2009 Ninth IEEE International Conference on Data Mining

Semi-Supervised Sequence Labeling with Self-Learned Features

Yanjun Qi*, Pavel Kuksa†, Ronan Collobert*, Kunihiko Sadamasa*, Koray Kavukcuoglu‡ and Jason Weston§
*Machine Learning Department, NEC Labs America Inc, Princeton, NJ, USA.
{yanjun,collober,kunihiko}@nec-labs.com
†Dept of Computer Science, Rutgers University, Piscataway, NJ, USA
pkuksa@cs.rutgers.edu
‡Computer Science Dept, New York University, New York, NY, USA
koray@cs.nyu.edu
§Google Research NY, New York, NY, USA
jaseweston@gmail.com

***Abstract*—Typical information extraction (IE) systems can be seen as tasks assigning labels to words in a natural language sequence. The performance is restricted by the availability of labeled words. To tackle this issue, we propose a semi-supervised approach to improve the sequence labeling procedure in IE through a class of algorithms with *self-learned features* (SLF). A supervised classifier can be trained with annotated text sequences and used to classify each word in a large set of unannotated sentences. By averaging predicted labels over all cases in the unlabeled corpus, SLF training builds class label distribution patterns for each word (or word attribute) in the dictionary and re-trains the current model iteratively adding these distributions as extra word *features*. Basic SLF models how likely a word could be assigned to target class types. Several extensions are proposed, such as learning words' class boundary distributions. SLF exhibits robust and scalable behaviour and is easy to tune. We applied this approach on four classical IE tasks: named entity recognition (German and English), part-of-speech tagging (English) and one gene name recognition corpus. Experimental results show effective improvements over the supervised baselines on all tasks. In addition, when compared with the closely related self-training idea, this approach shows favorable advantages.**

***Keywords*-semi-supervised learning; semi-supervised feature learning; information extraction; structural output learning; sequence labeling; self-learned features**

I. Introduction

Several typical problems in natural language processing (NLP) can be seen as the task of assigning labels to words in a text sequence. It is quite difficult to obtain labeled training sentences with word-level annotations compared with document-level classification tasks (such as text categorization), because hand-labeling individual words and word boundaries is much harder than assigning article-level class labels [1]. Supervised techniques have yielded great success in the NLP community, even though they are restricted by the expense of annotating the corpus. Semi-supervised learning has become one of the most natural forms of training for language processing tasks, since unlabeled language data is plentiful in this field.

Self-training [2], [3], [4], and co-training [5], [6] are popular semi-supervised methods applied in natural language processing tasks. They utilize large sets of unlabeled corpora and try to improve over supervised methods by iteratively adding self-labeled *examples* predicted by the current model. However, they are vulnerable to the incestuous training bias problem [7], [8], i.e. examples may be consistently mislabeled making the model even worse on the next iteration. To combat this, several authors have proposed schemes for only adding examples that meet a selection criterion [4], [7], [9], but these heuristic choices still might yield unreliable results.

In this paper[1] we propose a simple and scalable semi-supervised strategy that works by providing semi-supervision at the level of *feature attributes* rather than *examples*. Under the assumption that words (and word attributes) carry rich label information, we learn to derive a group of features (which we name "self-learned features" (SLF)) that are related to the distribution of target classes through a large unlabeled corpus. These distribution patterns are used as extra *features* to retrain our base supervised classifier in an iterative fashion. Basic SLF learns to model words' probabilities to every possible target class. We also propose several extensions to our basic approach: (1) Boundary SLF learns to represent words' class boundary patterns; (2) Attribute SLF tries to model target class distribution patterns for each value of the discrete word attributes (for instance, capitalization properties of words); (3) Clustered SLF clusters word according to SLF values and the derived cluster ID would be used as novel extra features.

We tested our approach on four classic sequence labeling tasks: two CoNLL-2003 [11] shared tasks (German and English Named Entity Recognition (NER)), one English Part-of-Speech tagging [12] and the GM BioCreativeII competition (gene name recognition) [13]. Two state-of-the-art natural language processing systems are used as baseline supervised methods to train SLF: (a) a neural network (NN) model [14] for unified NLP (b) a conditional random field

[1]This paper is an extension of a poster publication [10].

1550-4786/09 $26.00 © 2009 IEEE
DOI 10.1109/ICDM.2009.40

(CRF) [15] which has shown success in many information extraction tasks. We observed improvements from using SLF in various setups: compared to the baseline classifiers, to semi-supervised auxillary task and to self-training, whenever we applied it. In particular we achieved a state-of-the art result of 75.72 F1 on the German NER task and token error rate 2.73% on the English POS task. Moreover, SLF models exhibit robust behavior since noisy self-labeled examples are *not* added (as in self-training). It is also highly scalable to large unlabeled corpora (for instance, English Wikipedia).

II. Methods

Unlike most popular semi-supervised approaches (details in Section IV), we propose to induce features from a large corpus of unannotated examples in a supervised fashion, and then use these features to augment the feature space of the labeled set. Since this is an orthogonal method for improving accuracy, it can be combined with many of the other semi-supervised methods (section IV).

SLF relies on the key observation that individual words carry significant label information, since they are fundamental building blocks of NLP systems. In the following, we describe our basic model: word-level SLF first, and then go on to describe several extensions.

We consider the setting where one is given labeled training examples $\{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1,\ldots,L} \in \mathcal{X} \times \mathcal{Y}$ and a large unlabeled set of examples $\{\mathbf{x}_i^*\}_{i=1,\ldots,U} \in \mathcal{X}$ where the unlabeled set is much larger than the labeled one ($U \gg L$).

In particular for the sequence labeling tasks which we focus on in this paper, $\mathcal{X}$ is the set of all sequences composed of elements which take on a finite set of possible values, e.g. sequences of words (or in the general case this could include other discrete types of word attributes as well, e.g. capital types, POS tags, stem-ends, etc.). That is, we will assume an input sequence $\mathbf{x} = (x_1, \ldots, x_{|\mathbf{x}|})$, where $|\mathbf{x}|$ means the length of the sequence. Its j_{th} item is $x_j \in \mathcal{D}$, with $\mathcal{D}$ representing a dictionary of size $|\mathcal{D}|$, for instance a vocabulary of English words. The labels $\mathcal{Y} \in \{1, \ldots, K\}$ are the K classes a sequence can be assigned to, e.g. NER system labels atomic elements in the sentence into categories such as "PERSON", "COMPANY", or "LOCATION".

