_i) \tag{4.2}$$

$$P(t_i, c_i|MB(t_i, c_i)) \sim P(t_i|t_{i-1}, c_{i-1})P(c_i|t_{i-1}, c_{i-1}, t_i)P(t_{i+1}|t_i, c_i)\times$$
$$P(c_{i+1}|t_i, c_i, t_{i+1})P(w_i|t_i, c_i) \tag{4.3}$$

Figure 4.4: Sampling equations for each model, for the Gibbs sampler

In our experiments, we noticed that in the weakly supervised learning setting, the Gibbs sampler converged in $\sim 50-100$ iterations and for the supervised training setting the improvement in accuracy was not very significant beyond 100 iterations.

Chapter 5

Supervised Training Experiments

In this chapter, I first demonstrate the performance of the models in the supervised training setting, where the models are trained with annotated sentences from the CoNLL2000 and CCGbank datasets. I then show how the generative FHMM model can be used with the discriminative C&C model [10], in a single-round co-training setting, for supertagging.

5.1 Text Chunking

The HMM and FHMM models are trained on the 8936 sentences of the CoNLL2000 training dataset and is tested on 2012 sentences of the test set. In the HMM Model, for out of vocabulary words, the entire set of possible POS tags and chunk tags is considered in the dictionary for that word. In the FHMM models however, to save on computation time, only the top-50 most frequently occuring POS, chunk pairs in the training set is considered in the dictionary for out of vocabulary words.

There are 45 POS tags and 22 chunk tags in this dataset. Experiments were

run using the Dirichlet priors of $\alpha = \beta = 1.0$, with a burn-in time of 300 iterations of the Gibbs sampler. All results are reported as the stochastic average over 5 trials. Simulated annealing was used in the Gibbs sampler, by varying the temperature ϕ from an initial value of 2.0 to a final value of 0.08 following the approach in [16]. It was observed that using simulated annealing gave $\sim 1 - 1.5\%$ improvement in the accuracy of POS tags and $\sim 3 - 4\%$ improvement in F1 score for the text chunking.

5.1.1 Results

The results of this experiment are tabulated in Table 5.1. A plot of the performance is shown in Figure 5.1. While the state of the art text chunkers using MaxEnt models and SVM [28] achieved F1 scores of over of 93% on this task, our generative models do quite well. (Considering the fact that these models don't use any features from the neighbourhood of a word, unlike other discriminative models reported in [28]).

What particularly stands out from Table 5.1, is the performance of the factorial models with minimal supervision (100 annotated sentences). The factorial models outperform the HMM model by more than 20 points in the text chunking task. FHMMB model outperforms FHMMA. This can be understood by the fact that as the POS and NP chunks are actually correlated, so inducing a greater level of interdependence in the model structures, actually helps improving the prediction accuracy.

The POS accuracies on this task was 94.07% for FHMMB and 92.67% for FHMMA and 92.22% for the HMM model, with the entire training set and 69.37%,68.08% and 59.9% using a training set of 100 sentences respectively. Note that these values were obtained without using any tag specific smoothing for the transition and emission matrices respectively.

This establishes that modeling dependencies between individual predictions

No.Sen	HMM	FHMMA	FHMMB
100	37.57	57.39	61.20
1000	58.74	70.92	74.98
ALL	74.40	79.95	84.22

Table 5.1: F1 scores on Text Chunking (CoNLL2000 shared task)

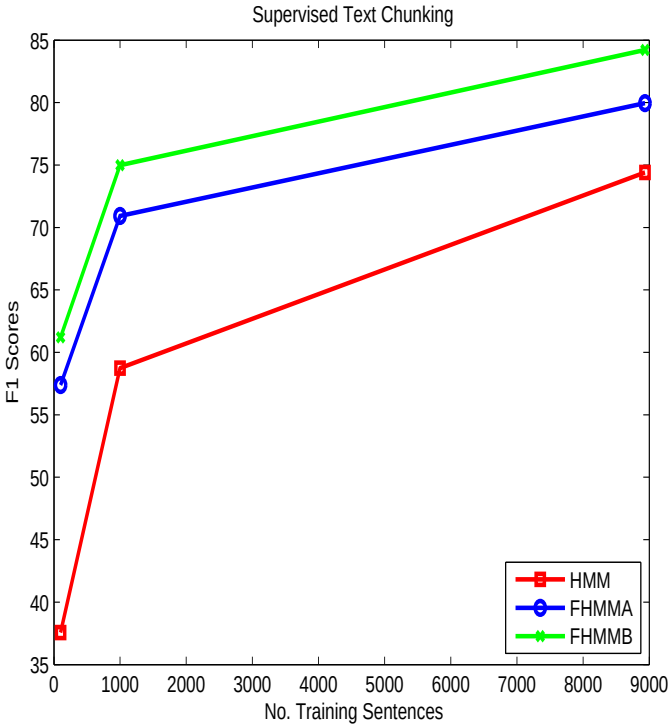

Figure 5.1: Performance on Text Chunking (CoNLL2000 shared task)

No. Sen.	HMM	FHMMA	FHMMB
100	14.09	38.79	57.75
1000	43.23	61.25	73.36
10000	69.77	76.12	83.70
ALL	77.2	79.57	85.17

Table 5.2: Supervised Supertagging with $\alpha = 1.0$, for ambiguous types.

can be used to improve prediction accuracy of the sequence as a whole and forms the motivation of applying FHMMs to supertagging.

