An Experimental Comparison of Active Learning Strategies for Partially Labeled Sequences

Diego Marcheggiani
Istituto di Scienza e Tecnologie dell'Informazione
Consiglio Nazionale delle Ricerche
Pisa, Italy
diego.marcheggiani@isti.cnr.it

Thierry Artières
LIP6
Pierre et Marie Curie University
Paris, France
thierry.artieres@lip6.fr

Abstract

Active learning (AL) consists of asking human annotators to annotate automatically selected data that are assumed to bring the most benefit in the creation of a classifier. AL allows to learn accurate systems with much less annotated data than what is required by pure supervised learning algorithms, hence limiting the tedious effort of annotating a large collection of data.

We experimentally investigate the behavior of several AL strategies for sequence labeling tasks (in a *partially-labeled* scenario) tailored on Partially-Labeled Conditional Random Fields, on four sequence labeling tasks: phrase chunking, part-of-speech tagging, named-entity recognition, and bio-entity recognition.

1 Introduction

Today, the state-of-the-art methods in most natural language processing tasks are supervised machine learning approaches. Their main problem lies in their need of large human-annotated training corpus, which requires a tedious and expensive work from domain experts. The process of *active learning* (AL) employs one or more human annotators by asking them to label new samples which are supposed to be the most informative in the creation of a new classifier. A classifier is incrementally retrained with all the data labeled by the annotator. AL has been demonstrated to work well and to produce accurate classifiers while saving much human annotation effort. One critical issue is to define a measure of the informativeness which should reflect how much new information a new example would give in the learning of a new classifier once annotated.

A lot of work has been done on the AL field in the past years (see (Settles, 2012) for an exhaustive overview). In particular, AL proved its usefulness in sequence labeling tasks (Settles and Craven, 2008). Yet, researchers have always adopted as annotation unit an entire sequence (i.e., the annotator is asked to annotate the whole sequence) while it looks like it could be much more relevant to ask for labeling only small parts of it (e.g., the ones with highest ambiguity). A few works have investigated this idea. For instance, Wanvarie et al. (2011) proposed to use Partially-Labeled Conditional Random Fields (PL-CRFs) (Tsuboi et al., 2008), a semi-supervised variation of Conditional Random Fields (CRFs) (Lafferty et al., 2001) able to deal with partially-labeled sequences, thus enabling to adopt as annotation unit single tokens and still learning from full sequences. AL with partially labeled sequences has proven to be effective in substantially reducing the amount of annotated data with respect to common AL approaches (see (Wanvarie et al., 2011)).

In this work we focus on AL strategies for partially labeled sequences adopting the single token as annotation unit and PL-CRFs as learning algorithm given its nature in dealing with partially labeled sequences. We propose several AL strategies based on measures of *uncertainty* adapted for the AL with partially labeled sequences scenario and tailored on PL-CRFs. We further propose two strategies that exploit the finer granularity given by the partially-labeled scenario. We also show that the choice of single-token annotation can bring to unpredictable results on sequence labeling tasks in which the structure of the sequences is not regular, e.g., named-entity recognition. We propose a first solution to the problem of unpredictability. The aim of this work is thoroughly compare the effectiveness and the behavior of all the proposed AL strategies on four standard sequence labeling tasks, phrase chunking, part-of-speech tagging, named-entity recognition and bio-entity recognition.

The remainder of this paper is as follows. In Section 2 we summarize the related work in AL, in Section 3 we describe PL-CRFs, the semi-supervised algorithm we adopt in this work. Section 4 describes in details the AL framework and the AL strategies we propose. Section 5 provides a description of the experimental setting, the datasets, and discusses the empirical results. Section 6 summarizes our findings.

2 Related Work

Our work belongs to the pool-based AL framework. It considers the case in which a large amount (pool) of unlabeled examples is available, from which samples to be labeled must be chosen. This framework fits all the sequence labeling problems we consider here. For a more exhaustive survey on other AL frameworks see

Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), pages 898–906,
October 25-29, 2014, Doha, Qatar.

(Settles, 2012).