A. Basic method: Word SLF

We define the basic self-learned features for a given word $w \in \mathcal{D}$ as a feature vector $\text{SLF}(w) \in \mathbb{R}^K$ which models the probability to each target class this word might be assigned with,

$$\text{SLF}(w) = (\text{SLF}(w)_1, \ldots, \text{SLF}(w)_K),$$

where

$$\text{SLF}(w)_i = P(\mathbf{y} = i | w, \text{ where } w \in \mathbf{x}), \tag{1}$$

That is, the i^{th} dimension measures the probability of label $y = i$ being assigned given that word w is present in the input sequence $\mathbf{x}$ (Figure 1). This distribution is of course unknown but can be estimated from the training set or, critically, can be *re-estimated* using an unlabeled corpus by applying a trained classifier.

We thus define the empirical SLF for a given word as:

$$\overline{\text{SLF}}(w)_i = \frac{|\{j : f(\mathbf{x}_j^*) = i \ \wedge \ w \in \mathbf{x}_j^*\}|}{|\{k : w \in \mathbf{x}_k^*\}|}, \tag{2}$$

where $f(\cdot)$ is a classifier trained to predict $\mathbf{y} \in \mathcal{Y}$ given $\mathbf{x} \in \mathcal{X}$. In equation 2 the numerator counts the number of sequences that have been classified as type i and include the word w. The denominator describes the total number of sequences including the word w in the unlabeled corpus. Essentially this distribution measures the proportion of text sequences assigned as class i given word w is present. This distribution could be smoothed by a Bayesian Beta Prior in extreme cases (e.g. infrequent words) [16].

We hence propose the following iterative semi-supervised training algorithm (pseudo-code in Table I):

1) Define the feature representation $\phi(w)$ for a word w, and the representation for an example, sequence $\mathbf{x}$, as:
$$\Phi(\mathbf{x}) = (\phi(x_1), \ldots, \phi(x_{|\mathbf{x}|}))$$
2) Train a classifier $f(\cdot)$ on training examples $(\mathbf{x}_i, \mathbf{y}_i)$ using the feature representation $\Phi(\cdot)$.
3) Augment the representation of words with their SLF (word-class distributions in the basic SLF model):
$$\overline{\phi}(w) = (\phi(w), \overline{\text{SLF}}(w)) \tag{3}$$
using the current model $f(\cdot)$ to compute equation (2) and redefine $\Phi(\mathbf{x}) = (\overline{\phi}(x_1), \ldots, \overline{\phi}(x_{|\mathbf{x}|}))$.
4) Iterate steps 2-3 until performance does not improve.

B. Modified Word SLF in Window-based Sequence Labeling

Many NLP tasks can be treated as sequence segmentation or sequence labeling where prediction performance is measured with word-level evaluations. A classical way to handle these tasks is to utilize a *window-based* approach where the tagging algorithm considers a window of a fixed size around each word we want to label (Figure 1).

To apply SLF learning for this case, we propose the *modified self-learned features* for words where we are interested in class distributions only for the words to be labeled:

$$\overline{\text{SLF}}(w)_i = \frac{|\{j : f(\mathbf{x}_j^*) = i \ \wedge \ w = (\mathbf{x}_j^*)_m\}|}{|\{k : w = (\mathbf{x}_k^*)_m\}|}, \tag{4}$$

where

$$m = (|\mathbf{x}_j^*| + 1)/2$$

Here we only count the sequences (sliding text windows) whose middle word (with the index m) matches to the word w. However, we still augment all words in the window with $\overline{\text{SLF}}(\cdot)$ features. This actually gives a pattern of possible context (neighboring) tags for the middle word of interest.

Table I
PSEUDO-CODE FOR BASIC SLF.

Algorithm of "Semi-Supervised Sequence Labeling with Self-Learned Features"
1: Define the feature representation $\phi(w)$ for a word w, and the representation for an example (sequence) $\mathbf{x}$ as $\Phi(\mathbf{x})$;
2: Train a classifier $f(\cdot)$ on training examples $(\mathbf{x}_i, \mathbf{y}_i)$ using the feature representation $\Phi(\cdot)$;
3: Augment the representation of words to $\overline{\phi}(w)$ (Equation 3), with their empirical SLF estimated on unlabeled data U;
4: Iterate steps 2 to 3 until stopping criterion is met.

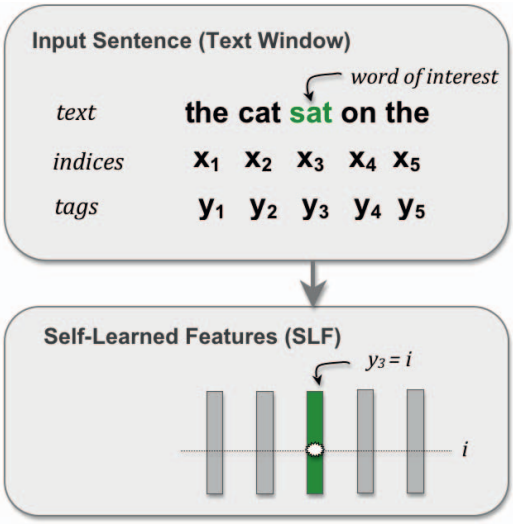

Figure 1. Modified SLF for window-based sequence labeling. Inputs $\mathbf{x}$ are windows of a fixed size ($|\mathbf{x}| = 5$ in this case). The middle word in the window is the word to be tagged.

C. Extension (1): Attribute SLF

In the basic SLF, we learn to derive the class-distribution features for each word in the vocabulary from a large unlabeled corpus. Besides words, there exist a number of word attributes that are important for typical NLP problems, such as capitalization properties of words, stem-ends and prefix strings. These attributes are normally discrete and from a finite dictionary of values. For instance, the attribute "capitalization" could have values of "YES" (the first letter of the word is capital letter) or "NO".

Similar to words, some important attributes describing words also carry rich class information, because they are also essential building blocks of natural languages. We could treat them the same way as words and learn their SLF from the unlabeled set: we call this 'Attribute SLF' (ASLF) learning.

