Mutual Information Independence Model Using Kernel Density Estimation for Segmenting and Labeling Sequential Data

Guodong Zhou, Lingpeng Yang, Jian Su, and Donghong Ji

Institute for Infocomm Research, 21 Heng Mui Keng Terrace, Singapore 119613
{zhougd, lpyang, sujian, dhji}@i2r.a-star.edu.sg

Abstract. This paper proposes a Mutual Information Independence Model (MIIM) to segment and label sequential data. MIIM overcomes the strong context independent assumption in traditional generative HMMs by assuming a novel pairwise mutual information independence. As a result, MIIM separately models the long state dependence in its state transition model in a generative way and the observation dependence in its output model in a discriminative way. In addition, a variable-length pairwise mutual information-based modeling approach and a kNN algorithm using kernel density estimation are proposed to capture the long state dependence and the observation dependence respectively. The evaluation on shallow parsing shows that MIIM can effectively capture the long context dependence to segment and label sequential data. It is interesting to note that using kernel density estimation leads to increased performance over using a classifier-based approach.

1 Introduction

A Hidden Markov Model (HMM) is a model where a sequence of observations is generated in addition to the Markov state sequence. It is a latent variable model in the sense that only the observation sequence is known while the state sequence remains "hidden". In recent years, HMMs have enjoyed great success in many tagging applications, most notably part-of-speech (POS) tagging [1,2,3] and named entity recognition [4,5]. Moreover, there have been also efforts to extend the use of HMMs to word sense disambiguation [6] and shallow/full parsing [7,8,9].

Given an observation sequence $O_1^n = o_1 o_2 \cdots o_n$, the goal of a HMM is to find a stochastic optimal state sequence $S_1^n = s_1 s_2 \cdots s_n$ that maximizes $P(S_1^n, O_1^n)$:

$$S^* = \arg\max_{S_1^n} \log P(S_1^n, O_1^n) = \arg\max_{S_1^n} \{\log P(S_1^n) + \log P(O_1^n \mid S_1^n) \quad (1)$$

Traditionally, HMM segments and labels sequential data in a generative way by making a context independent assumption that successive observations are independent given the corresponding individual state [10]:

A. Gelbukh (Ed.): CICLing 2005, LNCS 3406, pp. 155 – 166, 2005.

$$P(O_1^n \mid S_1^n) = \prod_{i=1}^{n} P(o_i \mid s_i) \tag{2}$$

By applying the assumption (2) and using the chain rule, equation (1) can be rewritten as:

$$\begin{aligned} S^* &= \arg\max_{S_1^n} \{\log P(S_1^n) + \sum_{i=1}^{n} \log P(o_i \mid s_i)\} \\ &= \arg\max_{S_1^n} \{\sum_{i=2}^{n} \log P(s_i \mid S_1^{i-1}) + \log P(s_1) + \sum_{i=1}^{n} \log P(o_i \mid s_i)\} \end{aligned} \tag{3}$$

More formally, a generative (first-order) HMM is given by a finite set of states S including an designated initial state and an designated final state, a set of possible observation O, two conditional probability distributions: a state transition model $P(s \mid s')$ from s' to s for $s', s \in S$ and an output model $P(o \mid s)$ for $o \in O, s \in S$. A sequence of observations is generated by starting from the designated initial state, transmiting to a new state according to $P(s \mid s')$, emitting an observation selected by that new state according to $P(o \mid s)$, transmiting to another new state and so on until the designated final state is generated.

There are several problems with this generative approach. First, many tasks would benefit from a richer representation of observations—in particular a representation that describes observations in terms of many overlapping features, such as capitalization, word endings, part-of-speech in addition to the traditional word identity. Note that these features always depends on each other. Furthermore, to define a joint probability over the observation and state sequences, the generative approach needs to enumerate all the possible observation sequences. However, in some tasks, the set of all the possible observation sequences is not reasonably enumerable. Second, the generative approach fails to effectively model the dependence in the observation sequence. Third, the generative approach normally estimates the parameters to maximize the likelihood of the observation sequence. However, in many NLP tasks, the goal is to predict the state sequence given the observation sequence. In other words, the generative approach inappropriately applies a generative joint probability model for a conditional probability problem. In summary, the main reasons behind these problems of the generative approach are the strong context independent assumption and the generative nature in modeling sequential data. While the dependence between successive states can be directly modeled by its state transition model, the generative approach fails to directly capture the observation dependence in the output model.