Most of the AL works on sequence labeling adopted the entire sequence as annotation unit (Settles and Craven, 2008) which was demonstrated by Wanvarie et al. (2011) to be less effective than using the single token as annotation unit. The main AL works in this latter line of work are (Shen et al., 2004), (Tomanek and Hahn, 2009) and (Wanvarie et al., 2011). Shen et al. (2004) adopted SVMs as learning algorithm and proposed two strategies that combine three criteria, informativeness, representativeness and diversity. SVMs allowed them to use as annotation unit a subset of the tokens in a sequence, without annotating, in any way, the rest of the tokens in the sequence. In (Tomanek and Hahn, 2009), the most uncertain tokens of the sequence are singularly annotated, but the rest of the labels in the sequence are then chosen by the classifier in a semi-supervised fashion. Wanvarie et al. (2011) is the closest work to ours, they adopt a minimum confidence selection strategy with re-estimation using the PL-CRFs. Differently from our work, Wanvarie et al. (2011) show that adopting the AL with partially labeled sequences using re-estimation, the annotation cost can be dramatically reduced (by annotating from 8% to 10% of the tokens of the entire training set), obtaining the same level of performance of the classifier trained on the entire, fully-labeled, training set. We started our work from this conclusion and we focused on AL with partially labeled sequences using re-estimation by comparing several AL strategies in order to find the strategy that allows to create the best classifier with the minimum annotation effort.

3 Partially-Labeled Conditional Random Fields

Nowadays, CRFs are the *de-facto* standard for the solution of sequence labeling tasks (Sarawagi, 2008). In traditional CRFs (Lafferty et al., 2001) the conditional probability of a sequence of labels $\mathbf{y}$ given a sequence of observed feature vectors $\mathbf{x}$ is given by:

$$p(\mathbf{y}|\mathbf{x}) = \frac{1}{Z(\mathbf{x})} \prod_{t=1}^{T} \Psi_t(\mathbf{y}, \mathbf{x}) \tag{1}$$

where a standard choice for sequence labeling tasks are the so called Linear-chain CRFs:

$$p(\mathbf{y}|\mathbf{x}) = \frac{1}{Z(\mathbf{x})} \prod_{t=1}^{T} \Psi_t(y_t, y_{t-1}, \mathbf{x}_t) \tag{2}$$

with:

$$\Psi_t(y_t, y_{t-1}, \mathbf{x}_t) = \Psi_u(y_t, x_t)\Psi_b(y_t, y_{t-1}) \tag{3}$$

where $\Psi_u(y_t, x_t)$ models the co-occurrence between features x_t, and label y_t at time t, and $\Psi_b(y_t, y_{t-1})$ models the co-occurrence between two adjacent labels y_t and y_{t-1}.

PL-CRFs introduced by Tsuboi et al. (2008) allow to learn a CRF model using partially-labeled sequences, marginalizing on those tokens that do not have an assigned label. In PL-CRFs, $\mathbf{L}$ denotes a partially labeled information about a sequence. It consists of a sequence of sets L_t in which $L_t = \mathcal{Y}$ (where $\mathcal{Y}$ is the set of all the possible labels) if there is no label information for token at time t. L_t is a singleton containing y_t if the label of the token at time t is known, and $\mathbf{Y_L}$ is the set of label sequences that fits the partial label information $\mathbf{L}$. Then the probability of a partial labeling may be computed as:

$$p(\mathbf{Y_L}|\mathbf{x}) = \sum_{\mathbf{y} \in \mathbf{Y_L}} p(\mathbf{y}|\mathbf{x}) \tag{4}$$

In order to perform inference and parameter learning on PL-CRFs, some modifications on traditional CRFs inference algorithms are required.

3.1 Forward-Backward Algorithm

Differently from traditional CRFs, the forward and backward scores (respectively α and β), are calculated as follows:

$$\alpha_{t,\mathbf{L}}(j) = \begin{cases} 0 & \text{if } j \notin L_t \\ \Psi_1(j, y_0, x_1) & \text{else if } t = 1 \\ & \text{and } j \in L_t \\ SA(j) & \text{otherwise} \end{cases} \tag{5}$$

$$\beta_{t,\mathbf{L}}(i) = \begin{cases} 0 & \text{if } j \notin L_t \\ 1 & \text{else if } t = T \\ & \text{and } j \in L_t \\ SB(j) & \text{otherwise} \end{cases} \tag{6}$$

where

$$SA(j) = \sum_{i \in L_{t-1}} \alpha_{t-1,\mathbf{L}}(i)\Psi_t(j, i, x_t) \tag{7}$$

$$SB(j) = \sum_{j \in L_{t+1}} \beta_{t+1,\mathbf{L}}(j)\Psi_{t+1}(j, i, x_{t+1}) \tag{8}$$

and y_0 is a special label that encodes the beginning of a sequence.