Suppose an input example $\mathbf{x}$ is a sequence of words, plus the "capitalization" of each word. Assuming c represents the "capitalization" of each word, each element in the sequence $\mathbf{x}$ has the format $x_i = (w, c)$, and Equation 1 is modified to handle c:

$$\text{SLF}'(c)_i = P(\mathbf{y} = i|c, \text{ where } c \in \mathbf{x}), \tag{5}$$

We then augment the representation of words with both the word SLF and attribute SLF compared to Equation 3 in the iterative algorithm (Table I):

$$\overline{\phi}'(w, c) = (\phi(w), \overline{\text{SLF}}(w), \phi'(c), \overline{\text{SLF}}'(c)) \tag{6}$$

Table II
WORDS CARRYING RICH CLASS BOUNDARY INFORMATION

Each row gives an example where the words next (or very close) to the named entity (person or gene name) are informative to the target class.

... former **captain** [Chris Lewis] ...
... [Washington] **said** ...
... [Hoddle] **said** ...
... [CRKL]**, an adapter protein** ...
... [SH2-SH3-SH3] **adapter protein** ...

This simple extension could be applied to any attribute that is discrete and from a finite dictionary of values.

D. Extension (2): Boundary SLF

For NLP tasks like named entity recognition or gene name recognition, rare words (words with very low frequency) are normally the hardest examples to label in the sequence since the training set could hardly cover them. In this case, we could consider to model those words which happen frequently before (or after) a certain class label. Later, when a rare word needs to be labeled, its neighborhood words might be able to provide strong indications of its target class types if these words are always in the boundary of a certain class. Table II lists several examplar words that are usually very close to named entities (person names or gene names). Clearly some words carry strong class boundary information.

Based on the above observation, we extend the basic SLF strategy to incorporate the class bounary distributions. Basic SLF (Equation 1) models the probability of label $y = i$ being assigned given the word w present in the input sequence $\mathbf{x}$. Boundary SLF (BSLF) models how likely the word w is right before or after a certain class (assuming class t):

$$\begin{aligned} \text{SLF}''(w)_{t,1} &= P(\mathbf{y}_i = t|w \in \{(\mathbf{x}_i)_1, \ldots, (\mathbf{x}_i)_{m-1}\}) \\ \text{SLF}''(w)_{t,2} &= P(\mathbf{y}_i = t|w \in \{(\mathbf{x}_i)_{m+1}, \ldots, (\mathbf{x}_i)_{|\mathbf{x}_i|}\}) \end{aligned}$$

where m is the middle index in a given "window" $\mathbf{x}_i$ (sequence), as before. Thus, similar to Equation 3, we have

$$\overline{\phi}''(w) = (\phi(w), \overline{\text{SLF}}(w), \overline{\text{SLF}}''(w)) \tag{7}$$

We show experimentally that this extension effectively improves the gene name recognition task which suffers a lot from the "rare word" problem.

E. Extension (3): Clustered SLF

For a given word w (or attribute), SLF defines a feature vector $\text{SLF}(w) \in \mathbb{R}^{K'}$. $K' = K$ in basic SLF and its value depends on the SLF extension (Figure 1). Words exhibiting similar empirical class distribution patterns have

similar SLF feature vectors. Thus grouping them together might give stronger indications of target sequence labels or class boundaries.

Here we explored a vector quantization (VQ) technique[17] to convert SLF distibution vectors to prototype vectors. VQ clusters (or quantizes) groups of values together instead of one at a time. The groups of values are called *input vectors* and the quantization levels are called *code vectors*. The set of all *code vectors* is called the *codebook*. Quantization techniques allow the modeling of SLF feature vectors by the distribution of codebook vectors.

Formally speaking, we have SLF feature vectors for every word w in the dictionary $\mathcal{D}$,

$$\Omega = \{\text{SLF}(w_1), \text{SLF}(w_2), ..., \text{SLF}(w_D)\}$$

where $\text{SLF}(w_i) \in \mathbb{R}^{K'}$. We use $\mathcal{C}$ to represent the codebook set which includes N codebook vectors, $\mathcal{C} = \{C_1, C_2, ..., C_N\}$.

Then VQ tries to optimize (minimize) the following objective function, to find the codebook $\mathcal{C}$ and to round off the input vectors into code vectors,

$$\frac{1}{\mathcal{D} * K'} \sum_{d=1}^{\mathcal{D}} ||\text{SLF}(w_d) - C_k||^2. \quad (8)$$

For each input vector $\text{SLF}(w_d)$, VQ tries to find the best code vector C_k it could be quantized into (to optimize the above function). Each vector $\text{SLF}(w)$ would be assigned to one of the code vectors in the codebook after VQ step. Thus we use the indices of this code vector as the new feature for the word w. Essentially the whole process clusters all the words in the dictionary into multiple (N) clusters and then uses the cluster ID as the feature representation instead (a binary vector with all zeros apart from the ID^{th} dimension which is set to 1). Since this is basically a k-means clustering, we call this extension "clustered SLF" (CSLF).

F. Advantages of SLF

Like self-training and co-training our algorithm (i) iteratively tries to improve its model; and (ii) is a wrapper approach that can use a supervised (or semi-supervised) classifier as a "base learner". However, our algorithm also has the following benefits:

- It has no incestuous bias from introducing new examples with incorrect labels as in self-training, as no examples are added.
- It does not require tricky selection heuristics as in self-training algorithms.
- The supervised model can *learn* if the $\overline{\text{SLF}}$ features are relevant or not (it can ignore or downweight them if it wants).
- The constructed $\overline{\text{SLF}}$ features contain information about the potential label of an example containing these words. This is collected by averaging over many unlabeled examples hence infrequent mistakes can be smoothed out and potentially corrected on the next iteration.
- In a sequence labeling task, the $\overline{\text{SLF}}$ features for neighboring words are highly informative for the word to be labeled.
- This algorithm is highly scalable (it adds a few features to the model, not lots of extra examples).