To resolve the inherent problems in generative HMMs, some researches (please see related works in Section 6 for details) have been done to move from generative HMMs to discriminative Markov models (DMMs). DMMs do not expend modeling effort on the observation sequnce, which are fixed at test time. Instead, DMMs model the state sequence depending on arbitrary, non-independent features of the observation sequence, normally without forcing the model to account for the distribution of those dependencies.

This paper proposes a Mutual Information Independence Model (MIIM), which separates the dependence of a state on the previous states and the observation sequence. Compared with generative HMMs, MIIM explicitly models the long state dependence in a generative way and the observation dependence in a discriminative way. In addition, a variable-length pairwise mutual information based modeling is proposed to capture the long state dependence of a state on the previous states while a kNN algorithm using kernel density estimation is proposed to capture the observation dependence of a state on the observation sequence.

The layout of this paper is as follows. Section 2 proposes the Mutual Information Independence Model (MIIM) and presents the variable-length pair-wise mutual information-based modeling approach to capture the long state dependence. Section 3 presents the kNN algorithm using kernel density estimation to capture the observation dependence. Section 4 introduces the shallow parsing task while Section 5 gives experimental results. Section 6 describes some of the related works in discriminative Markov modeling. Finally, some conclusion will be drawn in Section 7.

2 Mutual Information Independence Model

In principle, given an observation sequence $o_1^n = o_1 o_2 \cdots o_n$, the goal of a conditional probability model is to find a stochastic optimal state sequence $s_1^n = s_1 s_2 \cdots s_n$ that maximizes $\log P(s_1^n \mid o_1^n)$

$$s^* = \arg\max_{s_1^n} \{\log P(s_1^n \mid o_1^n)\} = \arg\max_{s_1^n} \{\log P(s_1^n) + \log \frac{P(s_1^n, o_1^n)}{P(s_1^n) \cdot P(o_1^n)}\} \quad (4)$$

Obviously, the second term $\log \frac{P(s_1^n, o_1^n)}{P(s_1^n) \cdot P(o_1^n)}$ captures the pairwise mutual information (PMI) between the state sequence s_1^n and the observation sequence o_1^n. Here, we define $PMI(s_1^n, o_1^n) = \log \frac{P(s_1^n, o_1^n)}{P(s_1^n) \cdot P(o_1^n)}$. To compute $PMI(s_1^n, o_1^n)$ efficiently, we propose a novel pairwise mutual information independence assumption:

$$PMI(s_1^n, o_1^n) = \sum_{i=1}^{n} PMI(s_i, o_1^n) \text{ or } \log \frac{P(s_1^n, o_1^n)}{P(s_1^n) \cdot P(o_1^n)} = \sum_{i=1}^{n} \log \frac{P(s_i, o_1^n)}{P(s_i) \cdot P(o_1^n)} \quad (5)$$

That is, we assume a state is only dependent on the observation sequence o_1^n and independent on other states in the state sequence s_1^n. This assumption is reasonable because the dependence among the states in the state sequence s_1^n has been directly captured by the first term $\log P(s_1^n)$ in equation (4).

By applying the assumption (5) into the equation (4), we have:

$$
\begin{aligned}
s^* &= \arg\max_{s_1^n}\{\log P(s_1^n) + \sum_{i=1}^{n} \log \frac{P(s_i, o_1^n)}{P(s_i)\cdot P(o_1^n)} \\
&= \arg\max_{s_1^n}\{\log P(s_1^n) - \sum_{i=1}^{n} \log P(s_i) + \sum_{i=1}^{n} \log \frac{P(s_i, o_1^n)}{P(o_1^n)}\} \\
&= \arg\max_{s_1^n}\{\sum_{i=2}^{n} \log P(s_i \mid s_1^{i-1}) + \log P(s_1) - \sum_{i=1}^{n} \log P(s_i) + \sum_{i=1}^{n} \log P(s_i \mid o_1^n)\} \qquad (6) \\
&= \arg\max_{s_1^n}\{\sum_{i=2}^{n} \log P(s_i \mid s_1^{i-1}) - \sum_{i=2}^{n} \log P(s_i) + \sum_{i=1}^{n} \log P(s_i \mid o_1^n)\} \\
&= \arg\max_{s_1^n}\{\sum_{i=2}^{n} PMI(s_i, s_1^{i-1}) + \sum_{i=1}^{n} \log P(s_i \mid o_1^n)\}
\end{aligned}
$$