3.2 Marginal Probability

The marginal probability $p(y_t = j|\mathbf{x}, \mathbf{L})$ is calculated as:

$$p(y_t = j|\mathbf{x}, \mathbf{L}) = \frac{\alpha_{t,\mathbf{L}}(j) \cdot \beta_{t,\mathbf{L}}(j)}{Z_{\mathbf{L}}(\mathbf{x})} \tag{9}$$

with:

$$\forall t, Z_{\mathbf{L}}(\mathbf{x}) = \sum_{j \in L_t} \alpha_{t,\mathbf{L}}(j) \cdot \beta_{t,\mathbf{L}}(j) \tag{10}$$

In case there is no label information, the formulas for forward and backward scores (Equations (5) and (6)) and for the marginal probabilities (Equation (9)) yield the standard results of CRFs.

3.3 Viterbi Algorithm

The most probable sequence assignment may be derived with a Viterbi algorithm by recursively computing the following quantities:

$$\delta_{t,\mathbf{L}}(j) = \begin{cases} 0 & \text{if } j \notin L_t \\ \Psi_1(j, y_0, x_1) & \text{else if } t = 1 \\ & \text{and } j \in L_t \\ M(j) & \text{otherwise} \end{cases} \quad (11)$$

where

$$M(j) = \max_{i \in L_{t-1}} \delta_{t-1,\mathbf{L}}(i) \Psi_t(j, i, x_t) \quad (12)$$

The most probable assignment is then calculated as: $\mathbf{y}^* = argmax_{\mathbf{y}} p(\mathbf{y}|\mathbf{x}, \mathbf{L})$

3.4 Log-Likelihood

PL-CRFs's parameters θ are learnt through maximum log-likelihood estimation, that is to maximize the log-likelihood function $LL(\theta)$:

$$\begin{aligned} LL(\theta) &= \sum_{i=1}^{N} \log p(\mathbf{Y}_{\mathbf{L}^{(i)}}|\mathbf{x}^{(i)}) \\ &= \sum_{i=1}^{N} \log Z_{\mathbf{Y}_{L^{(i)}}}(\mathbf{x}^{(i)}) - \log Z_{\mathbf{Y}}(\mathbf{x}^{(i)}) \end{aligned} \quad (13)$$

The parameters θ that maximize Equation (13) are computed via the LBFGS optimization method (Byrd et al., 1994).

4 Active Learning Strategies

Pool-based AL (see (Lewis and Catlett, 1994)) is probably the most common scenario in AL, where one has a large amount (pool) of unlabeled examples $\mathcal{U}_1$ and a small amount of labeled examples $\mathcal{T}_1$. In this scenario, the process of AL consists in a series of n iterations where a classifier Φ_i is trained with labeled examples $\mathcal{T}_i$, and then is used to classify the unlabeled examples $\mathcal{U}_i$. At this point an AL strategy $\mathcal{S}$ will select a number of examples B that once labeled will hopefully improve the performance of the next classifier Φ_{i+1}.

Algorithm 1 shows the pool-based AL framework for partially annotated sequences as introduced in (Wanvarie et al., 2011). Differently from AL for fully labeled sequences (Esuli et al., 2010), thanks to the finer granularity of the partially labeled model, we use the token as basic annotation unit, instead of the entire sequence.

The point of using the partial labeling is in saving the request for human annotations on tokens whose labels are already known (inferred) by the classifier and concentrate on those tokens that the classifier finds hard to label. Using the semi-supervised approach of the PL-CRFs we can take advantage of single-labeled tokens instead of an entire labeled sequence.

The entire pool-based AL process with partially labeled sequences is summarized in Algorithm 1. The

Algorithm 1 Pool-based active learning framework

Require: $\mathcal{T}_1$, the initial training set
$\mathcal{U}_1$, the initial unlabeled set
$\mathcal{S}$, the selected AL strategy
n, the number of iterations
B, the dimension of the update batch

1: **for** $i \leftarrow 1$ to n **do**
2: $\quad \Phi_i \leftarrow \text{train}(\mathcal{T}_i)$
3: $\quad \mathcal{L}_i \leftarrow \Phi_i(\mathcal{U}_i)$
4: $\quad$ **for** $b \leftarrow 1$ to B **do**
5: $\quad\quad \mathbf{x}_*^{(b)} \leftarrow \arg\min_{\mathbf{x}_t \in \mathbf{x}, \mathbf{x} \in \mathcal{L}_i} \mathcal{S}(t, \mathbf{x})$
6: $\quad\quad \mathcal{L}_i \leftarrow \mathcal{L}_i - \mathbf{x}^{(b)} \cup \Phi_i(\mathbf{x}^{(b)}, y_*)$
7: $\quad\quad \mathcal{U}_i \leftarrow \mathcal{U}_i - \mathbf{x}_*^{(b)} \cup (\mathbf{x}_*^{(b)}, y_*)$
8: $\quad\quad \mathcal{T}_i \leftarrow \mathcal{T}_i - \mathbf{x}_*^{(b)} \cup (\mathbf{x}_*^{(b)}, y_*)$
9: $\quad \mathcal{U}_{i+1} \leftarrow \mathcal{U}_i$
10: $\quad \mathcal{T}_{i+1} \leftarrow \mathcal{T}_i$

