Part-of-Speech Tagging using Virtual Evidence and Negative Training

Sheila M. Reynolds and **Jeff A. Bilmes**
Department of Electrical Engineering
University of Washington
Seattle, WA 98195-2500
{sheila,bilmes}@ee.washington.edu

Abstract

We present a part-of-speech tagger which introduces two new concepts: virtual evidence in the form of an "observed child" node, and negative training data to learn the conditional probabilities for the observed child. Associated with each word is a flexible feature-set which can include binary flags, neighboring words, etc. The conditional probability of *Tag* given *Word + Features* is implemented using a factored language-model with back-off to avoid data sparsity problems. This model remains within the framework of Dynamic Bayesian Networks (DBNs) and is conditionally-structured, but resolves the label bias problem inherent in the conditional Markov model (CMM).

1 Introduction

A common sequence-labeling task in natural language processing involves assigning a part-of-speech (POS) tag to each word in the input text. Previous authors have used numerous HMM-based models (Banko and Moore, 2004; Collins, 2002; Lee et al., 2000; Thede and Harper, 1999) and other types of networks including maximum entropy models (Ratnaparkhi, 1996), conditional Markov models (Klein and Manning, 2002; McCallum et al., 2000), conditional random fields (CRF) (Lafferty et al., 2001), and cyclic dependency networks (Toutanova et al., 2003). All of these models make use of varying amounts of contextual information. In this paper, we present a new model which remains within the well understood framework of Dynamic Bayesian Networks (DBNs), and we show that it produces state-of-the-art results when applied to the POS-tagging task. This new model is conditionally-structured and, through the use of virtual evidence (Pearl, 1988; Bilmes, 2004), resolves the explaining-away problems (often described as label or observation bias) inherent in the CMM.

This paper is organized as follows. In section 2 we discuss the differences between a hidden Markov model (HMM) and the corresponding conditional Markov model (CMM). In section 3 we describe our observed-child model (OCM), introducing the notion of virtual evidence, and providing an information-theoretic foundation for the use of negative training data. In section 4 we discuss our experiments and results, including a comparison of three simple first-order models and state-of-the-art results from our feature-rich second-order OCM.

For clarity, the comparisons and derivations in sections 2 and 3 are done for first-order models using a single binary feature. The same ideas are then generalized to a higher order model with more features (including adjacent words).

2 Generative vs. Conditional Models

In this section we discuss the tradeoffs between the generative hidden Markov model (HMM) and the conditional Markov model (CMM). For pedagogical reasons, the figures and equations are for first order models with a single word-feature.

The HMM shown in Figure 1 includes a single

feature (the binary flag *isCap*) in addition to the word itself. Each observation, $o_i = (w_i, f_i)$, is a word-feature pair. Let $\mathbf{o} = \{o_i\}$ be the observation sequence and $\mathbf{s} = \{s_i\}$ be the associated tag (state) sequence. The HMM[1] factorizes the joint probability distribution over these two sequences as:

$$P(\mathbf{s}, \mathbf{o}) = \prod_i P(s_i|s_{i-1})P(w_i|s_i)P(f_i|s_i)$$

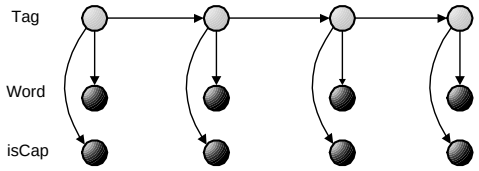

Figure 1: First order HMM.

A similar model often used for sequence labeling tasks is the conditional Markov model (CMM) which reverses the arrows between the words and the tags (Figure 2), and factorizes as:

$$P(\mathbf{s}, \mathbf{o}) = \prod_i P(s_i|s_{i-1}, w_i, f_i)P(w_i)P(f_i)$$

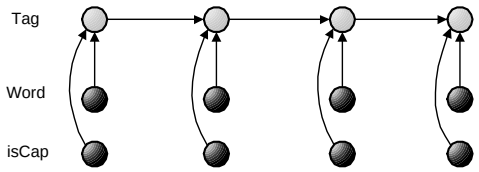

Figure 2: First order CMM.

