On the Use of Virtual Evidence in Conditional Random Fields

Xiao Li
Microsoft Research
One Microsoft Way
Redmond, WA 98052 USA
xiaol@microsoft.com

Abstract

Virtual evidence (VE), first introduced by (Pearl, 1988), provides a convenient way of incorporating prior knowledge into Bayesian networks. This work generalizes the use of VE to undirected graphical models and, in particular, to conditional random fields (CRFs). We show that VE can be naturally encoded into a CRF model as potential functions. More importantly, we propose a novel semi-supervised machine learning objective for estimating a CRF model integrated with VE. The objective can be optimized using the Expectation-Maximization algorithm while maintaining the discriminative nature of CRFs. When evaluated on the *CLASSIFIEDS* data, our approach significantly outperforms the best known solutions reported on this task.

1 Introduction

Statistical approaches to sequential labeling problems rely on necessary training data to model the uncertainty of a sequence of events. Human's prior knowledge about the task, on the other hand, often requires minimum cognitive load to specify, and yet can provide information often complementary to that offered by a limited amount of training data. Whenever prior knowledge becomes available, it is desired that such information is integrated to a probabilistic model to improve learning.

Virtual evidence (VE), first introduced by Pearl (1988), offers a principled and convenient way of incorporating external knowledge into Bayesian networks. In contrast to *standard evidence* (also known as observed variables), VE expresses a prior belief over values of random variables. It has been shown that VE can significantly extend the modeling power of Bayesian networks without complicating the fundamental inference methodology (Bilmes, 2004; Reynolds and Bilmes, 2005).

This work extends the use of VE to undirected graphical models and, in particular, to conditional random fields (CRFs). We show that VE can be naturally encoded into an undirected graphical model as potential functions. More importantly, we discuss a semi-supervised machine learning setting for estimating CRFs with the presence of VE. As the conditional likelihood objective of CRFs is not directly maximizable with respect to unlabeled data, we propose a novel semi-supervised learning objective that can be optimized using the Expectation-Maximization (EM) algorithm while maintaining the discriminative nature of CRFs.

We apply our model to the CLASSIFIEDS data (Grenager et al., 2005). Specifically, we use VE to incorporate into a CRF model two types of prior knowledge specified in previous works. The first is defined based on the notion of *prototypes*, *i.e.*, example words for a given label; and the other assumes that adjacent tokens tend to have the same label. When unlabeled data becomes available, we further extend the sparse prototype information to other words based on distributional similarity. This results in so-called *collocation lists*, each consisting of a relatively large number of noisy "prototypes" for a label. Given the fact that these noisy prototypes are often located close to each other in an input sequence, we create a new type of VE based on word collocation to reduce ambiguity.

We compare our CRF model integrated with VE with two state-of-the-art models, *i.e.*, constraint-driven learning (Chang et al., 2007) and generalized expectation criteria (Mann and McCallum, 2008). Experiments show that our approach leads to sequential labeling accuracies superior to the best results reported on this task in both supervised and semi-supervised learning.

2 Related work

There have been various works that make use of prior knowledge in sequential labeling tasks. Grenager et al. (2005) explicitly constrain the transition matrix of a hidden Markov model (HMM) to favor self transitions, assuming that fields tend to consist of consecutive runs of the same label.

Prototype-drive learning (Haghighi and Klein, 2006) specifies prior knowledge by providing a few *prototypes* (*i.e.*, canonical example words) for each label. This sparse prototype information is then propagated to other words based on distributional similarity. The relation between words and their prototypes are then used as features in a Markov random field (MRF) model. Since an MRF model aims to optimize the joint probability $p(\mathbf{x}, \mathbf{y})$ of input and state sequences, it is possible to apply the EM algorithm for unsupervised/semi-supervised learning.

Constraint-driven learning (Chang et al., 2007) expresses several kinds of constraints in a unified form. In inference, a new decision function is proposed to penalize the violation of the desired constraints as follows,

$$\underset{\mathbf{y}}{\text{argmax}}\ \lambda \cdot F(\mathbf{x}, \mathbf{y}) - \sum_k \rho_k d(\mathbf{y}, 1_{C_k}(\mathbf{x})) \quad (1)$$

Here $\lambda \cdot F(\mathbf{x}, \mathbf{y})$ is a linear decision function applicable to a number of sequential models, such as HMMs, MRFs and CRFs. Function d is implemented as the Hamming distance (or its approximation) between a hypothesis sequence and the space of state sequences that satisfy the constraint C_i. Due to the nature of the distance function, their work approximates EM training by finding the top K hypothesis sequences and using them as newly labeled instances to update the model. This process is repeated for a number of iterations in a self-training fashion (Yarowsky, 1995).