function $\mathcal{S}(t, \mathbf{x})$ is what, hereafter, we call an AL *strategy*. $\mathcal{S}(t, \mathbf{x})$ takes as input an automatically annotated sequence $\mathbf{x}$ and an element t of this sequence, from the set of sequences $\mathcal{L}_i$ annotated by the PL-CRF classifier Φ_i, and returns a measure of informativeness as a function of the classifier decision.

For each iteration through the update batch B, the most informative element $\mathbf{x}_*^{(b)}$, according to the AL strategy, is chosen. The subscript $*$, in this case, represents the most informative token, while the superscript (b) represents the sequence in which the token appears. After the choice of the most informative token the sets $\mathcal{L}_i$, $\mathcal{U}_i$ and $\mathcal{T}_i$ are updated. $\mathcal{L}_i$ is updated by removing the annotated sequence $\mathbf{x}^{(b)}$ and all the information given by the classifier, and by adding the same sequence with the new manually labeled token (y_*) and all the re-estimated annotation given by the classifier $\Phi_i(\mathbf{x}^{(b)}, y_*)$. In the unlabeled set $\mathcal{U}_i$ and the training set $\mathcal{T}_i$ the most informative token $\mathbf{x}_*^{(b)}$ is updated with its manually labeled version $(\mathbf{x}_*^{(b)}, y_*)$[1]. After B token annotations, the unlabeled set and the training set for the next iteration, respectively $\mathcal{U}_{i+1}$ and $\mathcal{T}_{i+1}$, are updated.

The inference methods of Section 3 allow not only to train a CRF model with partially labeled sequences, but give the possibility of classifying partially labeled sequences, using the known labels as support for the prediction of the other ones. Thus, in this AL scenario, each time a token is chosen it is immediately labeled, and this new information, as we can see from line 6 of Algorithm 1, is promptly used to re-estimate the informativeness of the other tokens in the sequence in which the chosen token appears.

One may argue that, for a human annotator, anno-

[1] In order to have a light notation we omit the fact that when the most informative token is the first annotated token of a sentence, the whole sentence, with just one annotated token, is added to the training set $\mathcal{T}_i$

tating only one or few tokens, instead of the entire sequence, is a difficult task. This would be correct in the scenario in which the text is presented to the human annotator without any visual clue about the annotations. However, in (Culotta and McCallum, 2005) it is shown that presenting to the human annotator the highlighted sequence to be annotated along with the associated sequence of labels obtained by the classifier requires much less effort from the annotator than performing the annotation without any visual and contextual clue.

4.1 Greedy Strategies

In this section we present three AL strategies that select the most informative tokens, regardless of the assignment performed by the Viterbi algorithm. The rationale behind these strategies is that, even though we are looking for the most probable sequence assignment, we also want to annotate the most informative tokens singularly.

The **Minimum Token Probability (MTP)** strategy employs as measure of informativeness the probability of the most probable assignment at time t. This strategy greedily samples the tokens whose highest probability among the labels is lowest.

$$\mathcal{S}^{MTP}(t, \mathbf{x}) = \max_{j \in \mathcal{Y}} p(y_t = j | \mathbf{x}, \mathbf{L}) \tag{14}$$

The **Maximum Token Entropy (MTE)** strategy relies on the entropy measure to evaluate the ambiguity about the label of a token. The rationale of it is that, if more than one label have the same assigned marginal probability, the entropy will be high, that is,

$$\mathcal{S}^{MTE}(t, \mathbf{x}) = \sum_{j \in \mathcal{Y}} p(y_t = j | \mathbf{x}, \mathbf{L}) \cdot \log p(y_t = j | \mathbf{x}, \mathbf{L}) \tag{15}$$

In order to directly plug the $\mathcal{S}^{MTE}$ strategy into the AL framework of Algorithm 1, we removed the minus sign at the beginning of the entropy formula. This allow us to use the $\min$ operator with a maximum entropy approach.