The above model consists of two models: the state transition model $\sum_{i=2}^{n} PMI(s_i, s_1^{i-1})$ which measures the state dependence of a state given the previous states in a generative way, and the output model $\sum_{i=1}^{n} \log P(s_i \mid o_1^n)$ which measures the observation dependence of a state given the observation sequence in a discriminative way. This is done by assuming a novel pair-wise mutual information independence model. Therefore, we call the above model as in equation (6) a Mutual Information Independence Model (MIIM). The main difference between a generative HMM and a MIIM lies in their output models in that the output model of a MIIM directly captures the context dependence between successive observations in determining the "hidden" states while the output model of the generative HMM fails to do so. That is, the output model of a MIIM overcomes the strong context independent assumption in the generative HMM and becomes observation context dependent. Alternatively, we can have equation (7) by rewriting equation (3) using the Bayes's rule:

$$
\begin{aligned}
S^* &== \arg\max_{S_1^n}\{\sum_{i=2}^{n} \log P(s_i \mid S_1^{i-1}) + \log P(s_1) + \sum_{i=1}^{n} \log P(o_i \mid s_i)\} \\
&= \arg\max_{S_1^n}\{\sum_{i=2}^{n} \log P(s_i \mid S_1^{i-1}) + \log P(s_1) \\
&+ \sum_{i=1}^{n} \{\log P(s_i \mid o_i) + \log P(o_i) - \log P(s_i)\} \\
&= \arg\max_{S_1^n}\{\sum_{i=2}^{n} \log P(s_i \mid S_1^{i-1}) + \log P(s_1) + \sum_{i=1}^{n} \{\log P(s_i \mid o_i) - \log P(s_i)\} \qquad (7) \\
&= \arg\max_{S_1^n}\{\sum_{i=2}^{n} \{\log P(s_i \mid S_1^{i-1}) - \log P(s_i)\} + \sum_{i=1}^{n} \log P(s_i \mid o_i)\} \\
&= \arg\max_{S_1^n}\{\sum_{i=2}^{n} PMI(s_i, S_1^{i-1}) + \sum_{i=1}^{n} \log P(s_i \mid o_i)\}
\end{aligned}
$$

Compared with MIIM as equation (6) and the generative HMM rewritten as equation (7), we can see that MIIM extends the notion of an observation and estimates the output model based on the observation sequence rather than the corresponding individual observation.

Computation of a MIIM consists of two parts. The first is to compute the state transition model: $\sum_{i=2}^{n} PMI(s_i, s_1^{i-1})$. Traditionally, ngram modeling(e.g. bigram for the first-order HMM and trigram for the second-order HMM) is used to estimate the state transition model. However, such approach fails to capture the long state dependence since it is not reasonably practical for ngram modeling to be beyond trigram. In this paper, a variable-length pairwise mutual information-based modeling approach is proposed as follow: For each $i(2 \le i \le n)$, we first find a minimal $k(0 \le k \prec i)$ where the frequency of s_k^i is not smaller than a threshold (e.g. 3) and then estimate $PMI(s_i, s_1^{i-1})$ using $PMI(s_i, s_1^{i-1}) = \log \frac{P(s_k^i)}{P(s_i)P(s_k^{i-1})}$. In this way, the long state dependence can be captured in a dynamical way. Here, the frequencies of variable-length state sequences are smoothed using the simple Good-Turing approach [11].