In addition, SLF vectors give a robust abstraction of predicted class (or boundary) distributions for each word (or discrete attribute) in a large unlabeled corpus. This semi-supervsied strategy provides a SLF matrix (having size $\mathcal{D}*K$ for basic SLF where $\mathcal{D}$ is the size of the word vocabulary and K is the number of possible values for the target label). This matrix can be trained before-hand and utilized later for NLP prediction as long as the basic words are available. For example, in the case of online NER prediction, efficiency and speed are critical for online systems. Some of the features, such as part-of-speech tags, are quite slow to extract online. In this situation, we could make NER predictions relying on only basic word features (that are fast to extract, such as low-case word and capitalzation information), plus the well-trained SLF patterns from better models that incorporate harder to compute features. In the Results section we show that this setting achieves much better performance compared to sequence tagging relying only on basic features.

III. Baseline Sequence Labeling Systems

In this paper, we focus on NLP sequence labeling problems which can be treated as multi-class classification (assigning labels to words). As a wrapper approach, our method could be applied on any baseline sequence tagger. We choose two state-of-the-art systems to test our approach: (1) a deep neural network (NN) framework for unified NLP processing [14]; (2) a conditional random field model (CRF) which has been proved successful in many NLP tagging tasks.

A. Sequence Labeling with Deep Neural Network (NN)

The authors in [14] proposed a deep neural network (NN) framework for semantic role labeling in English. In this paper we adapt this framework on named entity recognition and part-of-speech tagging problems.

Traditional NLP approaches usually extract a rich set of hand-crafted features from the sentence. In contrast, this deep neural network framework provides a unified framework to handle NLP tasks and processes an input sentence by several layers of feature extraction. The features in all the layers of the network are automatically trained by backpropagation to be relevant for the target task. The first layer is a lookup-table layer which map raw words into real-valued vectors (which are learnt) for processing by subsequent layers towards the targets. The second layer

extracts features from the sentence treating it as a *sequence* with local and global structure (a sliding window approach). The remaining layers are classical NN layers. We employ one hidden layer for both NER and POS tasks.

Leveraging Unlabeled Data with an Auxillary Task: Language Model Training Semi-supervision was used in this deep NN framework through a so-called "language modeling" strategy. Relying on the abundant unlabeled text data freely available on the web, the authors [14] proposed the unsupervised LM task to model natural human language sequences (English in their case) (also deep NN based). The motivation is: for most NLP tagging tasks, words semantically similar can be usually exchanged with no impact on the labels. The proposed LM tried to force two natural sentences s^1 and s^2 with same semantic tags to have a close representation in the feature extraction layers. The features (embedding) learned by the lookup-table layer from this LM-NN essentially clusters semantically similar words. Then weights of this LM lookup-table are used to initialize the entries in word lookup-table for the target sequence labeling task, e.g. NER (which ends up similar to multi-tasking with the LM). Our experimental results show that this auxillary task is very useful for NLP sequence labeling and our SLF could improve over this semi-supervision as well. We trained two language models (one for German and the other for English, both with the Wikipedia corpus).

Viterbi Training: For sequence tagging such as NER or chunking, each entity normally involves more than one word. The unified deep NN framework described above utilizes a labeling-per-word strategy without exploring dependencies among targeted classes. For example, for each NER tag type, the popular IOBES tagging style could make totally 17 classes for sequence classification, with clear dependencies between certain tag types. In this paper, we handle the local dependency through a Viterbi-style structured prediction technique (dynamic programming), which increases the performance of this NN framework effectively. Our SLF models used *without* Viterbi have a somewhat similar function to the Viterbi, since the predicted class-distribution of the surrounding neighbor words should obey the local depencies as well, and can be used during predictions. In the section V, we compared the performance of adding SLF features when using Viterbi analysis or not, to measure how much these two techniques overlap.

SLF in Deep NN Framework: It is straighforward to apply the SLF learning in the above deep NN framework. Unlike normal word features (essentially, letter patterns), SLF vectors (except clustered SLF) could be used directly, since they are numerical values already. For all words (or discrete attributes) in the sliding text window centered at the word of interest, SLF distributions are concatenated, added to other attributes' lookup-table outputs, and then fed into the second NN layer for sequential labeling.

B. Conditional Random Field

Conditional random fields (CRFs) [15] achieve state-of-the-art performance across a broad spectrum of sequence labeling tasks, including the gene mention task in the BioCreativeII competition [13]. Linear-chain CRFs are discriminative probabilistic models over observation sequences $\mathbf{x}$ and label sequences $\mathbf{y} = (y_1, ..., y_{|\mathbf{y}|})$, where $|\mathbf{x}| = |\mathbf{y}|$, and each label y_i has K different possible discrete values (multi-class). The conditional probability is defined as,

$$p_\theta(\mathbf{y}|\mathbf{x}) = \frac{1}{Z(\mathbf{x})} \exp(\sum_j \theta_j F_j(\mathbf{x}, \mathbf{y})) \quad (9)$$

where $F_j(\mathbf{x}, \mathbf{y}) = \sum_{i=1}^{n} f_j(\mathbf{x}, y_i, y_{i+1}, i)$
and $Z(\mathbf{x}) = \sum_{\mathbf{y}} \exp(\sum_j \theta_j F_j(\mathbf{x}, \mathbf{y}))$.

The model is trained by maximizing the log-likelihood of the training data by gradient methods, which is a convex optimization problem [15]. A dynamic algorithm (the forward/backward strategy) is used to compute all the required probabilities $p_\theta(y_i, y_{i+1})$ for calculating the gradient of the likelihood.

We used the CRF++ toolkit [18]. Since CRF++ accept discrete features, clustered SLF is tested as a wrapper for CRF in this case.

IV. Related Work

A. Supervised Sequence Labeling for Natural Languages

NLP research aims to convert human language into representations that are easy for computers to manipulate. Several sub-tasks are identified as useful for end applications such as information extraction or machine translation. Typical problems range from the syntactic, like part-of-speech tagging, chunking and parsing, to the semantic, such as semantic-role labeling and named entity extraction. Most of these sub-tasks are sequence segmenatation or sequence labeling problems. While early studies were mostly based on handcrafted rules, recent tagging systems use supervised machine learning techniques to automatically train labeling algorithms from a collection of training examples. Popular supervised techniques includes Hidden Markov Models (HMM), Maximum Entropy (ME) Models, Support Vector Machines (SVM), and Conditional Random Fields (CRF) (for NER review, see [19]).