Generalized expectation criteria (Mann and McCallum, 2008) represent prior knowledge as *labeled features*, and use such information to regularize semi-supervised learning for CRFs. Formally, their learning objective consists of the standard CRF training objective, plus a Gaussian prior on model parameters and an additional regularization term:[1]

$$\sum_i \log p_\lambda(\mathbf{y}^{(i)}|\mathbf{x}^{(i)}) - \frac{1}{2\sigma^2}\|\lambda\|^2 - \rho D(\hat{p}||\tilde{p}_\lambda) \quad (2)$$

In the last term, $\hat{p}$ and $\tilde{p}_\lambda$ both refer to conditional distributions of labels given a feature. While the former is specified by prior knowledge, and the latter is estimated from unlabeled data.

Our approach incorporates prior knowledge as virtual evidence to express preferences over the values of a set of random variables. The notion of VE was first introduced by Pearl (1998) and further developed by Bilmes (2004), both in the context of Bayesian networks. Different from constraint-driven learning, VE can be formally encoded as part of a graphical model. The fundamental inference methodology, therefore, does not need to be altered. Moreover, VE has the flexibility of representing various kinds of prior knowledge. For example, Reynolds and Bilmes (2005) use VE that explicitly favors self transitions in dynamic Bayesian networks.

This work extends the use of VE to CRFs. In essence, VE herein can be viewed as probabilistic constraints in an undirected graph that allow exact inference. One of the biggest challenges of such a model lies in the semi-supervised machine learning setting. Since the entire state sequence of an unlabeled instance remains hidden, the conditional likelihood objective of CRFs is not directly optimizable. There have been a number of works that address this problem for conditional models. For example, *minimum entropy regularization* (Grandvalet and Bengio, 2004; Jiao et al., 2006), aims to maximize the conditional likelihood of labeled data while minimizing the conditional entropy of unlabeled data:

$$\sum_i \log p_\lambda(\mathbf{y}^{(i)}|\mathbf{x}^{(i)}) - \frac{1}{2\sigma^2}\|\lambda\|^2 - \rho H(\mathbf{y}|\mathbf{x}) \quad (3)$$

This approach generally would result in "sharper" models which can be data-sensitive in practice.

Another approach (Suzuki and Isozaki, 2008) embeds a joint probability model (HMM in their

[1] We slightly modify the notation here to be consistent with the rest of the paper.

case) into a CRF model as a new potential function. Semi-supervised learning is then conducted by iteratively (1) fixing the HMM and updating CRF parameters on labeled data and (2) fixing the CRF model and updating the HMM on unlabeled data.

Additionally, when unlabeled instances have partial labeling information, it is possible to optimize a marginal distribution of the conditional likelihood, *i.e.*, $p_\lambda(\mathbf{y}_o^{(i)}|\mathbf{x})$, on unlabeled data. Here $\mathbf{y}_o^{(i)}$ is a subvector of $\mathbf{y}^{(i)}$ that denotes the set of observed state variables. The optimization can be done in a similar fashion as training a hidden-state CRF model (Quattoni et al., 2007).

3 Task

We consider the problem of extracting fields from free-text advertisements. We use the CLASSIFIEDS data (Grenager et al., 2005) which consists of 8767 ads for apartment rental. 302 of the ads in the CLASSIFIEDS data have been manually-labeled with 12 fields, including *size*, *rent*, *neighborhood* and so on. The labeled data has been divided into train/dev/test sets with 102/100/100 ads respectively. The evaluation metric is the token-level accuracy where tokens include both words and punctuations.

Our goal in this work is two folds: (1) leverage both the training data and the prior knowledge specified for this task for supervised learning, and (2) additionally use the unlabeled data for semi-supervised learning. We exploit two types of prior knowledge:

- K1: *label consistency with prototypes*;
- K2: *label consistency within a sentence*.

K1 involves a set of prototype lists. Each list is attached with a label and consists of a set of example words for that label. In this work, we use the prototype lists originally defined by Haghighi and Klein (2006) (HK06) and subsequently used by Chang et al. (2005) (CRR07) and Mann and McCallum (2008) (MM08). The labels as well as their prototypes are shown in the first two columns of Table 1. Our model is desired to be consistent with such prototype information. Secondly, K2 means that tokens tend to have consistent labels within a sentence. A similar type of prior knowledge is implemented by CRR07 as a constraint in inference.