The second is to estimate the output model: $\sum_{i=1}^{n} \log P(s_i \mid o_1^n)$. Ideally, we would have sufficient training data for every event whose conditional probability we wish to calculate. Unfortunately, there is rarely enough training data to compute accurate probabilities when decoding on new data. Traditionally, there are two existing approaches to resolve this problem: linear interpolation [12] and back-off [13]. However, these two approaches only work well when the number of different information sources is limited. When a long context is considered, the number of different information sources is exponential and not reasonably enumerable. The current trend is to recast it as a classification problem and use the output of a classifier, e.g. the maximum entropy classifier (ME) [14] to estimate the state probability distribution given the observation sequence. In the next section, we will propose a more effective kNN algorithm using kernel density estimation to resolve this problem.

3 *kNN* Using Kernel Density Estimation

The main challenge for the above MIIM is how to reliably estimate $P(s_i \mid o_1^n)$ in its output model. For efficiency, we can always assume $P(s_i \mid o_1^n) \approx P(s_i \mid E_i)$, where the pattern entry $E_i = o_{i-N} \cdots o_i \cdots o_{i+N}$. That is, we only consider the observation dependence in a window of 2N+1 observations (e.g. we only consider the current observation, the previous observation and the next observation when N=1). For convenience, we denote $P(\bullet \mid E_i)$ as the conditional state probability distribution of the states given E_i and $P(s_i \mid E_i)$ as the conditional state probability of s_i given E_i.

The kNN algorithm calculates $P(\bullet \mid E_i)$ by first finding the K nearest neighbors of frequently occurring pattern entries $kNN(E_i) = \{E_i^k \mid k = 1,2,..., K\}$ and then aggregating them to make a proper estimation of $P(\bullet \mid E_i)$ using kernel density estimation. Here, the conditional state probability distribution is estimated instead of the classification in a traditional kNN classifier.

3.1 Finding K Nearest Neighbors

To do so, all the frequently occurring pattern entries are extracted exhaustively from the training corpus and stored in a dictionary *FrequentEntryDictionary*. Here, the dictionary *FrequentEntryDictionary* is indexed using the tree-based indexing scheme as in TiMBL[1] [15]. In order to limit the dictionary size and keep efficiency, we constrain a valid set of pattern entry forms *ValidEntryForm* to consider only the most informative information sources. Obviously, *ValidEntryForm* defines the possible feature conjunctions. Generally, *ValidEntryForm* can be determined manually or automatically according to the applications. In Section 5, we will give an example.

Given a pattern entry E_i and the indexed dictionary *FrequentEntryDictionary*, the K nearest neighbors of the pattern entry E_i is found as follows:

- Extract all the compatible entries with E_i from the indexed dictionary
- Compute the similarity between E_i and each of the compatible entries using a kernel function
- Sort out the K nearest neighbors according to their similarities

Here, the kernel function $k(E_i^k, E_i)$ between E_i and E_i^k is determined by their shared feature conjunctions:

$$k(E_i^k, E_i) = \sum_{f^j \in E_i, f^j \in E_i^k} w^j \tag{8}$$

with the parameter vector $\boldsymbol{w}$. Here, w^j is the weight for the j-th possible feature conjunction $f^j \in ValidEntryForm$. Here, the parameter vector $\boldsymbol{w}$ is determined as follows: For each entry in *FrequentEntryDictionary*, we first find the 2*K nearest neighbors from the indexed dictionary using the above same algorithm (Here, all the w^j in $\boldsymbol{w}$ is initialized to 1 divided by $|ValidEntryForm|$, the number of possible feature conjunctions). For each possible feature conjunction, we calculate its weight (averaged over all the entries in *FrequentEntryDictionary*) as its non-occurrence frequency in the second K nearest neighbors divided its occurrence frequency in the first K neighbors. The intuition is that, if a feature conjunction occurs more in the first K nearest neighbors, and less in the second K nearest neighbors, such feature

[1] http://ilk.kub.nl/

conjunction contributes more. Finally, $\boldsymbol{w}$ is normalized ($\sum_{all \quad f^j \in ValidEntryForm} w^j = 1$). The above training algorithm is repeated until $\boldsymbol{w}$ becomes stable (e.g. the change in $|\boldsymbol{w}|$ is less than 1%).