B. Semi-supervised Learning

As already mentioned, self-training [2] (also called 'bootstrapping' in the traditional NLP field) and co-training [5] augment the training set with labeled examples from the unlabeled set which are predicted by the model itself. This can give improvements to a model, but care must be taken as the predictions are prone to noise. Many other semi-supervised learning algorithms exist, including Tranductive SVMs [20], [21], graph-based regularization [22], entropy regularization [23] and EM with generative mixture models [24], see [25] for a review.

Traditionally, generative models (with EM used on the unlabeled set) were dominant for structural learning tasks in NLP. Recently discriminative models ([26], [27]) have been shown to offer competitive performance over a variety of sequential and structured learning problems in NLP. Generative models generally do not achieve the same accuracy as discriminatively trained models, and therefore it is preferable to focus on discriminative approaches [1] in this paper. A number of semi-supervised learning systems can bootstrap from small sets of labeled data using discriminative learners, including self-training, co-training, and transductive SVM [20]. None of them seems to outperform the others across different domains, and each has its prons and cons.

Self-training can be used in combination with any discriminative learning model, but the predictions are prone to noise.

Co-training [5] is a weakly supervised learning technique in which the redundancy of the task is captured by training two classifiers using separate views of the same data. Pierce et al. [28] examined the scalability behavior of co-training on NLP tasks, and found that co-training is effective for learning from small amounts of labeled data but care must be taken due to possible degradation in the quality of bootstrapped data.

Transductive SVMs [20], [21] might learn better max-margin hyperplanes with the use of unlabeled data, but their optimization procedure is non-trivial [21]. The authors of [21] proposed a practical procedure to adapt transductive SVM to large scale nonlinear cases. The largest scale tested in that paper includes just 1000 labeled examples and 60,000 unlabeled examples, which takes about 40 hours for training. Hence, apparently the scalability of this approach is limited, and impractical for the real natural language processing tasks.

Graph-based semi-supervised approaches make use of dependencies introduced between the labels of nearby examples on a constructed graph [22], [29]. These models train to encourage nearby data points to have the same class labels. They can obtain impressive performance using a very small amount of labeled data. However, since they model pairwise similarities among instances, most of these techniques require joint inference over the whole data set of size $L+U$ at test time ($O((L+U)^3)$), which is not practical for a real NLP corpus.

Entropy regularization framework [1] has recently been proposed on linear-chain Conditional Random Fields (which are popular in NLP community). In its formulation, the traditional label likelihood (on supervised data) is augmented with an additional term that encourages the model to predict low entropy label distributions on the unlabeled set. However this technique was pointed out to be quite fragile and the solution might assign all tokens the same label [30].

Most of the above semi-supervised methods adapt well to thousand of training examples (including both labeled and unlabeled data) only [31]. Except self-training and co-training, other semi-supervised algorithms have scalability problems for realistic language modeling tasks, which normally involves hundreds of thousands labeled examples. Table III lists the size of both labeled and unlabeled corpora for the four NLP tasks we tested in this work. For instance, the two CoNLL-2003 [11] NER tasks have two hundred thousand training tokens and giga-word scale unlabeled corpora (Wikipedia). In this respect, our proposed SLF method is superior to many methods since it can make use of the extremely large amount of unlabeled data in the presence of a fairly large labeled set.

Table III
NUMBER OF TOKENS USED IN FOUR NLP TASKS

We tested four tasks: two CoNLL-2003 [11]) NER tasks, one POS task [12] and one GM task [13]. Unlabeled corpus includes subset of wikipedia English, ECI German corpus and pubmed abstracts.

Tokens Size	Training (Labeled)	Testing	Unlabeled
German NER	206,931	51943	∼60M
English NER	203,621	46,435	∼200M
English POS	1,029,858	126,528	∼300M
Bio GM	345,996	116,944	∼900M

Finally, there are some methods that use auxiliary tasks on large unlabeled corpora for training sequence models (i.e. through multi-task learning). Ando and Zhang (2005) [32] proposed a method based on defining multiple tasks using unlabeled data that are multi-tasked with the task of interest, which they showed to perform very well on POS and NER tasks. The methods using auxiliary information can often find good solutions, however the selection of auxiliary problem requires significant insights. Similarly, the language model strategy proposed in Collobert et al. [14] is another type of auxillary task as above.

C. Semi-supervised Learning with Labeled Features

Beyond the significant amount of work on semi-supervised learning with small amounts of fully labeled data, there has been a growing interest in the use of background domain knowledge to augument supervised learning.

This includes a number of methods that make use of human-provided associations of features to particular classes (for example, a human annotator might label the word "cashback" to the document category "shopping"). The prior class-bias of features (called *'labeled features'* in the following) could be used to generate *pseudo labeled* examples that are then used for augumenting standard supervised learning [33], [34], [35]. This trend of work is orthogonal to the traditional semi-supervised category.

Schapire et al [33] utilized a set of features annotated with their associated majority labels to label pseudo-examples for boosting a logistic regression model. Similarly, Wu and Srihari [34] assigned labels to unlabeled documents with 'labeled features' and then use these pseudo-examples in conjunction with labeled examples to train a weighted margin Support Vector Machine with regularization. Later

Haghighi and Klein [36] explored similar "labeled features" for a "pseudo-example" strategy of training a generative MRF sequence model.

Unlike the above approaches, Druck et al. [35] utilized a generalization expectation (GE) criterion with soft constraint of the model's predictions on unlabeled instances using "labeled features". Similarly, Mann et al. [30] introduced GE to linear-chain CRFs for targeting sequence modeling problems which makes use of "labeled features" rather than "labeled instances".

In contrast to all above methods, our semi-supervised method induces features, *instead of examples*, from a large corpus of unannotated sentences in a supervised fashion. Our SLF method is orthogonal to traditional semi-supervised techniques and also orthonal to those approaches relying on "labeled features" (we do not use human annotations). As a wrapper approach, it can be combined with many of the other supervised and semi-supervised methods.

V. Experiments

A. Datasets & Settings

We first test our approach on the English and German NER datasets provided in the CoNLL-2003 shared task [11]. For each language, four files are provided, including a training file, a development file (for parameter tuning), a test file and a large file with unannotated data. For both languages, more than 200,000 training tokens exist in the provided training files (size of training/testing/unlabeled sets in Table III). The ECI data file provided in CoNLL-2003 [11] is used as our unlabeled corpus for semi-supervised learning of German NER. The English Wikipedia web pages is used for the English NER case. We then test basic SLF on the English POS data set described in [12].