Because the words and features are observed, this model does not require that we compute the probability of the evidence, $P(\mathbf{o})$, when finding the optimal tag sequence. The tag-sequence $\mathbf{s}$ which maximizes the joint probability $P(\mathbf{s}, \mathbf{o})$ is the same one that maximizes the *conditional* probability $P(\mathbf{s}|\mathbf{o})$. The CMM, therefore, does not require that we model the language, allowing us to focus on modeling the conditional probability of the tags given the words.

The HMM has its advantages as well, principally that it is easier to train than the CMM because it factorizes the joint probability into simpler components. The tables required for $P(s_i|s_{i-1})$ and $P(o_i|s_i)$ are significantly smaller than the one for $P(s_i|s_{i-1}, o_i)$ which may be difficult to estimate due to either data sparsity or normalization issues. One potential disadvantage of the HMM is that when it is trained using a maximum likelihood procedure, it is not necessarily encouraged to optimally classify tags due to its generative nature. One solution is to train the HMM using a discriminative procedure. Another option is to use entirely different models.

A key disadvantage of the CMM is that it makes critical statements about independence that the HMM does not: the converging arrows at each tag put the parent nodes (the previous tag and the current observation) into causal competition and as a result the model states that the previous tag is independent of the current observation. In other words, all states (tags) are independent of future observations (words). The CMM thus incorporates a strong directional bias which does not exist in the HMM.

One way to eliminate this bias is to use a CRF (Lafferty et al., 2001; McCallum, 2003), where factors over neighboring tags may use features from anywhere in the observation sequence. The CRF is discriminative and avoids label/observation bias by using a model that is constrained only in that the conditional distribution factorizes over an undirected Markov chain. However, most popular training procedures for a CRF are time-consuming and complex processes.

3 Using Virtual Evidence

Our goals in this work are to: 1) keep the discriminative nature of the CMM to the extent possible; 2) avoid label and observation bias issues; and 3) stay entirely within the DBN framework where training is relatively simple. We thus propose a new solution to the problem, which retains the discriminative conditional form of "tag given word" from the CMM, but avoids label bias by temporally linking adjacent tags in a new way. Specifically, we employ virtual evidence in the form of a binary observed child node, c_i, between adjacent tags (Figure 3) or a windowed sequence of tags. During decoding, this node will always be observed to be equal to 1 (one). Intuitively, this binary variable acts as an indicator of

[1] In this HMM, Word and isCap are independent given Tag, but this need not be true in general.

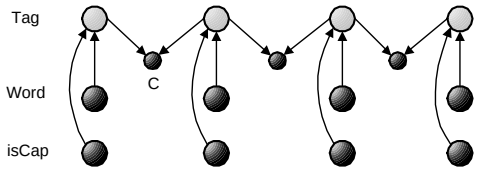

Figure 3: First order observed-child model (OCM) with the tags connected in pairs.

tag-pair consistency. When the tag pairs are consistent (as they are in real text), we should have a high conditional probability that $c_i = 1$; and when the tag pairs are not consistent, the conditional probability that $c_i = 1$ should be low. With this conditional distribution, observing $c_i = 1$ during decoding expresses a preference for consistent tag pairs.

The presence of this observed-child node results in a term in the factorization of the joint probability distribution that couples its parents:

$$P(\mathbf{c}, \mathbf{s}, \mathbf{o}) \propto \prod_i P(c_i|s_{i-1}, s_i)P(s_i|w_i, f_i)$$

where c_i is the observed-child node of tags s_{i-1} and s_i, and we omit the probability of the observations, $P(w_i, f_i)$ which do not affect the final choice of **s**.