The **Minimum Token Margin (MTM)** strategy is a variant of the margin sampling strategy introduced in (Scheffer et al., 2001). It calculates the informativeness by considering the two most probable assignments and by subtracting the highest probability by the lowest. With $\max'$ that calculates the second maximum value, MTM is defined as:

$$\mathcal{S}^{MTM}(t, \mathbf{x}) = \max_{j \in \mathcal{Y}} p(y_t = j | \mathbf{x}, \mathbf{L}) - \max_{j \in \mathcal{Y}}{}' p(y_t = j | \mathbf{x}, \mathbf{L}) \tag{16}$$

4.2 Viterbi Strategies

The following AL strategies take into consideration the most probable sequence assignments obtained from the Viterbi algorithm computed on already known labels in the sequence.

The rationale is that, with these strategies, the measure of uncertainty is chosen according to the information obtained from the outcome of the Viterbi algorithm (i.e., the most probable sequence assignment).

The **Minimum Viterbi Probability (MVP)** is the base strategy adopted in (Wanvarie et al., 2011). It takes as measure of informativeness the probability of the label chosen by the Viterbi algorithm.

$$\mathcal{S}^{MVP}(t, \mathbf{x}) = p(y_t^* | \mathbf{x}, \mathbf{L}) \tag{17}$$

where y_t^* is the label assignment chosen by the Viterbi algorithm. In general, the token assignments that maximize the probability of the sequence assignment y_t^* are different from the token assignments that maximize the probability of the individual token assignments $argmax_{j \in \mathcal{Y}} p(y_t = j)$.

The **Maximum Viterbi Pseudo-Entropy (MVPE)** strategy calculates for each token the "pseudo" entropy of the most probable sequences at the variation of the label at position t. The prefix pseudo is used because even though it is calculated as an entropy, the summation is over all the possible labels that can be associated to a token, and not all the possible sequence assignments.

$$\mathcal{S}^{MVPE}(t, \mathbf{x}) = \sum_{j \in \mathcal{Y}} p(\mathbf{y}^*_{y_t=j} | \mathbf{x}, \mathbf{L}) \cdot \log p(\mathbf{y}^*_{y_t=j} | \mathbf{x}, \mathbf{L}) \tag{18}$$

where $\mathbf{y}^*_{y_t=j}$ represents the most probable assignment with the label at time t constrained to the value j. As in the MTE strategy the minus sign is removed in order to plug the functions directly into the AL framework of Algorithm 1.

The **Minimum Viterbi Margin (MVM)** strategy calculates the difference of the sequence probabilities of the two most probable sequence assignments at the variation of the label at time t. When the difference at time t is low, the Viterbi algorithm, in that time, chooses between two almost equally probable, sequence assignments. Formally:

$$\mathcal{S}^{MVM}(t, \mathbf{x}) = p(\mathbf{y}^*_{y_t^*} | \mathbf{x}, \mathbf{L}) - p(\mathbf{y}'^*_{y_t'^*} | \mathbf{x}, \mathbf{L}) \tag{19}$$

where $\mathbf{y}'^*$ is the second most probable assignment.

PL-CRFs allow us to inspect one token at time in order to decide if it is worth to annotate. This fact give us the possibility of exploit two quantities in order to estimate the informativeness of a token, the sequence probability, usually adopted in the traditional AL for sequence labeling, and the marginal probabilities of the single tokens as in Section 4.1. The **Minimum Expectation (ME)** strategy combines the marginal probabilities, $p(y_t = j | \mathbf{x}, \mathbf{L})$ and $p(\mathbf{y}^*_{y_t=j} | \mathbf{x}, \mathbf{L})$.

$$\mathcal{S}^{ME}(t, \mathbf{x}) = \sum_{j \in \mathcal{Y}} p(y_t = j | \mathbf{x}, \mathbf{L}) \cdot p(\mathbf{y}^*_{y_t=j} | \mathbf{x}, \mathbf{L}) \tag{20}$$

Here the maximum sequence probability is seen as a function, and what we calculate is the expected value of this very function. The rationale of this strategy is picking those tokens in which both, the sequence probability returned by the Viterbi algorithm, and the marginal probability of the considered labels are low.

Given that the ME strategy gives a high weight to the sequence probability, one might expect that tokens that belongs to longer sequences are selected more frequently, given that, the sequence probability of longer sequences is usually lower than shorter ones. One way to normalize this difference is subtracting the current maximum sequence probability, that is, the maximum sequence probability calculated without any new label estimation, to the expected value obtained from the estimation of the label assignment of the token. This is the **Minimum Expectation Difference (MED)** strategy.

$$\mathcal{S}^{MED}(t, \mathbf{x}) = \mathcal{S}^{ME}(t, \mathbf{x}) - p(\mathbf{y}^*|\mathbf{x}, \mathbf{L}) \qquad (21)$$

The rationale of this strategy is that when the expected value is far from the maximum value, that is the value returned by the Viterbi algorithm, it means that we have uncertainty on the token taken into consideration.