3.2 Kernel Density Estimation

After the K nearest neighbors have been found, the conditional state probability distribution $P(\bullet \mid E_i)$ of the pattern entry E_i is calculated using kernel density estimation. That is, $P(\bullet \mid E_i)$ is estimated by the weighted average of its K nearest neighbors:

$$P(\bullet \mid E_i) = \frac{\sum_{k=1}^{K} k(E_i^k, E_i) \cdot F(E_i^k) \cdot P(\bullet \mid E_i^k)}{\sum_{k=1}^{K} k(E_i^k, E_i) \cdot F(E_i^k)} \tag{9}$$

where the kernel function $k(E_i^k, E_i)$ measures the similarity between the pattern entry E_i and its nearest neighbor E_i^k and $F(E_i^k)$ is the occurring frequency of E_i^k.

4 Shallow Parsing

In order to evaluate the MIIM, we have applied it in the application of shallow parsing.

For shallow parsing, we have $o_1 = p_i w_i$, where $w_1^n = w_1 w_2 \cdots w_n$ is the word sequence and $p_1^n = p_1 p_2 \cdots p_n$ is the part-of-speech (POS) sequence, while the states are represented as structural tags to bracket and differentiate various categories of phrases. The basic idea of using the structural tags to represent the states is similar to Skut et al [8] and Zhou et al [9]. Here, a structural tag consists of three parts:

- Boundary Category (BOUNDARY): it is a set of four values: "O"/"B"/"M"/"E", where "O" means that current word is a whOle phrase and "B"/"M"/"E" means that current word is at the Beginning/in the Middle/at the End of a phrase.
- Phrase Category (PHRASE): it is used to denote the category of the phrase.
- Part-of-Speech (POS): Because of the limited number of boundary and phrase categories, the POS is added into the structural tag to represent more accurate state transition model.

For example, given the following POS tagged sentence as the observation sequence:

He/PRP reckons/VBZ the/DT current/JJ account/NN deficit/NN will/MD narrow/VB to/TO only/RB $/$ 1.8/CD billion/CD in/IN September/NNP ./.

We can have a corresponding sequence of structural tags as the state sequence:

O_NP_PRP(He/PRP) O_VP _VBZ (reckons/VBZ) B_NP _DT (the/DT) M_NP _JJ (current/JJ) M_NP _NN (account/NN) E_NP _NN (deficit/NN) B_VP _MD (will/MD) E_VP _VB (narrow/VB) O_PP _TO (to/TO) B_QP _RB (only/RB) M_QP _$ ($/$) M_QP _CD (1.8/CD) E_QP _CD (billion/CD) O_PP _IN (in/IN) O_NP _NNP(September/NNP) O_O _. (./.)

and an equivalent phrase chunked sentence as the shallow parsing result:

[NP He/PRP] [VP reckons/VBZ] [NP the/DT current/JJ account/NN deficit/NN] [VP will/MD narrow/VB] [PP to/TO] [QP only/RB $/$ 1.8/CD billion/CD] [PP in/IN] [NP September/NNP] [O ./.]

5 Experimental Results

We have used the CoNLL'2000 standard chunking corpus in our experimentation. This corpus was first used in the CoNLL-2000 shared chunking task [16], which aims to annotate 10 base phrase classes (NP, VP, PP, ADJP, etc). This corpus consists of four sections (15-18) of the WSJ part of the Penn TreeBank [17] for the training data (211727 tokens) and one section (20) for the test data (47377 tokens)[2].

All the evaluations are measured using the F-measure. Here, the F-measure is the weighted harmonic mean of the precision (P) and the recall (R): $F = \frac{(\beta^2+1)RP}{\beta^2 R + P}$ with $\beta^2 = 1$ [18], where the precision (P) is the percentage of predicted phrase chunks that are actually correct and the recall (R) is the percentage of correct phrase chunks that are actually found.