4 Conditional Random Fields

Conditional random fields are a probabilistic model that directly optimizes the conditional probability of a state (label) sequence given an input sequence (Lafferty et al., 2001). Formally, we let $\mathbf{x} = (x_1, x_2, \ldots, x_T)$ denote an input sequence of T tokens, and $\mathbf{y} = (y_1, y_2, \ldots, y_T)$ the corresponding state sequence. We further augment $\mathbf{y}$ with two special states, *Start* and *End*,[2] represented by y_0 and y_{T+1} respectively. A linear-chain CRF model is an undirected graphical model as depicted in Figure 1(a), with the conditional probability given by

$$p_\lambda(\mathbf{y}|\mathbf{x}) = \frac{1}{Z_\lambda(\mathbf{x})} \prod_t \psi_\lambda^{(t)}(\mathbf{x}, y_{t-1}, y_t) \qquad (4)$$

The partition function $Z_\lambda(\mathbf{x})$ normalizes the exponential form to be a probability distribution. $\psi_\lambda^{(t)}$ are a set of potential functions defined on the *maximum cliques* of the graph, *i.e.*, $(\mathbf{x}, y_{t-1}, y_t)$ in the case of a linear-chain CRF model. The potential functions are typically in the form of

$$\psi_\lambda^{(t)}(\mathbf{x}, y_{t-1}, y_t) = \exp\Big(\lambda \cdot f(\mathbf{x}, y_{t-1}, y_t, t)\Big) \qquad (5)$$

where λ is a weight vector and f is a feature vector of arbitrary functions of the corresponding clique.

Given a set of labeled examples $\{\mathbf{x}^{(i)}, \mathbf{y}^{(i)})\}_{i=1}^m$, we can estimate model parameters in a supervised machine learning setting. The objective is to estimate λ that maximizes the conditional likelihood while regularizing the model size:

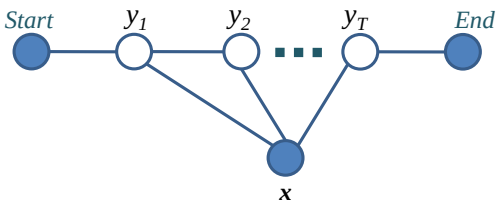
$$\mathcal{L}_1 = \sum_{i=1}^m \log p_\lambda(\mathbf{y}^{(i)}|\mathbf{x}^{(i)}) - \frac{1}{2\sigma^2}\|\lambda\|^2 \qquad (6)$$

In this work, we optimize $\mathcal{L}_1$ using stochastic gradient descent and use the accuracy on the development set as the stopping criterion.

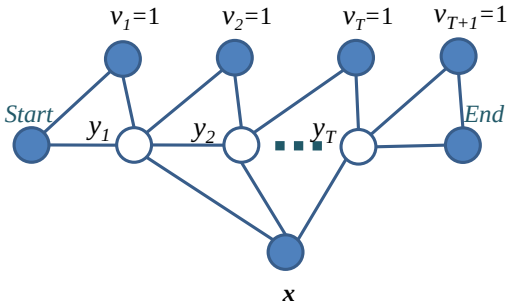
5 CRFs with Virtual Evidence

A canonical way of using virtual evidence (VE) in Bayesian networks is to have a directed edge from a hidden variable h to a VE variable v. The variable v will always be observed with a particular value, *e.g.*, $v = 1$, but the actual value itself does not matter. The prior knowledge about h is

[2] *Start* and *End* are with regard to a document, which are different from start and end of a sentence.



(a)



(b)

Figure 1: Graphical model representations of (a) a CRF model and (b) a CRF model integrated with virtual evidence. Solid and empty nodes denote observed and hidden variables respectively.

expressed via the conditional probability $p(v = 1|h)$. For example, by setting $p(v = 1|h = a) > p(v = 1|h = b)$, we know that $h = a$ is more likely a event than $h = b$. This conditional distribution is not learned from data, Instead, it is predefined in such a way that reflects a prior belief over the value of h.

VE can be encoded in an undirected graphical model in a similar fashion. For our task, we modify the structure of a linear-chain CRF model as depicted in Figure 1(b) — we create a sequence of VE variables, denoted by $v_1, v_2, \ldots, v_{T+1}$, in parallel to the state variables. Each v_t is assigned a constant 1 (one), and is connected with y_{t-1} and y_t, forming a new set of maximum cliques (y_{t-1}, y_t, v_t), $t = 1, \ldots, T + 1$. We create cliques of size 3 because it is the minimum size required to represent the prior knowledge used in our task, as will be discussed shortly. However, it is possible to have a different graph structure to incorporate other types of prior knowledge, *e.g.*, using large cliques to represent constraints that involve more variables.