The **Random (RAND)** strategy samples random tokens without any external information. It is used as *baseline* to compare the real effectiveness of the proposed strategy.

At the best of our knowledge the strategies presented in this section (with the exception of the MVP strategy) have never been applied in the context of AL with partially labeled sequences scenario.

5 Experiments

5.1 Datasets

We have experimented and evaluated the AL strategies of Section 4 on four sequence labeling tasks, part-of-speech tagging, phrase chunking, named-entity recognition and bio-entity recognition. We used the CoNLL2000 dataset (Tjong Kim Sang and Buchholz, 2000) for the phrase chunking task, the CoNLL2003 dataset (Tjong Kim Sang and De Meulder, 2003), for the named-entity recognition task, the NLPBA2004 dataset (Kim et al., 2004), for the biomedical entity recognition task and the CoNLL2000POS dataset[2] for the part-of-speech labeling task. All the datasets are publicly available and are standard benchmarks in sequence labeling tasks. Table 1 shows some statistics of the datasets in terms of dimensions, number of labels, distribution of the labels, etc. The data heterogeneity of the different datasets allowed us to test the AL strategies on different "experimental settings", thus to have a more robust empirical evaluation.

[2]This is the CoNLL2000 dataset annotated with part-of-speech labels instead of chunking labels.

5.2 Experimental Setting

We tested the AL strategies described in Section 4 on test sets composed by 2012 sequences and 47377 tokens for the CoNLL2000 and CoNLL2000POS datasets, by 3452 sequences and 46394 tokens for the CoNLL2003 dataset and by 3856 sequences and 101039 tokens for the NLPBA2004 dataset. We chose an initial training set $\mathcal{T}_1$ of $\sim$5 sequences on CoNLL2000 and CoNLL2000POS datasets, $\sim$7 sequences on CoNLL2003 dataset and $\sim$4 sequences on NLPBA2004 dataset, for a total of $\sim$100 labeled tokens for each dataset. The dimension of the batch update B has been chosen as a trade-off between an ideal case in which the system is retrained after every single annotation (i.e., $B = 1$) and a practical case with higher B to limit the algorithmic complexity (since the PL-CRF classifier must be retrained every iteration). We used in our experiments $B = 50$. We fixed the number of AL iterations n at 40 because what matters here is how the strategies behave in the beginning of AL process when the annotation effort remains low. For each strategy and for each dataset, we report averaged results of three runs with a different randomly sampled initial training set $\mathcal{T}_1$.

For each dataset we adopted a standard set of features. For the CoNLL2000 dataset we adopted the same standard features used in (Wanvarie et al., 2011) for the same dataset, for the CoNLL2003 and the NLPBA2004 dataset we adopted the features used in (Wanvarie et al., 2011) for the CoNLL2003 dataset, while for the CoNLL2000POS dataset we used the features presented in (Ratnaparkhi, 1996). As evaluation measure we adopted the token variant of the F_1 measure, introduced by Esuli and Sebastiani (2010). This variant, instead of the entire annotation (chunk/entity), calculates TPs, FPs, and FNs, singularly for each token that compose the annotation, bringing to a finer evaluation.

5.3 Results

From the learning curves of Figure 1 and Figure 2 it is clear that most of the strategies have the same trend throughout the different datasets. This results is somewhat different from the results obtained in (Settles and Craven, 2008) in which there is not a clear winner among the strategies they proposed in a fully-labeled scenario. The strategies that perform particularly bad (worse than the RAND strategy in CoNLL2000POS and in CoNLL2003 dataset) in all the datasets are the MTE and MTP. This is expected, because the choice of the measure of informativeness related to the token without taking in consideration the Viterbi path is suboptimal in this task. Surprisingly, the MTM strategy even though based on the same principle of MTE and MTP, is very effective in most of the datasets. The most effective strategies, that is, the ones that are the faster at helping the classifier to reach a better accuracy are the MTM, MVM, and MVP, in particular the margin-based strategies perform very good in all the

Table 1: Training Data Statistics. #S is the number of total sequences in the dataset, #T is the number of tokens in the dataset, #L is the number of positive labels (labels different from the negative label O), AAL is the average length, in tokens, of annotations (sequence of tokens that refer to the same instance of a label), APT is the average number of token in a sequence annotated with a positive label, ASL is the average length of a sequence, AA is the average number of annotations in a sequence, %AC is the percentage of sequences with more than one positive annotation, %DAC is the percentage of sequences that have two or more annotations with different labels.