5.2 Supertagging

In this experiment, we use the training and test sets used by [3] from the CCGbank. Experiments are run with varying amounts of training data of 100, 1000, 10000 and the entire set of 38015 sentences in the training set. Two different values for the transition prior α were used ($\alpha = 1.0$ and $\alpha = 0.05$) for the CCG tags. The emission prior β was held constant at 1.0. The rest of experimental settings from the text chunking task were maintained in this experiment as well.

5.2.1 Results

The results of these experiments, for $\alpha = 1.0$ are tabulated in Table 5.2 and a plot of it is shown in Figure 5.2, and for $\alpha = 0.05$ are tabulated in Table 5.4 and a plot of which is shown in Figure 5.3 respectively. The CCG accuracy inclusive of unambiguous types, for $\alpha = 1.0$ and $\alpha = 0.05$ are tabulated in Table 5.3 and Figure 5.5 respectively.

For comparison, the results from the C&C supertagger [10] is shown in Table 5.6, which is the state-of-the-art in supervised supertagging. The performance of our models (FHMMB in particular), when there is minimal supervision, is hence quite good.

No. Sen.	HMM	FHMMA	FHMMB
100	38.81	54.89	67.24
1000	57.15	69.22	77.33
10000	74.68	79.67	85.59
ALL	80.19	82.14	86.14

Table 5.3: Supervised Supertagging with $\alpha = 1.0$ (inclusive of unambiguous types)

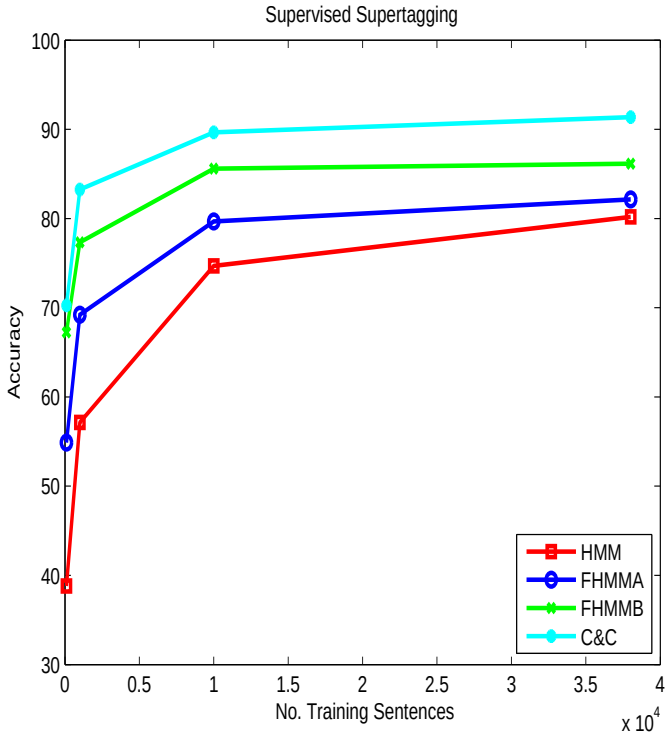

Figure 5.2: Performance on Supervised Supertagging for $\alpha = 1.0$

No. Sen.	HMM	FHMMA	FHMMB
100	17.23	39.79	58.07
1000	46.02	59.93	74.31
10000	70.49	73.79	83.85
ALL	76.65	76.98	86.21

Table 5.4: Supervised Supertagging with $\alpha = 0.05$, for ambiguous types.

No. Sen.	HMM	FHMMA	FHMMB
100	40.86	55.55	67.45
1000	59.04	68.39	78.01
10000	75.27	77.93	85.78
ALL	79.95	80.06	87.68

Table 5.5: Supervised Supertagging with $\alpha = 0.05$ (inclusive of unambiguous types)