For these three tasks, we used the unified deep Neural Network (NN) framework [14] as a base classifier for SLF. The first lookup-table layer maps words to 50-dimensional vectors (one vector for each word in the dictionary), the parameters of which are automatically *trained* during the learning process using backpropagation. The second layer is a classical layer of H hidden units (where H is optimized on the development set), and the final layer outputs probabilities of the class labels. The basic sequence labeling was handeled using a sliding window approach. We then extend this baseline with a Viterbi decoding of the entire sentence to incorporate local dependencies between target classes.

For the baseline deep NN model we tried various combinations of the following word features:

- For English NER, we use (i) words in a 7-word-window surrounding the current word, (ii) capitalization flags of the current and surrounding words (Caps); and (iii) gazetteer information, as provided by ConLL-2003;
- For German NER, we use (i) words in a 5-word-window surrounding the current word, (ii) capitalization flags of the current and surrounding words, (iii) prefix and suffix (length up to 4) of the current and surrounding words, (iv) the POS tags of the current and surrounding words; and (v) the chunk tag of the current and surrounding words. (Note in this case no gazetteer lists are used).
- For English POS, we use (i) words in a 5-word-window surrounding the current word, (ii) capitalization flags of the current and surrounding words, (iii) stem-ends of the current and surrounding words.

Table IV
F1 score on the test set for German NER

For each choice of baseline (left column) applying SLF improves over it (right column). LM means using language model semi-supervision.

Method	Baseline	+(Basic SLF)
Words only	45.89	**51.10**
Words only + Viterbi	50.61	**53.46**
All Features + LM	72.44	**73.32**
All Features + LM + Viterbi	74.33	**75.72**
Word only + Viterbi + best SLF	50.61	**64.41**

We then test SLF using conditional random fields (using CRF++ tool) as a base classifier on the gene-name-recognition (GM) data set (the number of tokens in training/testing/unlabeled sets in Table III) from the 2006 BioCreativeII BioNLP competition [13]. For this GM task, we utilized the following word features: (i) words in a 5-word-window surrounding current, (ii) capitalization features of current and surrounding words, (iii) prefix and suffix (up to length 4) of current and surrounding words, (iv) string patterns of current and surrounding words. (Note we did not use any gazetteer lists).

In the following, we compare basic and extended SLF models over various baselines, including supervision alone, using Viterbi decoding or semi-supervision via LM .

B. Basic SLF: Comparison with Supervised Deep NN & Semi-Supervision via LM

Table IV lists the test set performance of the German NER task using the F1 measure when applying basic SLF as a wrapper to various systems: using only word features (with and without a Viterbi decoding step), and using all features plus the language model (LM) based semi-supervised learning. In all cases SLF improves over the baseline. Our best performance of 75.72 (using all features + SLF) beats the state-of-the-art German NER performance of 75.27 which was reported in [32]. The best result during the competition was 74.17 [37].

We also considered taking our best model, and adding the SLF features predicted by it to a basic word-features only model (the last row of Table IV). This improved its accuracy from 50.61 to 64.1. Using the LM as well yields 72.45 (words+LM on their own are 69.05). This is interesting because these results do not require POS, chunk, stem or caps features any more, but are close to the state-of-the-art. As we mentioned in the section II-F, this shows that NER predictions relying on only basic word plus the well-trained

Table V
F1 SCORE ON THE TEST SET FOR ENGLISH NER

SLF improves over each baseline.

Method	Baseline	+(Basic SLF)
Words + Caps	77.82	**79.38**
Words + Caps + Viterbi	80.53	**81.51**
All Features + LM	86.49	**86.88**
All Features + LM + Viterbi	88.40	**88.69**

SLF patterns, (from better models which include harder to compute features), could achieve much better performance and this setting could be a feasible choice for online NER systems.

In addition, we also tried a transductive setting with basic SLF where we use the test set as the unlabeled corpus to build class-distribution feature vectors. The results for German NER either only help marginally, or not at all. For instance, when using "All Features+LM", this setup achieves F1 72.04 which is even slightly lower than the supervised NN baseline (F1 72.44 in Table IV). Moreover, we tried counting words' class in the train or validation sets. The resulting word-class distribution vectors could be used instead of SLF. Adding them to German NER achieves similar performance as the transductive SLF (results not shown). These results show that the semi-supervision from using the large unlabeled set in SLF is important.

Table V provides results for the English NER task. Again, SLF improves over all baselines; our best result was 88.69. In contrast, the best performing method during the competition was 88.76, and [32] have since reported 89.31 using multi-task semi-supervision. Here, our slightly worse performance seems to be due to our weaker baseline method (before even applying SLF) compared to these approaches.

C. Attribute SLF: Comparison with Supervised Deep NN & Semi-Supervision via LM

Table VI compared the performance of English POS with SLF applied to the deep NN, or not. Similar conclusions as with the two NER tasks were observed. The comparisons are under various settings (with different combinations of features added, and language model improvements). The best state-of-art [12] English POS system (as far as we know) achieved the token level error rate 2.76%. We could see that with only words, cap feature and stem-end feature, our SLF improves the deep-NN system to the state-of-the-art POS performance. The key word "attribute" in the last column means that we added not only the basic SLF feature for word but also added the attribute SLF features for 'caps' and 'stem-end' attributes. Attribute SLF makes improvements on basic SLF over both supervised baseline alone and semi-supervision LM.

D. Clustered & Boundary SLF: Comparison with CRF

The BioCreativeII GM [13] data set involves a sequence tagging task which looks for gene/protein names in a bio-literature text corpus. Since almost all the top teams in

Table VI
TOKEN LEVEL ERROR RATE OF ENGLISH POS

Attribute SLF improves over supervised baseline and semi-supervised LM. Our best POS performance achieves the state-of-the-art. (Attr: Attribute).