Dataset	#S	#T	#L	AAL	APT	ASL	AA	%AC	%DAC
CoNLL2000	8936	211727	11	1.6	20.6	24	12.0	98%	98%
CoNLL2000POS	8936	211727	35	1.0	20.8	24	20.8	100%	99%
CoNLL2003	17290	254979	4	1.4	2.5	15	2.2	45%	32%
NLPBA2004	18546	492551	5	2.5	5.9	27	3.1	72%	47%

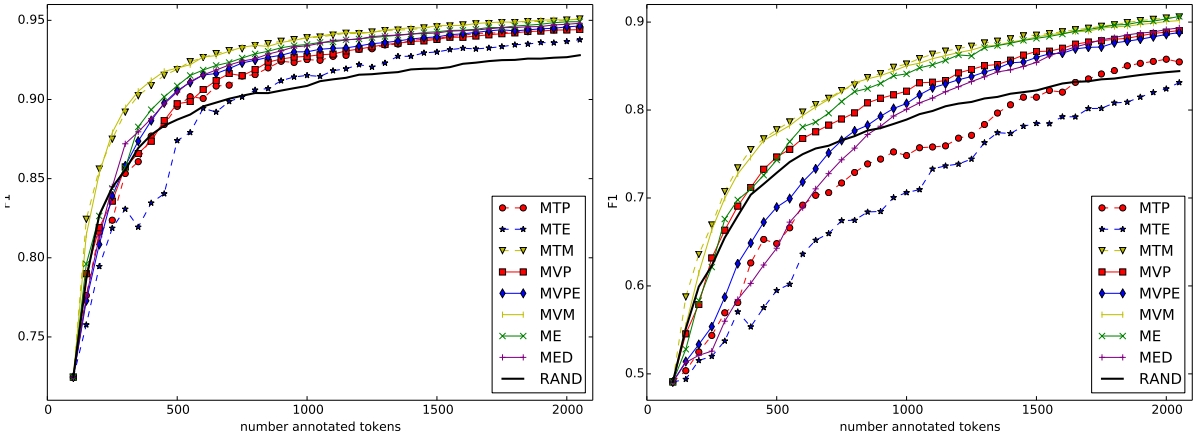

Figure 1: F_1 results on CoNLL2000 dataset (left) and CoNLL2000POS dataset (right). For both datasets the maximum number of annotated tokens used (2100) represents ∼1% of the entire training set.

datasets. The MVPE strategy performs particularly bad in the CoNLL2003 dataset but it performs better than the RAND strategy in the other datasets. The performance of the ME strategy is always above the average, in particular it is the best performing strategy in the NLPBA2004 dataset. However, in the CoNLL2003 dataset its performance is similar to the RAND's performance. Looking at the data, as expected, ME tends to choose tokens belonging to the longest sequences, regardless if the sequence is already partially annotated, that is, it tends to choose tokens from the same sequences. This behavior is not particularly relevant on the CoNLL2003 dataset given that the average number of positive tokens per sentence is not high (2.5, see Table 1). For the other datasets, the average number of positive tokens per sentence is high, and so the ME strategy is particularly effective. The MED strategy has the most heterogeneous behavior among the datasets. It shows average performances in the CoNLL2000 dataset and NLPBA2004 dataset, but is slower than the RAND strategy in the CoNLL2003 and CoNLL2000POS datasets.

In Figure 2 (left) we can notice that there are some strategies that are consistently worse than the RAND strategy. The difference between the strategies below the RAND strategy and the RAND strategy itself might be due to the fact that those strategies ask to label tokens that are "outliers" (if we imagine tokens as points of the features space) that rarely appear in the training and test set, and on which the classifier is very uncertain. Given that we are in a semi-supervised setting, with very few training examples, these "outliers" can introduce a lot of noise in the created models and so yielding poor results. This phenomenon does not happen in the RAND strategy given that it samples uniformly from the unlabeled set and given that the "outliers" (special cases) are not many, the probability of randomly selecting an "outlier" is low.