Next, in analogy to Equation (5), we define the corresponding potential functions as follows,

$$\phi^{(t)}(y_{t-1}, y_t, v_t) = \exp\Big(\omega \cdot s(y_{t-1}, y_t, v_t, t)\Big) \tag{7}$$

s is a vector of VE feature functions and ω is the corresponding weight vector with pre-defined values. Given the new graphical model in Figure 1(b). It is natural to model the conditional probability of the state sequence given *both* the standard evidence and the VE as follows,

$$p_\lambda(\mathbf{y}|\mathbf{x}, \mathbf{v}) = \frac{1}{Z_\lambda(\mathbf{x}, \mathbf{v})} \prod_t \psi_\lambda^{(t)}(\mathbf{x}, y_{t-1}, y_t)\phi^{(t)}(y_{t-1}, y_t, v_t) \tag{8}$$

Analogous to using $p(v = 1|h)$ in Bayesian networks, we can utilize $\phi^{(t)}(y_{t-1}, y_t, \mathbf{v} = \mathbf{1})$ to express preferences over state hypotheses in a CRF model. In general, the function form of $\phi^{(t)}$ may or may not depend on the input $\mathbf{x}$. Even when $\phi^{(t)}$ does depend on $\mathbf{x}$, the relation is completely determined by external knowledge/systems (as opposed to by data). Thus we do not explicitly connect $\mathbf{v}$ with $\mathbf{x}$ in the graph.

5.1 Incorporating prior knowledge

Now we show how to represent the prior knowledge introduced in Section 3 using the VE feature functions. Unless otherwise stated, we assume $v_t = 1$ for all $t = 1, \ldots, T$ and simply use v_t instead of $v_t = 1$ in all equations. First, we define a VE function s_1 that represents K1: *label consistency with prototypes*. We let P_l denote a prototype list associated with the label l. If x_t belongs to P_l, we should prefer $y_t = l$ as opposed to other values. To this end, for cases where $x_t \in P_l$, we set s_1 as

$$s_1(y_t, v_t, t) = \begin{cases} 1 & \text{if } y_t = l \\ 0 & \text{otherwise} \end{cases} \tag{9}$$

On the other hand, if x_t is not a prototype, we will always have $s_1(y_t, v_t, t) = 0$ for all hypotheses of y_t. The impact of this prior knowledge is controlled by the weight of s_1, denote by ω_1. At one extreme where $\omega_1 = 0$, the prior knowledge is completely ignored in training. At the other extreme where $\omega_1 \to +\infty$, we constrain the values of state variables to agree with the prior knowledge. Note that although s_1 is implicitly related to $\mathbf{x}$, we do not write s_1 as a function of $\mathbf{x}$ for consistency with the general definition of VE.

To represent K2: *label consistency within a sentence*, we define a second VE feature function s_2 with weight ω_2. Assume that we have an external system that detects sentence boundaries. If it is determined that x_t is *not* the start of a sentence, we set s_2 as

$$s_2(y_{t-1}, y_t, v_t, t) = \begin{cases} 1 & \text{if } y_{t-1} = y_t \\ 0 & \text{otherwise} \end{cases} \quad (10)$$

It is easy to see that this would penalize state transitions within a sentence. On the other hand, if x_t is a sentence start, we set $s_2(y_{t-1}, y_t, v_t, t) = 0$ for all possible (y_{t-1}, y_t) pairs. In this work, we use a simple heuristics to detect sentence boundaries: we determine that x_t is the start of a sentence if its previous token x_{t-1} is a period (.), a semi-colon (;) or an acclamation mark (!), *and* if x_t is not a punctuation.

5.2 Semi-supervised learning

When a large amount of unlabeled data is available, it is often helpful to leverage such data to improve learning. However, we cannot directly optimize $p(\mathbf{y}|\mathbf{x}, \mathbf{v})$ since the correct state sequences of the unlabeled data are hidden. One heuristic approach is to adapt the *self-training* algorithm (Yarowsky, 1995) to our model. More specifically, for each input in the unlabeled dataset $\{\mathbf{x}^{(i)}\}_{i=m+1}^{n}$, we decode the best state sequence,

$$\hat{\mathbf{y}}^{(i)} = \underset{\mathbf{y}^{(i)}}{\text{argmax}}\ p(\mathbf{y}^{(i)}|\mathbf{x}^{(i)}, \mathbf{v}^{(i)}) \quad (11)$$

Then we use $\{(\mathbf{x}^{(i)}, \hat{\mathbf{y}}^{(i)})\}_{i=m+1}^{n}$ in addition to the labeled data to train a supervised CRF model. This approach, however, does not have a theoretical guarantee on optimality unless certain nontrivial conditions are satisfied (Abney, 2004).