By the rules of d-separation (Pearl, 1988), the existence of c_i defined in this way means that the parents (the adjacent tags) are *not* conditionally independent given the child. This link between adjacent tags through an observed-child node allows for a probabilistic relationship to exist between the adjacent tags. Thus, future words can influence tags, which is not true for the CMM. Whether or not a relationship between tags will actually be learned, however, will critically depend on how the model is trained. In a graphical model, it is the *lack* of an edge that ensures some form of independence; the presence of an edge (or a path made up of two or more edges) does not necessarily ensure the reverse.

3.1 Training

The introduction of virtual evidence into a graphical model requires that careful thought be given to the training process. If we were to naïvely add $c_i = 1$ to all samples of the training data, the model would learn that c_i is constant rather than random, and therefore that it is independent of its parents, s_{i-1} and s_i. In other words, this naïvely-trained model would assume that $P(c_i = 1|s_{i-1}, s_i) = 1 \ \forall \ (s_{i-1}, s_i)$, and when used to tag the sentences in the test-set (also labeled with $c_i = 1$), it would maximize this simplified joint probability in which the relationship between s_{i-1} and s_i has been lost:

$$P(\mathbf{c}, \mathbf{s}, \mathbf{o}) \propto \prod_i P(s_i|w_i, f_i)$$

In order to induce and thereby have the model *learn* the relationship between the adjacent tags s_{i-1} and s_i, the training has to be modified to include samples that are labeled with $c_i = 0$. The probability table $P(c_i = 1|s_{i-1}, s_i)$ should favor common (consistent) tag-pairs with high probabilities, while discouraging rare tag-pairs with low probabilities.

Although all observations (in both training and test sets) are labeled with $c_i = 1$, we hypothesize an alternate set of observations labeled with $c_i = 0$. This alternate set will be the source of the negative training data [2]. It is a set of nonsensical sentences with the same distribution over individual tags, i.e. the same $P(s_i)$, but in this set adjacent tags are independent. We denote the total number of training samples by M. This is divided into positive training samples, M_1, and negative training samples, M_0, with $M_1 + M_0 = M$. The ratio of the amount of positive to negative training data should be the same as the ratio of our prior beliefs about tag-pair consistency, namely the ratio of $P(c_i = 1)$ to $P(c_i = 0)$. With no evidence to support that one is more likely than the other, one option is to use the strategy of "assuming the least" and use a maximum entropy prior, setting $M_0 = M_1$. More flexibly, we can define n to be the ratio of the two so that $M_0 = n \cdot M_1$.

Now we derive a method for training the conditional probability table $P(c_i|s_{i-1}, s_i)$ in terms of the pointwise mutual information between the adjacent tags s_{i-1} and s_i. We first rewrite the conditional probability (henceforth abbreviated as p) as:

$$p = P(c_i = 1|s_{i-1}, s_i) = \frac{P(c_i = 1, s_{i-1}, s_i)}{P(s_{i-1}, s_i)}$$

If the probabilities are maximum likelihood (ML) estimates derived from counts on the training data, we can equivalently write:

$$p = \frac{\mathbf{N}(c_i = 1, s_{i-1}, s_i)}{\mathbf{N}(s_{i-1}, s_i)}$$

[2] This use of implied negative training data is similar to the "neighborhood" concept described in (Smith and Eisner, 2005)

where $\mathbf{N}(\cdot)$ is the count function.

Expanding the denominator into two terms:

$$p = \frac{\mathbf{N}(c_i = 1, s_{i-1}, s_i)}{\mathbf{N}(c_i = 1, s_{i-1}, s_i) + \mathbf{N}(c_i = 0, s_{i-1}, s_i)}$$

Without any negative training data (labeled with $c_i = 0$), this ratio would always evaluate to 1, and no probabilistic relationship between s_{i-1} and s_i would be learned.