In this paper, the valid set of pattern entry forms *ValidEntryForm* is defined to include those pattern entry forms within a window of 7 observations(including current, left 3 and right 3 observations) where for w_j to be included in a pattern entry, all or one of the overlapping features in each of $p_j, p_{j+1}..., p_i (j \le i)$ or $p_i, p_{i+1}..., p_j (i \le j)$ should be included in the same pattern entry while for p_j to be included in a pattern entry, all or one of the overlapping features in each of $p_{j+1}, p_{j+2}..., p_i (j \prec i)$ or $p_i, p_{i+1}..., p_{j-1} (i \prec j)$ should be included in the same pattern entry. For example of a window of 3:

$$ValidEntryForm = \{p_i, p_i w_i, p_{i-1} p_i, p_{i-1} p_i w_i, p_{i-1} w_i p_i, p_{i-1} w_{i-1} p_i w_i, p_i p_{i+1}, p_i w_i p_{i+1},$$
$$p_i p_{i+1} w_{i+1}, p_i w_i p_{i+1} w_{i+1}, p_{i-1} p_i p_{i+1}, p_{i-1} w_{i-1} p_i p_{i+1}, p_{i-1} p_i w_i p_{i+1}, p_{i-1} p_i p_{i+1} w_{i+1},$$
$$p_{i-1} w_{i-1} p_i w_i p_{i+1}, p_{i-1} w_{i-1} p_i p_{i+1} w_{i+1}, p_{i-1} p_i w_i p_{i+1} w_{i+1}, p_{i-1} w_{i-1} p_i w_i p_{i+1} w_{i+1}\}$$

Table 1 shows the effect of different number of nearest neighbors in the kNN algorithm and considered previous states in the variable-length pair-wise mutual information modeling approach of the MIIM on the CoNLL'2000 chunking corpus. It

[2] http://cnts.uia.ac.be/conll2000/chunking/

shows that finding 3 nearest neighbors in the kNN algorithm using kernel density estimation performs best. It also shows that further increasing the number of nearest neighbors does not increase or even decrease the performance. This may be due to introduction of noisy neighbors when the number of nearest neighbors increases. Moreover, Table 1 shows that the MIIM performs best when six previous states is considered in the variable-length pair-wise mutual information-based modeling approach and further considering more previous states does not increase the performance. This suggests that the state dependence exists well beyond traditional ngram modeling (e.g. bigram and trigram) to six previous states and the variable-length pair-wise mutual information-based modeling approach can capture the long state dependence. In the following experimentation, we will use the MIIM with 3 nearest neighbors used in the kNN algorithm and 6 previous states considered in the variable-length pair-wise mutual information modeling approach.

Table 1. Effect of different numbers of nearest neighbors in the kNN algorithm and previous states considered in the variable-length pair-wise mutual information modeling approach of the MIIM

Shallow Parsing		Number of nearest neighbors				
		1	2	3	4	5
Number of considered previous states	1	92.06	92.51	92.83	92.82	92.83
	2	92.55	93.02	93.35	93.36	93.30
	4	92.82	93.34	93.72	93.67	93.61
	6	93.01	93.63	**93.96**	93.91	93.88
	8	93.14	93.68	93.92	93.85	93.83

Table 2. Comparison of the MIIM with generative HMMs and the kNN algorithm using kernel density estimation with a classifier-based approach

Model		CoNLL'2000 chunking
HMM	First order	91.84
	Second order	92.01
MIIM	kNN	**93.96**
	MaxEnt	93.59
	SNoW	93.74

Table 2 compares the MIIM with generative HMMs. It also compares the kNN algorithm using kernel density estimation with a classifier-based approach, such as SNoW [19,20] and MaxEnt [14] in estimating the output model of the MIIM. Here, all the classifiers (SNoW and MaxEnt) use the same observation history as the kNN algorithm in the MIIM with a windows of 7 observations including current, left 3 and right 3 observations, and use the same feature conjunctions as defined in *ValidEntryForm*. It shows that the MIIM significantly outperforms generative HMMs due to the modeling of the observation dependence and allowing for non-independent, difficult to enumerate observation features. It also shows that the kNN algorithm using kernel density estimation outperforms these classifier-based

approaches. This may be because kernel density estimation can output better probability distribution than a classifier-based approach. This suggests that the kNN algorithm using kernel density estimation captures the dependence between the features of the observation sequence more effectively by forcing the model to account for the distribution of those dependencies.