On the other hand, it is well known that unlabeled data can be naturally incorporated using a generative approach that models a joint probability (Nigam et al., 2000). This is achieved by maximizing a marginal distribution of the joint probability over hidden variables. Inspired by the generative approach, we propose to explicitly model $p(\mathbf{y}, \mathbf{v}|\mathbf{x})$. In contrast to Equation (8), here we jointly model $\mathbf{y}$ and $\mathbf{v}$ but the probability is still conditioned on $\mathbf{x}$. This "joint" distribution should be chosen such that it results in the same conditional distribution $p(\mathbf{y}|\mathbf{x}, \mathbf{v})$ as defined in Equation (8). To this end, we define $p_\lambda(\mathbf{y}, \mathbf{v}|\mathbf{x})$ as

$$p_\lambda(\mathbf{y}, \mathbf{v}|\mathbf{x}) = \frac{1}{Z'_\lambda(\mathbf{x})} \prod_t \psi_\lambda^{(t)}(\mathbf{x}, y_{t-1}, y_t)\phi^{(t)}(y_{t-1}, y_t, v_t) \quad (12)$$

Here $Z'_\lambda(\mathbf{x})$ is a normalization function obtained by summing the numerator over both $\mathbf{y}$ and $\mathbf{v}$. By applying the Bayes rule, it is easy to see that $p(\mathbf{y}|\mathbf{x}, \mathbf{v})$ is exactly equal to Equation (8).

Given unlabeled data $\{\mathbf{x}^{(i)}\}_{i=m+1}^{n}$, we aim to optimize the following objective,[3]

$$\mathcal{L}_2 = \sum_{i=1}^{m+n} \log p_\lambda(\mathbf{v}^{(i)}|\mathbf{x}^{(i)}) - \frac{1}{2\sigma^2}\|\lambda\|^2 \quad (13)$$

This is essentially the marginal distribution of $p(\mathbf{y}, \mathbf{v}|\mathbf{x})$ over hidden variables $\mathbf{y}$. Here we ignore the labels of the dataset $\{(\mathbf{x}^{(i)}, \mathbf{y}^{(i)})\}_{i=1}^{m}$, but we do use the label information in initializing the model which will described in Section 6. To optimize such an objective, we apply the EM algorithm in the same fashion as is used in a generative approach. In other words, we iteratively optimize $Q(\lambda) = \sum_{\mathbf{y}} p_{\lambda^g}(\mathbf{y}|\mathbf{x}, \mathbf{v}) \log p_\lambda(\mathbf{y}, \mathbf{v}|\mathbf{x})$ where λ^g denotes the model estimated from the previous iteration. The gradient of the Q function is straightforward to compute with the result given by

$$\frac{\partial Q(\lambda)}{\partial \lambda_k} = \sum_t \sum_{y_{t-1}, y_t} f_k(y_{t-1}, y_t, \mathbf{x}, t) \cdot \Big(p_\lambda(y_{t-1}, y_t|\mathbf{x}, \mathbf{v}) - p_\lambda(y_{t-1}, y_t|\mathbf{x}) \Big) \quad (14)$$

We keep two sets of accumulators in running the Forward-Backward algorithm, one for computing $p_\lambda(y_{t-1}, y_t|\mathbf{x}, \mathbf{v})$ and the other for computing $p_\lambda(y_{t-1}, y_t|\mathbf{x})$. Loosely speaking, the model will converge to a local optimum if the difference between these two posterior probabilities becomes trivial.

5.3 Collocation based virtual evidence

Prior knowledge represented by prototypes is typically sparse. This sparse information, however, can be propagated across all data based on distributional similarity (Haghighi and Klein, 2006). Following the same idea, we extend the prototype lists as follows. (1) We merge all prototypes in P_l into a single word type w_l. (2) For each word

[3] In Equation (13), the fact that $\mathbf{v}$ is assigned a constant 1 does not mean $p(\mathbf{v} = \mathbf{1}|\mathbf{x}) = 1$ (Bilmes, 2004)