From the start, we have implicitly postulated a relationship between adjacent tags. We now formally state two hypotheses: H_1 that there *is* a relationship between adjacent tags which can be described by some joint probability distribution $P(s_{i-1}, s_i)$, and the null hypothesis, H_0, that there is no such relationship, i.e. s_{i-1} and s_i are independent:

$$P_{H_1} = P(s_{i-1}, s_i)$$

$$P_{H_0} = P(s_{i-1})P(s_i)$$

Now we can express the counts as follows:

$$\mathbf{N}(c_i = 1, s_{i-1}, s_i) = M_1 \cdot P(s_{i-1}, s_i)$$

$$\mathbf{N}(c_i = 0, s_{i-1}, s_i) = M_0 \cdot P(s_{i-1})P(s_i)$$

where M_1 is the total number of tokens in the (positive) training data, and M_0 is the total number of tokens in the induced negative training data. We substitute M_0 with $n \cdot M_1$ for the reasons mentioned earlier, and simplify to obtain:

$$p = \frac{P(s_{i-1}, s_i)}{P(s_{i-1}, s_i) + nP(s_{i-1})P(s_i)}$$

which can be simplified to obtain:

$$p = \frac{1}{1 + n\left[\frac{P(s_{i-1}, s_i)}{P(s_{i-1})P(s_i)}\right]^{-1}}$$

The ratio of probabilities in the denominator is the ratio used in computing the pointwise mutual information between s_{i-1} and s_i. This ratio, which we will call λ, is also the likelihood ratio between the two previously stated hypotheses. Finally, we write the conditional probability as a function of λ:

$$P(c_i = 1|s_{i-1}, s_i) = \frac{1}{1 + n\lambda^{-1}} = \frac{\lambda}{\lambda + n}$$

$$\text{where} \quad \lambda = \frac{P_{H_1}}{P_{H_0}} = \frac{P(s_{i-1}, s_i)}{P(s_{i-1})P(s_i)} = \frac{P(s_i|s_{i-1})}{P(s_i)}$$

The conditional probability, $P(c_i = 1|s_{i-1}, s_i)$ is a mapping $g(\lambda)$ from $\lambda \in [0, \infty)$ to $p \in [0, 1)$.

Beginning with (Church and Hanks, 1989), numerous authors have used the pointwise mutual information between pairs of words to analyze word co-locations and associations. This ratio tells us whether s_{i-1} and s_i co-occur more or less often than would be expected by chance alone.

Consider, for example, the tags DT (determiner) and NN (noun), and the four possible ordered tag-pairs. The probabilities $P(s_i)$ and $P(s_i|s_{i-1})$ derived from the training data (see section 4.1), the likelihood ratio score λ, the conditional probability $p = P(c_i = 1|s_{i-1}, s_i)$, and the occurrence counts **N** are shown in Table 1. As expected, the sequence DT-NN (e.g. *the surplus*) occurs very often and gets a high score, while DT-DT (e.g. *this a*) and NN-DT (e.g. *surplus the*) occur infrequently and get low scores. The sequence NN-NN (e.g. *trade surplus*) gets a neutral score ($\lambda \approx 1$) indicating that if the preceding word is a noun, the likelihood that the current word is a noun is nearly equal to the likelihood that any randomly chosen word is a noun.

We present two methods for inducing the negative training counts that are required to train the conditional probability table for $P(c_i|s_{i-1}, s_i)$.

In the first method, we generate "nonsense" sentences by randomly scrambling each sentence in the training-set n times, using a uniform distribution over all possible permutations. This results in n negative training sentences for each positive training sentence and therefore $M_0 = n \cdot M_1$. Effectively, the ratio of priors on c_i is now:

$$\frac{P(c_i = 1)}{P(c_i = 0)} = \frac{M_1}{M_0} = \frac{1}{n}$$

The conditional probability table $P(c_i|s_{i-1}, s_i)$ is

s_{i-1}-s_i	$P(s_i)$	$P(s_i\|s_{i-1})$	λ	p	**N**
DT-NN	0.129	0.4905	3.80	0.79	37301
NN-NN	0.129	0.1270	0.98	0.49	15571
NN-DT	0.080	0.0071	0.09	0.08	870
DT-DT	0.080	0.0018	0.02	0.02	134