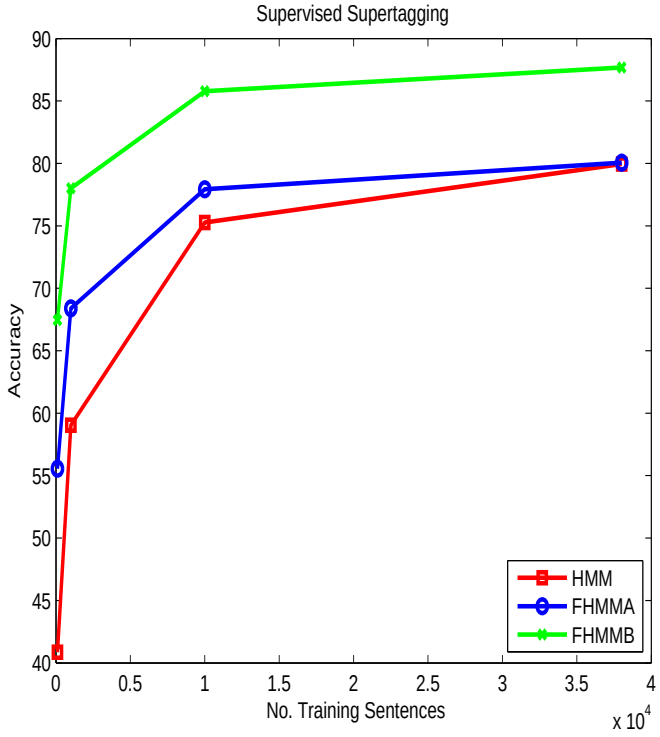

Figure 5.3: Performance on Supervised Supertagging for $\alpha = 0.05$

No. Sen.	CCG Accuracy
100	70.25
1000	83.25
10000	89.67
ALL	91.36

Table 5.6: Supervised Supertagging with C&C (inclusive of unambiguous types)

What stands out from the figures reported in the table is the performance of the FHMM models with minimal amount of training data (100 sentences). The FHMMA model achieves a 24% improvement in accuracy of CCG tags (ambiguous types alone) when compared to the HMM model and the FHMMB model achieves a 43% improvement compared to the HMM model. Between the FHMM models, FHMMB greatly outperforms FHMMA for all the training set sizes.

The prior α which determines the sparsity of transition matrix has been reported to have a greater influence on the performance of the tagger in [16] in a weakly supervised POS tagging. Notice that this can also be seen in supervised supertagging (as seen in Figure 5.3), in the models HMM and FHMMB, although the difference is not very noticeable. FHMMA performs relatively poorly when the α value is decreased.

While the state of the art POS taggers report accuracies in the range of $96 - 97\%$, the performance of the FHMMB model was comparable (95.35% for $\alpha = 0.05$ and 94.41 for $\alpha = 1.0$). The FHMMA model and the HMM model achieved 91% and 92.5% accuracy on POS tags, respectively.

5.3 Chunk and CCG specific Emission Priors

We also examined the effect of choosing a chunk/category specific emission prior β.

$$\beta = K.count(c_i, w)/|V|$$

where $count(c_i, w)$ is the number of word types associated with chunk/category c_i and $|V|$ is the vocabulary size of the words in the Lexicon and K is a multiplication constant. The idea is to allocate a higher probability mass to those categories which are more generic (can be associated with multiple word types) than those which are very specific to a small collection of words.

Setting $K = 10$ in the above formulation of β, the FHMMB model achieved an F1 score of 85.64 for the Text Chunking task, using the entire training set (with a POS accuracy of 95.06). The improvement in CCG accuracy varied between 1 to 2%, for CCG specific β.

5.4 Single Round Co-Training Experiments

In this section I discuss the single round co-training experiments with the C&C supertagger and the FHMMB model. C&C supertagger is the state-of-the-art in discriminative models for supertagging, however it cannot work on unannotated sentences as the training data. The FHMMB model performance is pretty close to C&C, the when training data is limited. In this section, both these models are used in single round of co-training setting, for supertagging.

5.4.1 Motivation

The idea behind co-training [7], is that the two participating models learn two different *views* of the data and the errors they commit are typically uncorrelated. The different views thus complement each other thereby helping one model to boost the performance of the other, by providing it with more annotated training data. Given that unannotated data is plentiful, we could use co-training iteratively to enhance the prediction performance of the two models. [19] use the C&C tagger and the TNT tagger [9] in a co-training setting, using unlabelled data for boostrapping and obtain significant performance improvements in POS tagging.

5.4.2 Approach

I start by training the FHMM model with 25, 50 and 100 training sentences respectively and use it to label the remaining sentences from the training set. The C&C supertagger is then trained on the entire collection of sentences labeled by the

Num Sen.	Alone	Bootstrapped
25	53.86	60.18
50	62.03	63.97
100	70.25	69.35

Table 5.7: C&C bootstrapped with FHMMB

Num Sen.	Alone	Bootstrapped
25	58.25	56.72
50	62.36	65.37
100	67.45	71.7

Table 5.8: FHMMB bootstrapped with C&C

FHMM model and is evaluated on the test set. I then present results of boostrapping the FHMMB model with C&C supertagger, trained on 25, 50 and 100 annotated sentences respectively.

5.4.3 Results

Table 5.4.3 shows the results of bootstrapping a C&C supertagger with the FHMMB model trained on 25, 50 and 100 annotated sentences respectively. For comparison, the standalone performance of C&C is also shown. The FHMMB model was used to annotate the remaining sentences in the training set and the C&C supertagger was trained on this larger annotated dataset and its prediction performance was tested on the test set. Table 5.4.3 shows the results of using a C&C supertagger trained on 25, 50 and 100 annotated sentences, to boostrap the FHMMB model. Again, for comparison, the standalone performance of the FHMMB model is also shown. The C&C supertagger is used to annotate the remaining sentences in the training dataset and the FHMMB model is trained on them and its prediction performance is tested on the test dataset.