Method	Baseline	+(Basic SLF)	+(Attr SLF)
Words only	4.99	**4.06**	-
Words only + LM	3.93	**3.89**	-
All Features	3.28	**2.99**	**2.86**
All Features + LM	2.79	**2.75**	**2.73**

Table VII
F1 SCORE ON THE TEST SET FOR BIOCREATIVEII GM

Clustered SLF and Boundary SLF added on CRF. Both improves over the supervised CRF when using only words or all word attributes.

Method	Baseline (CRF)	+Clustered SLF
Words+Caps	82.02	**84.01** (basic)
Words+Caps	82.02	**85.24** (boundary)
All Features	86.34	**87.16** (boundary)

BioCreativeII GM competition [13] utilized CRF, we tried to use CRF on this sequence tagging problem as well. We used CRF++ as our baseline and apply clustered SLF as a wrapper semi-supervised approach. Considering there are only two classes in this tagging task (gene name or not), we tested boundary SLF on this corpus as well.

Table VII uses F1 test score to compare GM performances under multiple setups. We could see that both clustered basic SLF and clustered boundary SLF improve over CRF supervised baselines. Boundary SLF shows better improvements compared to basic SLF.

The best performance achieved here (F1 87.16 in the last row of Table VII) is slightly lower than the best team (87.21 F1 test) in the BioCreativeII GM competition. Here we used only basic word, "caps" features, string patterns and prefix/suffix attributes. All top teams in the competition used many other word attributes, such as POS, and they all utilized other complex techniques such as the bidirectional training of models and extensive domain lexicons. In summary, SLF could improve CRF on this GM corpus effectively and we could achieve state-of-the-art performance using only basic word features.

E. Comparison with Self-training

We applied self-training to the same baseline methods to compare to the performance of SLF. There are numerous variants of self-training. We adopt the following weighting scheme: given L training examples, we choose L/R (R is a parameter to choose) unlabeled examples to add in the next round's training. By varying R, we get a range of impacts from self-training.

Table VIII and Table IX give results of self-training for the English and German NER tasks. Table X provides the results for English POS. For all three corpus, self-training only helped marginally, or not at all, depending on the parameters.

The above comparison indicates that SLF has better behavior than self-training with a random selection strategy. Since there exist many selection strategies for self-training, other selection techniques might bring improvements, see

Table VIII
TEST F1 OF GERMAN NER USING SELF-TRAINING.

Method	Baseline	R=1	R=10	R=100
Words only	**50.61**	47.07	47.92	47.9
All+LM	74.33	73.42	**74.41**	73.9

Table IX
TEST F1 OF ENGLISH NER USING SELF-TRAINING.

Method	Baseline	R=1	R=20	R=100
Words only	80.53	79.51	**81.01**	80.85
All+LM	**88.40**	87.64	88.07	88.17

Table X
ENGLISH POS USING SELF-TRAINING.

Method	Baseline	R=1	R=20	R=100
Words only	**4.99**	5.08	5.10	5.12
All+LM	**2.79**	**2.79**	2.80	2.81

e.g. [9], [7] for other strategies. Still, these heuristic choices are difficult and need careful tuning [7]. In contrast, the proposed SLF method does not seem to suffer from these issues.

Further, the performance in multiple rounds of self-training might oscillate because of degradation by noisy labels (see e.g. [7], [8]). We observed that basic SLF's iterative training gives stable results. Figure 2 shows the test F1 from the iterations (as a wrapper for the "All features + LM + Viterbi" baseline) for the German NER set. It appears to converge in only a few iterations.

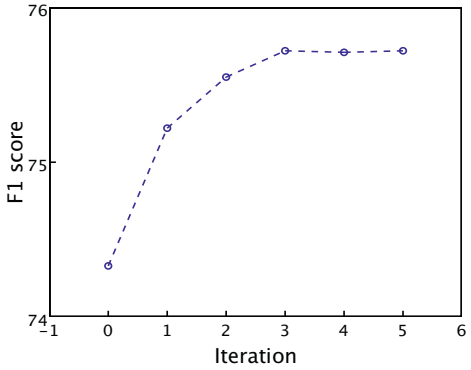

Figure 2. Test F1 over basic SLF rounds on German NER.

VI. CONCLUSIONS

In this work we proposed a novel semi-supervised algorithm for learning features from a large unlabeled corpus in a supervised fashion. These features try to model class-distribution patterns for words or word attributes. We applied this method and several extensions to four classical labeling tasks, where we obtained improvements over the supervised and semi-supervised baselines tested. Our method is highly scalable, contains no difficult parameters to tune, and we found it to be empirically robust.

The proposed method can easily be extended to other cases or domains. For example, instead of calculating predicted class distributions for each word, we could consider n-gram distributions instead. Moreover, one can generalize beyond word-level evaluation tasks. For instance in text categorization problems (document classification or sentiment analysis) a word's class distribution is the distribution of labels of *documents* that contained that word.

REFERENCES

[1] F. Jiao, S. Wang, C.-H. Lee, R. Greiner, and D. Schuurmans, "Semi-supervised conditional random fields for improved sequence segmentation and labeling," in *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics.*, 2006, pp. 209–216.

[2] H. Scudder, "Probability of error of some adaptive pattern-recognition machines," *IEEE Transactions on Information Theory*, vol. 11, no. 3, pp. 363–371, 1965.

[3] R. J. Kate and R. J. Mooney, "Semi-supervised learning for semantic parsing using support vector machines," in *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics, Short Papers (NAACL/HLT-2007)*, 2007, pp. 81–84.

[4] Z. Kozareva, B. Bonev, and A. Montoyo, "Self-training and co-training applied to spanish named entity recognition," in *MICAI 2005: Advances in Artificial Intelligence*, 2005.

[5] A. Blum and T. Mitchell, "Combining labeled and unlabeled data with co-training," in *COLT' 98:*. New York, NY, USA: ACM, 1998, pp. 92–100.

[6] M. Collins and Y. Singer, "Unsupervised models for named entity classification," in *Proceedings of the Joint SIGDAT Conference on EMNLP*, 1999, pp. 100–110.

[7] H. Shan and D. Gildea, "Self-training and co-training for semantic role labeling: Primary report," University of Rochester, Comp. Sci. Dept., Tech. Rep. TR891, 2006.