Table 1: Sample likelihood ratio scores (λ), probabilities, p (for $n = 1$), and counts for four tag-pairs.

then trained using all $n+1$ versions of each sentence, thus inducing the desired dependence between s_{i-1} and s_i. The method of scrambling sentences n-times only approximates the theory described above because it is performed on a sentence-by-sentence basis rather than across the entire training set. Also, the resulting negative training data represents only n realizations of a random process, so the total number of samples may not be large enough to approximate the underlying distribution.

In the second method, rather than generate the negative training data in the form of scrambled sentences, we compute the negative-training counts directly, based on the positive unigram counts and the hypotheses presented in section 3.1. For example, the negative bigram counts are a function of the marginal probability of each tag, $P(s_i)$:

$$\mathbf{N}(c_i = 0, s_{i-1}, s_i) = nM_1 \cdot P(s_{i-1})P(s_i)$$

Negative unigram and trigram counts are computed in a similar fashion, and then the conditional probability table is derived as a smoothed back-off model directly from the combined sets of counts.

These two methods are conceptually similar but may exhibit subtle differences: one is randomizing at the sentence level while the other operates over the entire training set and does not have the same sensitivity to small values of n.

4 Experiments and Results

In this section we describe our experiments and the results obtained. Sections 4.1 and 4.2 describe the data sets and features. Section 4.3 presents comparisons between several simple models using just the tags, the words, and a single binary feature for each word. Section 4.4 presents results from a feature-rich second-order observed-child model in which tags are linked in groups of three.

All training of language models is done using the SRILM toolkit (Stolcke, 2002) with the FLM extensions (Bilmes and Kirchhoff, 2003), and the implementation and testing of the various graphical models is carried out with the help of the graphical models toolkit (GMTK) (Bilmes and Zweig, 2002).

4.1 Data Sets

The data used for these experiments is the Wall Street Journal data from Penn Treebank III (Marcus et al., 1994). We extracted tagged sentences from the parse trees and divided the data into training (sections 0-18), development (sections 19-21), and test (sections 22-24) sets as in (Toutanova et al., 2003). Except for the final results for the feature-rich model, all results are on the development set.

4.2 Features

The tagged sentences extracted from the Penn Treebank are pre-processed to generate appropriately-formatted training data for the SRILM toolkit, as well as the vocabulary and observation files to be used during testing.

The pre-processing includes building a dictionary based on the training data. All words containing uppercase letters are converted to lowercase before being written to the dictionary. Words that occur rarely are excluded from the dictionary and are instead mapped to a single out-of-vocabulary word. This is based on the idea from (Ratnaparkhi, 1996) that rare words in the training set are similar to unknown words in the test set, and can be used to learn how to tag the unknown words that will be encountered during testing. In this work, rare words are those that occurr fewer than 5 times. The dictionary also includes special *begin-sentence* and *end-sentence* words, as well as punctuation marks, resulting in a total of 10,824 words. A list of the 45 tags found in the training data is also created, and is similarly augmented with special *begin-sentence* and *end-sentence* tags, for a total of 47 distinct tags.

Each word has associated with it a set of features. During training, these features are used to learn a smoothed back-off model for $P(s_i|w_i, \mathbf{f}_i)$ (where $\mathbf{f}_i$ is a vector of features associated with word w_i).

The following five binary flags, taken from (Toutanova et al., 2003), are derived from the current word w_i and used as features :

- is-capitalized (refers to the first letter only);
- has-digits (word contains one or more digits);
- is-hyphenated (word contains '-');
- is-all-caps (*all* letters are capitalized);
- is-conjunction (true if is-all-caps, has-digits, and is-hyphenated are all true, for example *CFC-12* or *F/A-18*).