Notice how the FHMMB model outperforms the C&C model when provided

with minimal supervision (25, 50 sentences), in stand alone. This makes it ideal for boostrapping more powerful discriminative models like the C&C. Clearly, from Table 5.4.3, we can see that co-training has helped improve the performance of the C&C supertagger, with minimal supervision (25, 50 sentences). Again from Table 5.4.3 we can see that the C&C supertagger aids in boosting the performance of the FHMMB model.

Thus, the results from these experiments have validated the initial intuitions that the generative FHMM model and the discriminative model C&C can complement each other and are hence suitable for co-training.

Chapter 6

Weakly Supervised Supertagging

Since annotation is costly, we are interested in automatic annotation of unlabeled sentences with minimal supervision. In the weakly supervised learning setting, we are provided with a Lexicon that provides the set of all possible POS tags and supertags for any word. We analyze the performance of our three models by varying a frequency cut-off on CCG categories in the Lexicon. In our HMM model, for every Out of Vocabulary word (OOV), the entire set of POS tags and supertags is considered for the dictionary. However in case of the FHMM models, to save on computation time, only the top 50 most frequently occuring POS, CCG pairs in the Lexicon were used for for OOV words. Similar results were obtained on using the entire set of POS, CCG pairs for the OOV words, during our trial runs.

The initial sample of CCG tag sequences corresponding to the observation sequence is drawn using probabilities based on *grammar informed initialization* as used in [3]. The POS tag corresponding to an observed word w_i is drawn uniformly at random from the set of all tags corresponding to w_i in the dictionary. In case of the FHMM models, we first draw a POS tag t_i corresponding to a word w_i uniformly

at random from the tag dictionary of w_i and then from the set of all CCG tags that have occurred with t_i and w_i in the dictionary, we randomly sample a CCG tag c_i based on its *complexity*.

Given a lexicon L, the probability of a category c_i is inversely proportional to its complexity:

$$\Lambda_i = (1/complexity(c_i))/\sum_{j \in L}(1/complexity(c_j)) \tag{6.1}$$

where $complexity(c_i)$ is defined as the number of sub-categories contained in category c_i.

6.1 Effect of frequency cut-off on Supertags

In this experiment, the performance of the models for supertagging in the weakly supervised setting is examined, where in a threshold is used to filter out infrequently occurring CCG categories from the Lexicon. Any category c, that occurs less than $k\%$ of the times with a word type w, is removed from the tag dictionary of that word, when the Lexicon is constructed. This serves in reducing the lexical ambiguity of the categories. Results of this experiment for $\alpha = 1.0$, on ambiguous CCG categories, are tabulated in Table 6.1 and a plot of the same is shown in Figure 6.1. The same results for $\alpha = 0.05$ is tabulated in Table 6.3 and its corresponding plot is shown in Figure 6.2 respectively. The CCG accuracy inclusive of unambiguous types is reported in Table 6.2 for $\alpha = 1.0$ and Table 6.4 respectively.

6.1.1 Results

The performance of the HMM model (31%) in Figure. 6.1 without any frequency cut-off on the CCG categories, is comparable to the bitag HMM of [3] that uses EM (33%). The complexity based initialization is not directly comparable to the

CCG cut-off	HMM	FHMMA	FHMMB
0.1	63.99	47.16	60.66
0.01	65.77	45.72	61.61
0.001	46.49	39.51	52.60
None	30.89	37.36	47.98

Table 6.1: Weakly Supervised Supertagging with $\alpha = 1.0$, for ambiguous types.

CCG cut-off	HMM	FHMMA	FHMMB
0.1	78.78	70.23	76.99
0.01	74.85	60.86	71.95
0.001	55.70	50.07	60.62
None	37.46	46.93	52.95

Table 6.2: Weakly Supervised Supertagging with $\alpha = 1.0$ (inclusive of unambiguous types)

CCG cut-off	HMM	FHMMA	FHMMB
0.1	59.21	47.15	62.38
0.01	45.42	42.73	51.08
0.001	27.41	36.03	35.94
None	23.02	30.5	34.03

Table 6.3: Weakly Supervised Supertagging with $\alpha = 0.05$, for ambiguous types.

CCG cut-off	HMM	FHMMA	FHMMB
0.1	76.44	70.36	77.96
0.01	60.68	58.91	64.63
0.001	40.31	47.3	47.19
None	31.74	41.3	40.67

Table 6.4: Weakly Supervised Supertagging with $\alpha = 0.05$ (inclusive of unambiguous types)