Label	Prototype lists of HK06	Collocation lists (top examples)
ADDRESS	address carlmont	[4-digit] street [3-digit] streets
AVAILABLE	immediately begin cheaper	available
CONTACT	[phone] call [time]	[email] appointment email see today ...
FEATURES	kitchen laundry parking	room new covered building garage ...
NEIGHBORHOOD	close near shopping	transportation center located restaurants ...
PHOTOS	pictures image link	[url] click view photos
RENT	$ month [amount]	lease deposit security year agreement ...
RESTRICTIONS	pets smoking dog	ok sorry please allowed negotiable ...
ROOMMATES	roommate respectful drama	
SIZE	[1-digit] br sq	[4-digit] [3-digit] ft bath ba ...
UTILITIES	utilities pays electricity	water included owner garbage paid

Table 1: Field labels (except *other*) for the CLASSIFIEDS task, their respective prototype lists specified by prior knowledge, and collocation lists mined from unlabeled data.

in the corpus, we collect a context vector of the counts of all words (excluding stop words) that occur within a window of size k in either direction, where the window is applied only within sentence boundaries. (3) Latent semantic analysis (Deerwester et al., 1990) is performed on the constructed context vectors. (4) In the resulting latent semantic space, all words (except stop words) that have a high enough dot product with w_l will be grouped to form a new set, denoted as C_l, which is a superset of P_l. In this regard, C_l can be viewed as lists of noisy "prototypes". As observed in HK06, another consequence of this method is that many neighboring tokens will share the same prototypes.

Differently from previous works, we use C_l directly as virtual evidence. We could apply s_1 in Equation (9) when $x_t \in C_l$ (as opposed to when $x_t \in P_l$). This, however, would contaminate our model since C_l are often noisy. For example, "water" is found to be distributionally similar to the prototypes of *utilities*. Although in most cases "water" indeed means *utilities*, it can mean *features* in the context of "water front view". To maximally reduce ambiguity, we propose to apply s_1 in Equation (9) if *both* of the following conditions hold,

(1) $x_t \in C_l$

(2) There exists τ s.t. $|\tau - t| < k$, and $x_\tau \in C_l$

In other words, we will impose a non-uniform prior on y_t if $x_t \in C_l$ "collocates", within k tokens, with another word that belongs to C_l. Based on K2, it is reasonable to believe that neighboring tokens tend to share the same label. Therefore, knowing that two tokens close to each other both belong to C_l would strengthen our belief that either word is likely to have label l. We thus refer to this type of virtual evidence as *collocation-based VE*, and refer to C_l as *collocation lists*.

6 Evaluation

We use the CLASSIFIEDS data provided by Grenager et al. (2005) and compare with results reported by CRR07 (Chang et al., 2007) and MM08 (Mann and McCallum, 2008) for both supervised and semi-supervised learning. Following all previous works conducted on this task, we tokenized both words and punctuations, and created a number of regular expression tokens for phone numbers, email addresses, URLs, dates, money amounts and so on. However, we did not tokenize newline breaks, as CRR07 did, which might be useful in determining sentence boundaries. Based on such tokenization, we extract n-grams, $n = 1, 2, 3$, from the corpus as features for CRFs.

As described in Section 3, we integrate the prior knowledge K1 and K2 in our CRF model. The prototypes that represent K1 are given in Table 1. CRR07 used the same two kinds of prior knowledge in the form of constraints, and they implemented another constraint on the minimum number of words in a field chunk. MM08 used almost the same set of prototypes as labeled features, but they exploited two sets of 33 additional features for some experiments. In this regard, the comparison between CRR07, MM08 and the method presented here cannot be exact. However, we show that while our prior knowledge is no more than that used in previous works, our approach is able

Supervised model	# labeled examples 10	25	100
CRR07: HMM	61.6	70.0	76.3
+ Constr in decoding	66.1	73.7	80.4
MM08: CRF	64.6	72.9	79.4
CRF	62.3	71.4	79.1
+ VE in decoding	**68.9**	**74.6**	**81.1**
CRF + VE (auto weights)	48.0	54.8	59.8
+ VE in decoding	66.0	72.5	80.9

Table 2: Token-level accuracy of supervised learning methods; "+ VE" refers to the cases where both kinds of prior knowledge, K1 and K2, are incorporated as VE in the CRF model.

Semi-supervised models	# labeled examples 10	25	100
CRR07: HMM + Constr	70.9	74.8	78.6
+ Constr in decoding	74.7	78.5	81.7
MM08: CRF + GE	72.6	76.3	80.1
CRF + VE (Self-train)	69.0	74.2	81.4
+ VE in decoding	69.1	75.2	81.2
CRF + Col-VE (Self-train)	73.1	76.4	81.8
+ Col-VE in decoding	75.7	77.6	82.9
CRF + Col-VE (EM)	78.3	79.1	82.7
+ Col-VE in decoding	**78.8**	**79.5**	**82.9**

Table 3: Token-level accuracy of semi-supervised learning methods. "+ Col-VE" refers to cases where collocation-based VE is integrated in the CRF model in addition to the VE representing K1 and K2.

to achieve the state-of-art performance.