Table 3. Comparison of the MIIM with the best-reported systems on shallow parsing

Model	CoNLL'2000 chunking
Zhang et al [23](ensemble)	94.13
LSD-DMM(individual)	**93.96**
Kudoh et al [21] (ensemble)	93.91
Zhou et al [9](individual)	92.12

Table 3 compares the MIIM with the best-reported systems on shallow parsing, where one best individual system (using a single classifier) and two ensemble systems (using an ensemble of classifiers) are included. It shows that our system based on an individual MIIM significantly outperforms other best-reported individual systems and gains comparable performance with the best-reported ensemble systems.

6 Related Work

To resolve the inherent problems in generative HMMs, some researches have been done to move from generative HMMs to discriminative Markov models (DMMs). Punyakanok and Roth [22] proposed a projection-based DMM (PDMM) which represents the probability of a state transition given not only the current observation but also past and future observations and used the SNoW classifier [19,20] to estimate it. McCallum et al [24] proposed the extact same model and used maximum entropy to estimate it in the application of information extraction. Lafferty et al [25] extanded ME-PDMM using conditional random fields by incorporating the factored state representation of the same model (that is, representing the probability of a state given the observation sequence and the previous state) to alleviate the label bias problem in PDMMs, which can be biased towards states with few successor states. Similar work can also be found in Bouttou [26]. McCallum et al [27] further extended Lafferty et al [25] using dynamic conditional random fields. Punyakanok and Roth [22] also proposed a non-projection-based DMM which separates the dependence of a state on the previous state and the observation sequence, by rewriting the generative HMM in a discriminative way and heuristically extending the notation of an observation to the observation sequence. Zhou et al [9] systematically derived the exact same model as in Punyakanok and Roth [22] and used back-off modeling with error driven learning to esimate the probability of a state given the observation sequence.

Compared with the above DMMs, the MIIM explicitly models the long state dependence using a variable length pair-wise mutual information-based modeling approach and the non-projection nature of the MIIM alleviates the label bias problem inherent in projection-based DMMs. Another difference is the use of the kernel density estimation in estimating the output model of the MIIM. It is interesting to

show that the kernel density estimation leads to increased performance over a classifier-based approach.

7 Conclusion

Hidden Markov Models (HMM) are a powerful probabilistic tool for modeling sequential data and have been applied with success to many text-related tasks, such as shallow parsing. In these cases, the observations are usually modified as multinomial distributions over a discrete dictionary and the HMM parameters are set to maximize the likelihood of the observations. This paper presents a Mutual Information Independence Model (MIIM) that allows observations to be represented as arbitrary overlapping features and defines the conditional probability of the state sequence given the observation sequence. It does so by assuming a novel pair-wise mutual information independence to separate the dependence of a state given the observation sequence and the previous states. Finally, the long state dependence and the observation dependence can be effectively captured by a variable-length pair-wise mutual information model and a kNN algorithm using kernel density estimation respectively. It is also interesting to note that kernel density estimation leads to increased performance over a classifier-based approach in estimating the output model of the MIIM.

In future work, we will explore the effect of an ensemble in MIIM and its application in other tasks, such as named entity recognition and full parsing.

References

1. Church K.W. (1998). A Stochastic Pars Program and Noun Phrase Parser for Unrestricted Text. *Proceedings of the Second Conference on Applied Natural Language Processing (ANLP'1998).* Austin, Texas.
2. Weischedel R., Meteer M., Schwartz R., Ramshaw L. & Palmucci J. (1993). Coping with Ambiguity and Unknown Words through Probabilistic Methods. *Computational Linguistics.* 19(2): 359-382.
3. Merialdo B. (1994). Tagging English Text with a Probabilistic Model. *Computational Linguistics.* 20(2): 155-171.
4. Bikel D.M., Schwartz R. & Weischedel R.M. (1999). An Algorithm that Learns What's in a Name. *Machine Learning* (Special Issue on NLP). 34(3): 211-231.
5. Zhou G.D. & Su J. (2002). Named Entity Recognition Using a HMM-based Chunk Tagger, *Proceedings of the Conference on Annual Meeting for Computational Linguistics (ACL'2002).* 473-480, Philadelphia.
6. Segond F., Schiller A., Grefenstette & Chanod F.P. (1997). An Experiment in Semantic Tagging using Hidden Markov Model Tagging. *Proceedings of the Joint ACL/EACL workshop on Automatic Information Extraction and Building of Lexical Semantic Resources.* pp. 78-81. Madrid, Spain.
7. Brants T., Skut W., & Krenn B. (1997). Tagging Grammatical Functions. *Proceedings of the Conference on Empirical Methods on Natural Language Processing (EMNLP'1997).* Brown Univ.
8. Skut W. & Brants T. (1998). Chunk Tagger – Statistical Recognition of Noun Phrases. *Proceedings of the ESSLLI'98 workshop on Automatic Acquisition of Syntax and Parsing.* Univ. of Saarbrucken. Germany.