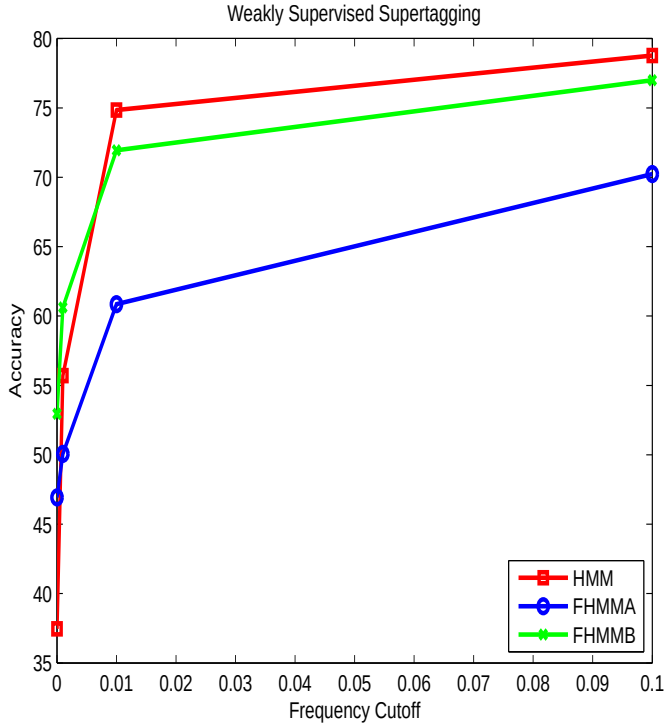

Figure 6.1: Performance on Weakly Supervised Supertagging for $\alpha = 1.0$

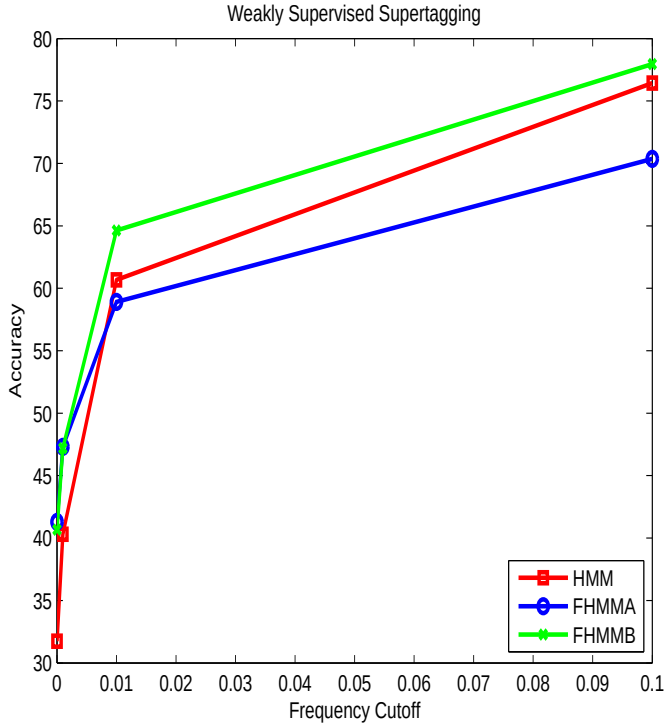

Figure 6.2: Performance on Weakly Supervised Supertagging for $\alpha = 0.05$

results in [3] (the values reported in [3] are based on a weighted combination of complexity based initialization and modified transition priors based on CCG formalism that dictates what categories can combine and what cannot). However, it is encouraging to see that when there is no cut-off based filtering of the categories, the FHMMB model (47.98%) greatly outperforms the HMM-EM model of [3] but is quite short of the 56.1% accuracy achieved by the HMM-EM model of [3] using grammar informed initialization (combination of category based initialization along with category transition rules).

While the model of [3] has the advantage of having greater *seeded knowledge* through the use of CCG formalism, the FHMM models have the advantage of using extra information in the form of the POS tag dictionary corresponding to the word sequence. We thus believe, we can achieve much better results with our FHMM models to outperform the HMM-EM of [3], on incorporating more *seeded knowledge*, based on CCG formalism.

Between the three models, without any frequency cut-off on CCG categories, FHMMB achieves over 17% improvement in the prediction accuracy of ambiguous CCG categories, over the HMM. The HMM model performs much better when there is a high level of frequency based filtering of the categories. However frequency based filtering of categories is a strong form of supervision and it might not be feasible in all real world settings (we seldom have enough training data to reasobly estimate such frequencies). Thus, the performance of the FHMMB model in the weakly is very encouraging.

The choice of the transition prior α of 0.05 lead to severe degredation in the prediction accuracy of CCG tags. Unlike POS tagging, where a symmetric transition prior of $\alpha = 0.05$ captured the sparsity of the tag transition distribution [16], in Supertagging the transition priors are asymmetric. There are rules in CCG formalism which dictate which categories can combine and those that cannot. [3] uses these

rules in performing a grammar informed initialization of the CCG transition probabilities to greatly outperform the HMM using EM model, with uniform smoothing for the category transitions. The use of asymmetric priors is examined in section 6.2.

The POS accuracies in these experiments were 83.5-85%, 84.5-86.2% and 78.3-78.4% for models FHMMB, FHMMA and HMM respectively (without any frequency cut-off on CCG categories).