Prefixes and suffixes are also known to be informative and so we add a prefix-feature and a suffix-

feature to our set. Previous work used all possible prefixes and suffixes ranging in length from 1 to k characters, with $k = 4$ (Ratnaparkhi, 1996), and $k = 10$ (Toutanova et al., 2003). This method results in very long lists of thousands of suffixes and prefixes. In this work, we instead analyzed the rare words in the training data to generate shorter lists of informative prefixes and suffixes, with lengths between 1 and 7 characters. Each prefix/suffix was scored based on the number of times it appeared with a particular tag, and all prefixes/suffixes that scored above 20 (an arbitrarily chosen threshold) were kept. This process resulted in two separate lists: one with 377 prefixes, and the other with 704 suffixes. Each word is then assigned a single prefix feature and a single suffix feature from these lists (which both include an entry for "unknown"). When assigning prefix and suffix features to the rare words (in the training data) or the unknown words (in the test data), we assume that the longest string is the most informative. (This may not necessarily be true: for example, although the suffix *ing* is certainly more informative than *g*, it is less clear whether *ulating* would be more or less informative than *ing*.)

We also include the two adjacent words as features of the current word. Our model provides great flexibility in the choice of features to be included in the current word's feature-set. This feature-set is not limited to binary flags and indeed can include anything that can be extracted from the observation sequence in the pre-processing stage. By using a smoothed back-off model, issues related to data-sparsity and over-fitting are avoided.

4.3 First Order Model Comparisons

In this section we compare results obtained from three first-order models: HMM, CMM, and OCM, using a Naïve Bayes (NB) model as a baseline. The Naïve Bayes model is a zeroth-order model with no connection between adjacent tags, while the first-order models connect adjacent tags in pairs. (Note that the HMM in this case is just a "temporal" NB since given the tag, the features are independent.) In these experiments, the only feature used is the is-capitalized flag (the most informative of the binary flags tested). The results are shown in Table 2.

The conditional probability tables (CPTs) for the CMM and the OCM were generated using the

model type	token accur.	known-w. accur.	unk.-w. accur.
Naïve Bayes	90.56%	93.83%	43.4%
$\text{OCM}_{n=0}$	90.89%	94.07%	45.2%
CMM	93.23%	95.69%	57.9%
$\text{OCM}_{n=1}$	93.94%	96.39%	58.6%
HMM	94.30%	96.53%	62.3%
$\text{OCM}_{n=4}$	94.42%	96.63%	62.7%

Table 2: Scores for first order models.

factored language model (FLM) extensions to the SRILM toolkit, wth generalized parallel backoff and Witten-Bell smoothing. (Modified Kneser-Ney smoothing could not be applied because some of the required low-order meta-counts needed by the discount estimator were zero.) The negative training data for the OCM was generated using the scramble method, with values of n as in the table. When no negative training data is used ($n = 0$), the CPT for the observed-child shows a very weak dependence on the specific tag-pair (s_{i-1}, s_i): the probability values in the tag-bigram model range only between 0.89 and 1. This weak dependence results in performance comparable to that of the Naïve Bayes model. That there is any dependence at all is due to the smoothing since $c_i = 0$ is never observed in the training data. With negative training data ($n = 4$), there is a much stronger dependence on the tag-pair, and the values for $P(c_i = 1|s_{i-1}, s_i)$ range between 0.0002 and 1.

We found experimentally that the OCM reached peak performance with $n = 4$ and that for larger n the performance stayed relatively constant: the variation for values of n up to 14 was only 0.05%.

4.4 Feature-Rich Second-Order OCM

In this section we describe the results obtained from a more complex second order OCM with the additional word features described in section 4.2.