6.1 Decoding settings

Depending on whether VE is used at test time, we explore two decoding settings in all experiments:

1. Find $\mathbf{y}$ that maximizes $p_\lambda(\mathbf{y}|\mathbf{x})$ as in standard CRF decoding, ignoring virtual evidence.
2. Find $\mathbf{y}$ that maximizes $p(\mathbf{y}|\mathbf{x}, \mathbf{v})$. We use "+ VE in decoding" to represent this setting.

These two scenarios are analogous to those in CRR07 which conducted HMM decoding without/with constraints applied. We use "+ constr. in decoding" to represent the latter scenario of their work. MM08, on the other hand, found no accuracy improvement when adding constraints at test time.

Note that in our second decoding setting, the weights for the VE feature functions, *i.e.*, ω_1 and ω_2, are tuned on the development set. This is done by a greedy search that first finds the best ω_1, and then finds the best ω_2 while fixing the value of ω_1, both with a step size 0.5.

6.2 Supervised learning results

First, we experimented with a standard CRF model with VE applied neither in training nor in decoding. As shown in Table 2, our CRF implementation performed slightly worse than the implementation by MM08, probably due to slight difference in tokenization. Secondly, we used the same CRF model but additionally applied VE in decoding, corresponding to the second setting in Section 6.1. This method gave a significant boost to the tagging performance, yielding the best supervised learning results (shown as bolded in the table). This proves that the prior knowledge is indeed complementary to the information offered by the training data.

Similar to the second decoding setting that incorporates VE, we can have a counterpart setting at training time. In other words, we can optimize $p_\lambda(\mathbf{y}|\mathbf{x}, \mathbf{v})$ instead of $p_\lambda(\mathbf{y}|\mathbf{x})$ during learning. In deciding $\omega = (\omega_1, \omega_2)$, it is possible to learn ω from data in the same way as how we learn λ. This, however, might undermine the role of other useful features since we do not always have sufficient training data to reliably estimate the weight of prior knowledge. As shown in Table 2, we experimented with learning ω automatically (shown as "auto weights"). While applying VE with such weights in both training and decoding worked reasonably well, applying VE only in training but not in decoding yielded very poor performance (probably due to excessively large estimates of ω_1 and ω_2). Additionally, we repeated the above experiment with manually specified weights, but did not find further accuracy improvement over the best supervised learning results.

6.3 Semi-supervised learning results

One natural way of leveraging the unlabeled data (more than 8K examples) is to perform semi-supervised learning in a self-training fashion. To this end, we used our best supervised model in Table 2 to decode the unlabeled examples as well as the test-set examples (by treating them as unlabeled). Note that by doing this our comparison with CRR07 and MM08 cannot be exact as they

sampled the unlabeled examples, with different rates, for semi-supervised learning, while we used as much data as possible. We applied the same ω that was used for the supervised model, and then combined the newly labeled examples, in addition to the manually labeled ones, as training data to learn a supervised CRF model. On this particular dataset, we did not find it helpful by selecting automatically labeled data based on a confidence threshold. We simply used all data available in self-training. This paradigm is referred to as "CRF + VE (self-train)" in Table 3. When no VE is applied at test time, this semi-supervised CRF model significantly outperformed the best model in Table 2. When applying VE at test time, however, the improvement over its supervised counterpart became trivial.

Next, following Section 5.3, we collected context vectors on the unlabeled data using a window size $k = 3$, and extracted the top 50 singular vectors therefrom.[4] We created collocation lists that contain words close to the merged prototype words in the latent semantic space. Some examples are given in the last column of Table 1. We then augmented the prototype-based VE based on the following rules: If x_t belongs to any prototype list P_l, we directly apply s_1 in Equation (9); otherwise, we apply s_1 if x_t *and* at least one neighbor (within 3 tokens from x_t) belong to the same collocation list C_l. In our experiments, we let "Col-VE" represent such collocation-based VE. We conducted self-training using a CRF model integrated with Col-VE, where ω was tuned *a priori* by testing the same model on the development set. As shown in the table, "CRF + Col-VE (self-train)" gave significant accuracy improvement over "CRF + VE", while adding Col-VE at test time further boosted the performance. The accuracies were already on par with the best results previously reported on this task.