6.2 Asymmetric Transition Priors

I experimented with asymmetric priors α_i and β_i for the multinomials representing the transition and emission distributions, in the FHMM models. For smaller training set sizes, asymmetric priors did better than symmetric prior of 1.0, but on the complete training set asymmetric priors did poorly compared to the symmetric prior of 1.0. The results from these experiments were thus inconclusive. The theoretical intuition that grammar informed initialization of the hyper-parameters of the Dirichlet priors should improve supertagging, did not work well in practice. This is possibly because of the values for the CCG transitions being very small compared to the values for the POS transitions and the word emissions. The choice of t_i, c_i for any w_i, was hence dominated by the tag transition and the word emissions. Further, beyond the first couple of iterations, the priors don't make a significant difference as likelihoods would have accumulated larger values compared to the priors.

6.3 Latent Middle Layer Experiments

In this section, I describe my experiments with the FHMM models in which the middle layer is treated as latent. The basic notion is to learn a grouping of categories, as the middle layer and use that instead of the POS layer.

6.3.1 Motivation

Traditionally, the lexicon for a categorial grammar specifies for each word its own category. Some categories are strongly related to one another and can be grouped into families. For example, in OpenCCG [8], categories are organized into lexical families. The families here are related to whole sets of words and is loosely connected to the notion of tree families in XTAG [12]. One way of associating words with each other is through their POS tag. Every word has a POS tag and for every family a set of POS tags can be associated, for a word to belong to that family.

The POS layer in our models is not necessarily the best intermediate annotation of words, that could help in discerning the CCG tags. Ideally we would like to have an intermediate layer with a tagset that is compact and also aids in improving the prediction of categories. Such a layer can be automatically learned (does not require a Lexicon) and in that sense it is regarded as latent. The latent layer if successful in improving the performance of the model, has the added advantage of further minimizing supervision.

Previous work on POS taggers [13] have experimented with a collapsed tagset of the Penn Treebank. The original POS tagset in the Penn Treebank consists of 47 POS tags. The collapsed subset contains only 17 tags. Refer [24] for further details about the collapsed tagset. Table 6.5 shows the results of training the FHMMB model with varying amounts of annotated sentences from the collapsed tagset dataset. For comparison, the results from the complete 47 tagset dataset is also reported. Clearly, when training data is scarce, the FHMMB model benefits more from the collapsed tagset than from the entire tagset dataset. This indicates that it is possible that a reduced tagset might be helpful in discerning the CCG tags for words, more accurately.

$\#sen$	47 tagset	17 tagset
100	67.8	69.1
1000	77.48	77.79
10000	85.59	85.32
All	87.70	87.37

Table 6.5: CCG accuracy of FHMMB model on the collapsed POS tagset, supervised training

Tagset size	CCG Accuracy
10	38.4
8	39.43
5	40.82
3	42.25
2	43.99

Table 6.6: CCG accuracy of FHMMB model for different sizes of the latent layer tagset

6.3.2 Approach

I experimented with a latent layer with a tagset of size 2, 5 and 10 respectively, for both the FHMM models. For each word, the entire tagset of the latent layer is considered, as possible tags (it is not learned from a Lexicon). For the CCG categories though, the same Lexicon (CCGbank) is made use of as before.

6.3.3 Results

Table 6.6 presents the results for the FHMMB model, for different tagset sizes for the middle layer.

The results of these experiments were not encouraging. The FHMM models perform better with smaller tagsets for the latent layer. Having a large tagset for the middle layer did not help in better performance. Table 6.7 shows the clustering of words into the different tags of the latent layer, for the FHMMB model, for a

Tag: 0 , #words: 2653
The In But he It I Mr. A He be We And That This At it They said There If As For one When – buy n't have the many Sony On for Some Japan CNN traders 34 One
Tag: 1 , #words: 1588
months years 30 be been ; , income his report the loss few days n't already this think again points week money comment never get its people have weeks it 40 just show sell do us 1 each along trade
Tag: 2 , #words: 1327
n't be month been it week 10 have not three do the sharply one common stake, rates support cash 9 still day even nine just very rate give 100 Meanwhile are 150 really move put see money longer to
Tag: 3 , #words: 7101
, . of to in and 's for that is % said on a million was by at with from are will has as market the have were – would had billion Corp. : Inc. & Co. than ; prices
Tag: 4 , #words: 5808
the a $ its it , and as to an which Mr. year company that about market but or share their stock more U.S. Friday new of in be they New this two . some other said 1987 up most

Table 6.7: Top 20 words under each tag type for the hidden middle layer, for the model FHMMB

tagset of size 5. Although the distribution of words in each cluster is not highly unbalanced, it does not have an intuitive/interpretable pattern.

The FHMMA model, produced an uninteresting clustering of words into just a single cluster, with an accuracy of prediction of 43%. Overall the latent middle layer experiments, aimed at identifying *category families* were not successful. This might indicate that sequential relationships (as in our FHMMs) aren't helpful in capturing the syntactic notion of category families. One direction of exploring this approach further would be to use alternative models employing a bag-of-words representation (ex Topic Models), to capture the concept of category families.