This model is illustrated in Figure 4 which, for clarity highlights the details only for one (tag,word) pair. The observed-child node, c_i, now has three parents: the tags s_{i-1}, s_i, and s_{i+1}. Each tag, s_i, in turn has $K + 1$ parents: the current word, w_i, and a set of K features (shown bundled together). The model switches between the two feature bundles as

model description	token accuracy	known-word accuracy	unknown-word accuracy
OCM-I, scramble, $n = 4$	96.39%	96.87%	89.5%
OCM-I, computed counts, $n = 4$	96.41%	96.90%	89.3%
OCM-I, computed counts, $n = 1$	96.41%	96.92%	89.0%
OCM-II, computed counts, $n = 1$	96.64%	97.12%	89.5%
OCM-II, as above, on test-set	96.77%	97.25%	90.0%

Table 3: Tagging accuracy using the feature-rich 2^{nd} order observed-child model.

illustrated, based on the current word. For known words, a small set of features is used, while a much larger set of features is used for unknown words. This switching increases the speed of the model at no cost: the additional features increase the tagging accuracy for unknown words but are redundant for known words.

This model factorizes the joint probability as:

$$P(\mathbf{c}, \mathbf{s}, \mathbf{o}) \propto \prod_i P(c_i|s_{i-1}, s_i, s_{i+1}) P(s_i|w_i, \mathbf{f}_i)$$

where $\mathbf{f}_i$ is the appropriate feature bundle for word w_i, depending on whether w_i is known or unknown.

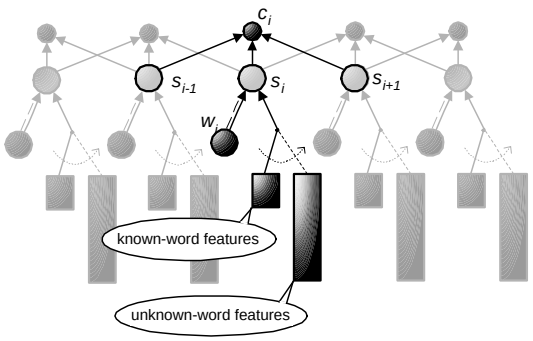

Figure 4: Second order OCM with tags connected in triples and switching sets of word features.

Two sets of experiments were performed using two models, which we will refer to as OCM-I and OCM-II. Both of these are second order models (connecting tags in triples), but with different sets of features. In model OCM-I, the only feature used for known words is the is-capitalized flag used in section 4.3. The unknown words use a total of seven features: suffix, prefix, and all five of the binary flags described in section 4.2. Model OCM-II adds the adjacent words (w_{i-1} and w_{i+1}) to the feature-set for both known and unknown words.

As seen above, the model factorizes the joint probability into two conditional probability terms. Each of these CPTs is implemented as a smoothed, factored-language back-off model.

The observed-child CPT uses generalized backoff, combining at run-time the results of backing off from each of the three parents if the specific tag-triple is not found in the table. The tag CPT uses linear backoff, dropping the adjacent words first. The backoff order for the other features was chosen based on experiments to determine the relative information content of each feature. This resulted in the following backoff order: prefix, has-digit, is-conjunction, is-all-caps, is-hyphenated, suffix, is-capitalized, word (where the least informative feature, prefix, is the first feature to be dropped).

Results from these experiments are shown in Table 3. Except for the last line, which reports results on the test set, all results are on the development set. The first three lines show results obtained from OCM-I (without adjacent word features). The two methods of generating negative training data yield nearly identical results, showing that they are comparable. Comparing rows 2 and 3 in the table we see that the computed-counts method is relatively insensitive to the value of n (for $n \geq 1$).

OCM-II, which uses the adjacent words as features for both known and unknown words further improves overall accuracy, and produces state-of-the-art results. The token-level accuracy result obtained from the OCM-II model on the development set (96.64%) can be directly compared to an accuracy of 96.57% reported in (Toutanova et al., 2003) for a cyclic dependency network using similar word features and the same three tag context.