9. Zhou G.D. & Su J. (2000). Error-driven HMM-based Chunk Tagger with Context-Dependent Lexicon. *Proceedings of the Joint Conference on Empirical Methods on Natural Language Processing and Very Large Corpus (EMNLP/ VLC'2000)*. Hong Kong.
10. Rabiner L.R. (1989). A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition. *Proceedings of the IEEE,* 77(2): 257-286.
11. Gale W.A. & Sampson G. 1995. Good-Turing frequency estimation without tears. *Journal of Quantitative Linguistics*. 2:217-237.
12. Jelinek F. (1989). Self-Organized Language Modeling for Speech Recognition. In Alex Waibel and Kai-Fu Lee (Editors). *Readings in Speech Recognitiopn*. Morgan Kaufmann. 450-506.
13. Katz S.M. (1987). Estimation of Probabilities from Sparse Data for the Language Model Component of a Speech Recognizer. *IEEE Transactions on Acoustics. Speech and Signal Processing*.35:400-401.
14. Ratnaparkhi A. (1999). Learning to parsing natural language with maximum entropy models. *Machine Learning*. 34:151-175.
15. Daelemans W. Buchholz S. & Veenstra. (1999). Memory-based shallow parsing. In *Proceedings of CoNLL'1999*. Bergen, Norway.
16. Tjong K.S. Daelemans, Dejean H. etc. (2000). Applying system combination to base noun noun phrase indentification. In *Proceedings of COLING 2000*. Saarbrucken Germany.
17. Marcus M., Santorini B. & Marcinkiewicz M.A. (1993). Buliding a large annotated corpus of English: The Penn Treebank. *Computational Linguistics*. 19(2):313-330.
18. van Rijsbergen C.J. (1979). *Information Retrieval*. Buttersworth, London.
19. Roth D. (1998). Learning to resolve natural language ambiguities: A unified approach. In *Proceedings of the National Conference on Artificial Intelligence.* 806-813.
20. Carlson A, Cumby C. Rosen J. & Roth D. 1999. The SNoW learning architecture. *Techinical Report* UIUCDCS-R-99-2101. UIUC.
21. Kudoh T. & Matsumoto Y. (2001). Chunking with support vector machines. In *Proceedings of NAACL'2001*. Pittsburgh, PA, USA.
22. Punyakanok V. & Roth D. (2000). The Use of Classifiers in Sequential Inference *NIPS-13.*
23. Zhang T., Damerau F. & Johnson D. (2002). Text chunking based on a generalization of winnow. Journal of Machine Learning Research. Vol. 2 (March). Pp 615-637).
24. McCallum A. Freitag D. & Pereira F. 2000. Maximum entropy Markov models for information extraction and segmentation. ICML-19. 591-598. Stanford, California.
25. Lafferty J. McCallum A & Pereira F. (2001). Conditional random fields: probabilistic models for segmenting and labeling sequence data. *ICML-20.*
26. Bottou L. (1991). Une approche theorique de l'apprentissage connexionniste: Applications a la reconnaissance de la parole. *Doctoral dissertation, Universite de Paris XI.*
27. McCallum A., Rohanimanesh K. & Sutton C. (2003). Dynamic conditional random fields for jointly labeling multiple sequences. In Proceedings of IJCAI 2003.