Chapter 7

Conclusion

In this chapter, I present the broad list of conclusions I have learned in this thesis. I started with the goal of experimenting with Factorial Hidden Markov Models, in the Bayesian framework, to tackle the problem of supertagging, with a particular emphasis on weakly supervised methods. In chapters 5 and 6 I presented my results on full and weakly supervised supertagging with the FHMM models respectively. Through these experiments I first showed that supervised training of FHMM models improves performance compared to standard HMMs, especially when labeled training material is scarce. Secondly, I showed that FHMMs trained from tag dictionaries rather than labeled examples also perform better than a standard HMM. Finally, I showed that an FHMM and a maximum entropy markov model can complement each other in a single step co-training setup that improves the performance of both models when there is limited labeled training material available.

In the remainder of this chapter I present my conclusions and chart out directions for future research. I then conclude this thesis.

7.1 Full and Weakly Supervised Supertagging

I demonstrated that joint inference in supertagging, boosts the prediction accuracy of both POS and CCG tags by a considerable margin. The improvement in prediction accuracy is more significant when training data is scarce. The results from the weakly supervised supertagging using the FHMM models was particularly encouraging, given the pressing need to reduce the amount of supervision for producing accurate taggers. What was also observed from these results is that the model (FHMMB) that made least assumptions about the independence in the input state sequences, performed the best. These general observations also hold for the CoNLL2000 shared task.

Further, the FHMM models performed particularly well when the per token ambiguity was highest, as demonstrated by the experiments varying the cut-off on CCG categories. The choice of hyper-parameters which captured the sparsity in the distribution of POS tags, was detrimental to the task of supertagging, indicating that the CCG transition distribution does not reflect the same kind of sparsity as the tag transition distribution. In supervised training experiments, an asymmetric category specific prior β increased the prediction accuracy of CCG tags.

These are certainly a small set of experiments from the vast array of possibilities we can consider. Finding good values for our hyperparameters α and β based on trial and error, which is long and tiring process that is infeasible in the worst case. We can instead learn the optimal values for these priors by searching through a range of values for α and β using the Metropolis algorithm as shown in [16].

7.2 Asymmetric Priors

Asymmetric priors failed to improve the performance of the FHMM models. The intuition that grammar informed initialization of the hyper-parameters of the Dirichlet

priors should improve supertagging, did not work in practice. I surmise that this is because of the fact that the values for the CCG transitions were very small compared to the values for the POS transitions and the word emissions. Due to this, the asymmetric prior on the CCG transitions did not make a big enough an impact, as the choice of t_i, c_i for any w_i, was dominated by the tag transition and the word emissions. Further, beyond the first couple of iterations, the priors don't make a significant difference as likelihoods would have accumulated larger values compared to the priors.

One way to explore this approach further, would be to plot the actual distribution of t_i, c_i pair for each w_i and analyze how closely does it resemble samples from a Dirichlet distribution constructed through the grammar informed initialization. It could lead to a better understanding of the sparsity of the transition and emission distributions and hence help in modeling them with a better choice of hyperparameters for the prior.

7.3 Single Round Co-Training

The results from the single round co-training experiments were encouraging. The generative FHMM model was able to catch up with the prediction accuracy of the powerful discriminative model like the C&C supertagger, when more annotated sentences were made available by a supertagger (which per se was boostrapped with a FHMM tagger trained on 50 to 100 sentences). Given the fact that unannotated sentences are available aplenty, we could use co-training using the FHMM and C&C models to obtain better prediction accuracies of supertags than either of them running as a stand-alone.

It would be interesting to see how far can co-training increase the performance of these models, for supertagging.

7.4 Latent Middle Layer

The attempt to capture the concept of CCG families, by treating learning the middle layer in a completely unsupervised fashion, was not successful. The FHMM models either learned a distribution of words that was extremely unbalanced (as in FHMMA model) or not very informative even otherwise (as in FHMMB model).

One direction of exploring this approach further would be to use alternative models employing a bag-of-words representation (ex Topic Models [27]), to capture the concept of category families.

7.5 Conclusion

From the discussions above, it is clear that there are several avenues of extending this thesis. The success of the factorial models in the Bayesian framework, is an important contribution in weakly supervised supertagging. While my work has built upon the success of Bayesian models on simpler tasks such as POS tagging, it has demonstrated the significant gains that can be achieved through joint inference while reducing annotation costs, on harder problems such as supertagging. My broad spectrum of experiments adds to the existing literature on statistical language learning.

Further, where my experiments have failed to produce interesting results, I believe they have provided valuable insights on what does not work. This should help others in improving on my work, to advance the state-of-the art in supertagging.

Bibliography

[1] C. Andreiu, N. de Freitas, A. Daucet, and M. I. Jordan. An Introduction to MCMC for Machine Learning. 2003.

[2] J. Baldridge. *Lexically Specified Derivational Control in Combinatory Categorial Grammar*. PhD thesis, University of Edinburgh, 2002.

[3] J. Baldridge. Weakly Supervised Supertagging with Grammar Informed Initialization. In *COLING*, 2008.

[4] S. Bangalore and A. K. Joshi. Supertagging: An Approach to Almost Parsing. *Computational Linguistics*, 25:237–265, 1999.

[5] J. Besag. Introduction to Markov Chain Monte-Carlo Methods. In *Mathematical Foundations of Speech and Language Processing*, pages 247–270. Springer, New York, 2004.