Finally, we implemented the EM algorithm proposed in Section 5.2 that iteratively optimizes $p(\mathbf{v}|\mathbf{x})$ on all data. The model was initialized by the one obtained from "CRF + Col-VE (self-train)". After the model was initialized, we performed the EM algorithm until the model reached a maximum accuracy on the development set. Note that in some cases, we observed a development-set accuracy degradation after the first iteration of the EM, but the accuracy quickly recovered from the second iteration and kept increasing until a maximum accuracy was reached.[5] As shown in the last two rows in Table 3, this method is clearly advantageous over self-training, leading to the best tagging accuracies in both decoding settings. Our model achieved $2.6\% - 5.7\%$ absolute accuracy increases in the three training settings compared with MM08 which had the best results without using any constraints in decoding. When applying VE at test time, our model was $1.2\% - 4.1\%$ better than CRR07 which had the best overall results. Additionally, when compared with supervised learning results, our best semi-supervised model trained on only 10 labeled examples performed almost as well as a standard supervised CRF model trained on 100 labeled examples.

[4]The same configuration is used in HK06.

7 Conclusions

We have presented the use of virtual evidence as a principled way of incorporating prior knowledge into conditional random fields. A key contribution of our work is the introduction of a novel semi-supervised learning objective for training a CRF model integrated with VE. We also found it useful to create so-called collocation-based VE, assuming that tokens close to each other tend to have consistent labels. Our evaluation on the CLASSIFIEDS data showed that the learning objective presented here, combined with the use of collocation-based VE, yielded remarkably good accuracy performance. In the future, we would like to see the application of our approach to other tasks such as (Li et al., 2009).

References

Steven Abney. 2004. Understanding the Yarowsky algorithm. *Association for Computational Linguistics*, 30(3):365–395.

Jeff Bilmes. 2004. On soft evidence in Bayesian networks. Technical Report UWEETR-2004-0016, University of Washington, Dept. of Electrical Engineering.

Ming-Wei Chang, Lev Ratinov, and Dan Roth. 2007. Guiding semi-supervision with constraint-driven learning. In *Proceedings of ACL*.

[5]The initial degradation is probably due to the fact that self-training can result in an initial model with decent accuracy but low $p(\mathbf{v}|\mathbf{x})$; thus the EM algorithm that maximizes $p(\mathbf{v}|\mathbf{x})$ may temporarily decrease the accuracy.

Scott Deerwester, Susan Dumais, Thomas Landauer, George Furnas, and Richard Harshman. 1990. Indexing by latent semantic analysis. *Journal of the American Society of Information Science*, 41(6):391–407.

Yves Grandvalet and Yoshua Bengio. 2004. Semi-supervised learning by entropy minimization. In *Advances in Neural Information Processing Systems*.

Trond Grenager, Dan Klein, and Christopher D. Manning. 2005. Unsupervised learning for field segmentation models for information extraction. In *Proceedings of Association of Computational Linguistics*.

Aria Haghighi and Dan Klein. 2006. Prototype-driven learning for sequence models. In *Proceedings of HLT-NAACL*.

Feng Jiao, Shaojun Wang, Chi-Hoon Lee, Russell Greiner, and Dale Schuurmans. 2006. Semi-supervised conditional random fields for improved sequence segmentation and labeling. In *Proceedings of ACL*.

John Lafferty, Andrew McCallum, and Fernando Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the International Conference on Machine Learning*, pages 282–289.

Xiao Li, Ye-Yi Wang, and Alex Acero. 2009. Extracting structured information from user queries with semi-supervised conditional random fields. In *Proceedings of SIGIR*.

Gideon Mann and Andrew McCallum. 2008. Generalized expectation criteria for semi-supervised learning of conditional random fields. In *Proceedings of ACL*.

Kamal Nigam, Andrew Kachites Mccallum, Sebastian Thrun, and Tom Mitchell. 2000. Text classification from labeled and unlabeled documents using EM. *Machine Learning*, 39:103–134.

Judea Pearl. 1988. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann, 2nd printing edition edition.

A. Quattoni, S. Wang, L.-P. Morency, M. Collins, and T. Darrell. 2007. Hidden conditional random fields. *IEEE Transaction on Pattern Analysis and Machine Intellegence*, 29(10):1848–1852.

Sheila Reynolds and Jeff Bilmes. 2005. Part-of-speech tagging using virtual evidence and negative training. In *Proceedings of HLT/EMNLP*.

Jun Suzuki and Hideki Isozaki. 2008. Semi-supervised sequential labeling and segmentation using gigaword scale unlabeled data. In *Proceedings of ACL/HLT*.

David Yarowsky. 1995. Unsupervised word sense disambiguation rivaling supervised methods. In *Proceedings of ACL*, pages 189–196.