[8] T. Zhang, F. Damerau, and D. Johnson, "Text chunking using regularized winnow," in *ACL '01: Proceedings of the 39th Annual Meeting on Association for Computational Linguistics*, Morristown, NJ, USA, 2001, pp. 539–546.

[9] H. Daumé III, "Cross-task knowledge-constrained self training," in *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, Honolulu, Hawaii, October 2008, pp. 680–688.

[10] Y. Qi, R. Collobert, P. Kuksa, K. Kavukcuoglu, and J. Weston, "Combining labeled and unlabeled data for word-class distribution learning," in *CIKM'09: Proceedings of the eighth ACM Conference on Information and Knowledge Management*. ACM, 2009.

[11] Erik and F. De Meulder, "Introduction to the conll-2003 shared task: language-independent named entity recognition," in *Proceedings of the seventh conference on Natural language learning at HLT-NAACL 2003*, 2003, pp. 142–147.

[12] K. Toutanova, D. Klein, C. D. Manning, and Y. Singer, "Feature-rich part-of-speech tagging with a cyclic dependency network," in *NAACL '03: Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology*, Morristown, NJ, USA, 2003, pp. 173–180.

[13] L. Smith, L. Tanabe, R. Ando, et al., and J. W. Wilbur, "Overview of biocreative ii gene mention recognition," *Genome Biology*, vol. 9, no. Suppl 2, 2008.

[14] R. Collobert and J. Weston, "A unified architecture for nlp: deep neural networks with multitask learning," in *ICML '08: Proceedings of the 25th international conference on Machine learning*, 2008, pp. 160–167.

[15] J. Lafferty, A. McCallum, and F. Pereira, "Conditional random fields: Probabilistic models for segmenting and labeling sequence data," in *ICML'01:*, 2001.

[16] C. D. Manning and H. Schtze, *Foundations of Statistical Natural Language Processing*. MIT press, 1999.

[17] A. Gersho and R. M. Gray, *Vector quantization and signal compression*. Norwell, MA, USA: Kluwer Academic Publishers, 1991.

[18] "Crf++: Yet another crf toolkit." [Online]. Available: http://crfpp.sourceforge.net/

[19] D. Nadeau and S. Sekine, "A survey of named entity recognition and classfication," *Linguisticae Investigationes*, vol. 30, no. 1, pp. 3–26, 2007.

[20] T. Joachims, "Transductive inference for text classification using support vector machines," in *ICML '99: Proceedings of the Sixteenth International Conference on Machine Learning*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1999, pp. 200–209.

[21] R. Collobert, F. Sinz, J. Weston, and L. Bottou, "Large scale transductive svms," vol. 7. Cambridge, MA, USA: MIT Press, 2006, pp. 1687–1712.

[22] X. Zhu, Z. Ghahramani, and J. Lafferty, "Semi-supervised learning using gaussian fields and harmonic functions," in *ICML'03: Proceedings of the 20th International Conference on Machine Learning*, 2003, pp. 912–919.

[23] Y. Grandvalet and Y. Bengio, "Semi-supervised learning by entropy minimization," in *Advances in Neural Information Processing Systems 17*, L. K. Saul, Y. Weiss, and L. Bottou, Eds. Cambridge, MA: MIT Press, 2005, pp. 529–536.

[24] K. Nigam, A. K. McCallum, S. Thrun, and T. Mitchell, "Text classification from labeled and unlabeled documents using em," vol. 39, no. 2-3. Hingham, MA, USA: Kluwer Academic Publishers, 2000, pp. 103–134.

[25] O. Chapelle, B. Schölkopf, and A. Zien, Eds., *Semi-Supervised Learning (Adaptive Computation and Machine Learning)*. MIT Press, 2006.

[26] H. Daumé III, "Semi-supervised or semi-unsupervised?" in *Proceedings of the NAACL HLT Workshop on Semi-supervised Learning for Natural Language Processing*, vol. Invited Position Paper,, no. 84-85, Boulder, Colorado, USA, June 2009, pp. 19–27.

[27] S. Dasgupta and V. Ng, "Discriminative models for semi-supervised natural language learning", invited position paper," in *Proceedings of the NAACL HLT Workshop on Semi-supervised Learning for Natural Language Processing*, Boulder, Colorado, USA, June 2009, pp. 84–85.

[28] D. Pierce and C. Cardie, "Limitations of co-training for natural language learning from large datasets," in *In Proceedings of the 2001 Conference on Empirical Methods in Natural Language Processing*, 2001, pp. 1–9.

[29] A. Blum, "Semi-supervised learning using randomized mincuts," in *ICML '04: Proceedings of the twenty-first international conference on Machine learning*, 2004.

[30] G. Mann and A. McCallum, "Generalized expectation criteria for semi-supervised learning of conditional random fields," in *ACL-08: Association for Computational Linguistics.*, 2008, pp. 870–878.

[31] A. Goldberg and X. Zhu, "Keepin' it real: Semi-supervised learning with realistic tuning," in *Proceedings of the NAACL HLT Workshop on Semi-supervised Learning for Natural Language Processing*, Boulder, Colorado, USA, June 2009, pp. 19–27.

[32] R. K. Ando and T. Zhang, "A framework for learning predictive structures from multiple tasks and unlabeled data," *Journal of Machine Learning Research*, vol. 6, pp. 1817–1853, 2005.

[33] R. E. Schapire, M. Rochery, M. G. Rahim, and N. Gupta, "Incorporating prior knowledge into boosting," in *ICML '02: Proceedings of the Nineteenth International Conference on Machine Learning*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2002, pp. 538–545.

[34] X. Wu and R. Srihari, "Incorporating prior knowledge with weighted margin support vector machines," in *KDD '04: Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*. New York, NY, USA: ACM, 2004, pp. 326–333.

[35] G. Druck, G. Mann, and A. McCallum, "Learning from labeled features using generalized expectation criteria," in *SIGIR '08: Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, 2008, pp. 595–602.

[36] A. Haghighi and D. Klein, "Prototype-driven learning for sequence models," in *Proceedings of the main conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics*, Morristown, NJ, USA, 2006, pp. 320–327.

[37] R. Florian, A. Ittycheriah, H. Jing, and T. Zhang, "Named entity recognition through classifier combination," in *Proceedings of the seventh conference on Natural language learning at HLT-NAACL 2003*, 2003, pp. 168–171.