5 Conclusions

In this paper, we have introduced two new concepts to the problem of part-of-speech tagging: virtual evidence and negative training data. We have moreover shown that this new model can produce state-of-the-art results on this NLP task with appropriately chosen features. The model stays entirely within the mathematically formal language of Bayesian networks, and even though it is conditional in nature, the model does not suffer from label or observation (or directional) bias. Staying within this framework has other advantages as well, including that the training procedures remain within the relatively simple maximum likelihood framework, albeit with appropriate smoothing. We believe that this model holds great promise for other NLP tasks as well as in other applications of machine-learning such as computational biology. In particular the way it factorizes the joint probability into a "horizontal" component which connects various nodes to the virtual-evidence node, and a "vertical" component (used here to link a tag to a set of observations), provides great simplicity, flexibility, and power.

6 Acknowledgements

The authors would like to thank the anonymous reviewers for their constructive comments. Sheila Reynolds is supported by an NDSEG fellowship.

References

Michele Banko and Robert C. Moore. 2004. Part of Speech Tagging in Context. *Proceedings of COLING*.

Jeff Bilmes. 2004. On Soft Evidence in Bayesian Networks. Tech. Rep. UWEETR-2004-0016, U. Washington Dept. of Electrical Engineering, 2004.

Jeff Bilmes and Katrin Kirchhoff. 2003. Factored language models and generalized parallel backoff. *Proceedings of HLT-NAACL: Short Papers*, 4-6.

Jeff Bilmes and Geoffrey Zweig. 2002. The graphical models toolkit: An open source software system for speech and time-series processing. *Proceedings of ICASSP*, vol4, 3916-3919.

Kenneth W. Church and Patrick Hanks. 1989. Word Association Norms, Mutual Information, and Lexicography. *Proceedings of ACL*, 76-83.

Michael Collins. 2002. Discriminative Training Methods for Hidden Markov Models: Theory and Experiments with Perceptron Algorithms. *Proc. EMNLP*.

Dan Klein and Christopher D. Manning. 2002. Conditional Structure versus Conditional Estimation in NLP Models. *Proceedings of EMNLP*, 9-16.

John Lafferty, Andrew McCallum and Fernando Pereira. 2001. Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data. *Proceedings of ICML*, 282-289.

Sang-Zoo Lee, Jun-ichi Tsujii and Hae-Chang Rim. 2000. Part-of-Speech Tagging Based on Hidden Markov Model Assuming Joint Independence. *Proceedings of 38th ACL*, 263-269.

Mitchell P. Marcus, Beatrice Santorini and Mary A. Marcinkiewicz. 1994. Building a large annotated corpus of English: The Penn Treebank. *Computational Linguistics*, 19:313-330.

Andrew McCallum. 2003. Effi ciently Inducing Features of Conditional Random Fields. *Proceedings of UAI*.

Andrew McCallum, Dayne Freitag and Fernando Pereira. 2000. Maximum-Entropy Markov Models for Information Extraction and Segmentation. *Proc. 17th International Conf. on Machine Learning*, 591-598.

Judea Pearl. 1988. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann.

Adwait Ratnaparkhi. 1996. A maximum entropy model for part-of-speech tagging. *EMNLP 1*, 133-142.

Noah A. Smith and Jason Eisner 2005. Contrastive Estimation: Training Log-Linear Models on Unlabeled Data. *Proceedings of ACL*.

Andreas Stolcke. 2002. SRILM – an extensible language modeling toolkit. *Proc. ICASSP*, vol 2, 901-904.

Scott M. Thede and Mary P. Harper. 1999. A Second-Order Hidden Markov Model for Part-of-Speech Tagging. *Proceedings of 37th ACL*, 175-182.

Kristina Toutanova, Dan Klein, Christopher D. Manning, and Yoram Singer. 2003. Feature-Rich Part-of-Speech Tagging with a Cyclic Dependency Network. *Proceedings of HLT-NAACL*, 252-259.