[6] A. Birch, M. Osborne, and P. Koehn. CCG supertags in factored statistical machine translation. In *2nd Workshop on Statistical Machine Translation*, 2007.

[7] A. Blum and T. Mitchell. Combining labeled and unlabeled data with co-training. In *ICML*, pages 92–100. Morgan Kaufmann Publishers, 1998.

[8] C. Bozsahin, G.-J. M. Kruijff, and M. White. Specifying grammars for openccg: A rough guide. 2007.

[9] T. Brants. Tnt - a statistical part-of-speech tagger. In *Proc. of the 6th Conference on Applied Natural Language Processing*, pages 224–231, 2000.

[10] S. Clark and J. Curran. Wide-coverage efficient statistical parsing with CCG and log-linear models. *Computational Linguistics*, 33(4), 2007.

[11] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the em algorithm. *JOURNAL OF THE ROYAL STATISTICAL SOCIETY, SERIES B*, 39(1):1–38, 1977.

[12] C. Doran, D. Egedi, B. A. Hockey, B. Srinivas, and M. Zaidel. Xtag system - a wide coverage grammar for english. In *In Proceedings of the 17 th International Conference on Computational Linguistics (COLING '94)*, pages 922–928, 1994.

[13] K. Duh. Joint Labelling of Multiple Sequences: A Factorial HMM Approach. In *ACL Student Research Workshop*, 2005.

[14] J. Gao and M. Johnson. Comparison of Bayesian Estimators for unsupervised Hidden Markov Model POS Taggers. In *EMNLP*, 2008.

[15] Z. Ghahramani and M. Jordan. Factorial Hidden Markov Models. In *Journal of Machine Learning Research*, 1998.

[16] S. Goldwater and T. Griffiths. A fully Bayesian approach to unsupervised part-of-speech tagging. In *ACL*, 2007.

[17] H. Hassan, K. Sima'an, and A. Way. Supertagged phrase-based statistical machine translation. In *Proceedings of the 45th ACL*, 2007.

[18] J. Hockenmaier and M. Steedman. CCG-bank: A corpus of CCG derivations and dependency structures extracted from the Penn Treebank. In *Computational Linguistics*, 2007.

[19] S. C. James, J. R. Curran, and M. Osborne. Bootstrapping pos taggers using unlabelled data. In *Proceedings of CoNLL-2003*, pages 49–55, 2003.

[20] M. Johnson, T. Griffiths, and S. Goldwater. Bayesian inference for PCFGs via Markov chain Monte Carlo. In *In Human Language Technologies: The Conference of the North American Chapter of the Association for Computational Linguistics*, pages 139–146, 2007.

[21] A. Mccallum, K. Rohanimanesh, and C. Sutton. Dynamic Conditional Random Fields for Jointly Labeling Multiple Sequences. In *NIPS Workshop on Syntax, Semantics and Statistics*, 2003.

[22] M. A. M. Mitchell P. Marcus, Beatrice Santorini. Building a large annotated corpus of English: the Penn Treebank. *Computational Linguistics*, 19, 1994.

[23] L. Rabiner. A tutorial on Hidden Markov Models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, 1989.

[24] N. A. Smith and J. Eisner. Contrastive estimation: Training log-linear models on unlabeled data. In *In Proc. of ACL*, 2005.

[25] M. Steedman. *The Syntactic Process*. The MIT Press, Cambridge Mass, 2000.

[26] M. Steedman and J. Baldridge. Combinatory categorial grammar. In *Robert Borsley and Kersti Borjars (eds.) Constraint-based approaches to grammar: alternatives to transformational syntax. Oxford: Blackwell*, 2009.

[27] M. Steyvers and T. Griffiths. *Probabilistic Topic Models*. Lawrence Erlbaum Associates, 2007.

[28] E. Tjong Kim Sang and S. Buchholz. Introduction to the CoNLL-2000 Shared Task: Chunking. In *Proceedings of CoNLL-2000*, Lisbon, Portugal, 2000.

Vita

Srivatsan Ramanujam was born in Chennai, India on the 28th of February, 1983. He spent his early childhood in Bangalore and teenage years in Chennai. He attended PSG College of Technology, Coimbatore, India, to obtain a Bachelors in Engineering, Information Technology, in 2004. After working as a Software Engineer at Computer Sciences Corporation (Hyderabad, India) and at Strand Life Sciences (Bangalore, India), he attended graduate school at the University of Texas at Austin from Aug 2007 to Aug 2009 to obtain a Masters in Computer Science.

Permanent Address: G1, Raghav Flats,
IV Main Road, Rajalakshmi Nagar, Velachery,
Chennai - 600042,
Tamil Nadu, India

This thesis was typeset with LaTeX 2$_\varepsilon$[1] by the author.

[1] LaTeX 2$_\varepsilon$ is an extension of LaTeX. LaTeX is a collection of macros for TeX. TeX is a trademark of the American Mathematical Society. The macros used in formatting this thesis were written by Dinesh Das, Department of Computer Sciences, The University of Texas at Austin, and extended by Bert Kay, James A. Bednar, and Ayman El-Khashab.