5.3.1 Performance Drop

The AL strategies applied on the CoNLL2003 dataset (Figure 2 (left)) suffer of some "random" drop of performance. We believe that the first reason that yield such a behavior is that named entities often appear once in a sentence, and have heterogeneous structures with respect to some homogenous structures as the chunk and POS. The second reason is that, it may happen that the strategies are not accurate enough to localize precisely the best token to label or that getting the label of an isolated token does not help the classifier much

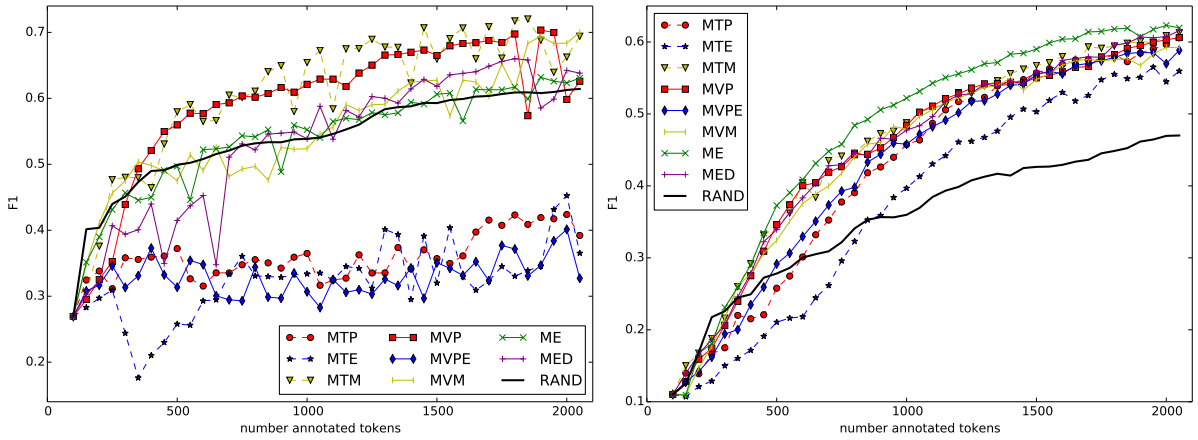

Figure 2: F_1 results on CoNLL2003 dataset (left) and NLPBA2004 dataset (right). 2100 annotated tokens represent the $\sim$0.8% and $\sim$0.4% respectively of the CoNLL2003 training set and the NLPBA2004 training set.

for the remaining of the (unlabeled) tokens in the sequence.

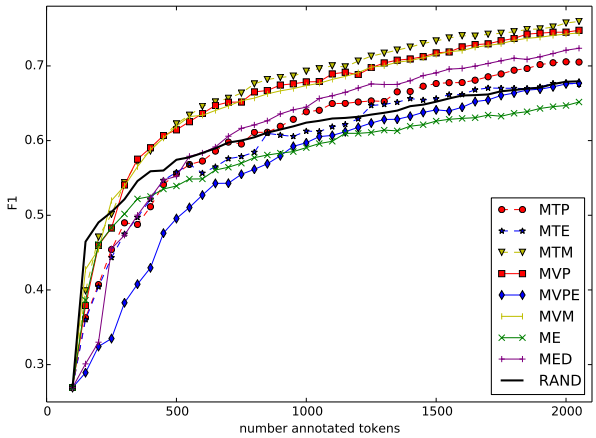

Figure 3: F_1 results on CoNLL2003 dataset, three tokens annotation. 6100 annotated tokens represent the $\sim$2.4% of the CoNLL2003 training set.

A similar phenomenon, called *missed class effect* (Tomanek et al., 2009), happens in AL when the strategies inspect regions of the example space around the decision boundaries, bringing to a slow AL process. In (Tomanek et al., 2009) the missed class effect problem is solved by helping the AL strategies to inspect regions far from the decision boundaries, that is, by choosing an entire sequence instead of a single token. This solution is not suitable in this context given that we will loose all the advantages we have in the partially-labeled scenario, thus, we decided to annotate for each chosen token the previous token and the next token. The learning curves of the AL strategies adopting this method (Figure 3) show a monotonically increasing performance in function of the number of annotated tokens.

By annotating three tokens at time, the tokens that were considered "outliers" in the scenario with a single token annotation are now supported by other tokens of the sequence. This fact helps to decrease the noise introduced in the semi-supervised model yielding better results.

5.3.2 Statistical Analysis

Figure 4 reports a few statistics that highlight the behavior of the methods on one of the datasets. One may see for instance that the MVM and ME strategies are very different from the other methods in that they select tokens that belong to significantly longer sentences on average. Also it may be seen that MVM in particular selects tokens that are far from already annotated tokens in the sentence. This strategy probably yields a particular behavior with respect to exploration and exploitation that seems to suit the two tasks well. The other strategies do exhibit different behaviors that intuitively should not work well. For instance the MED and the MVPE strategies select tokens from new fully unlabeled sentences (not shown statistics), preferably short, so that the distance from selected tokens to already labeled tokens in the sentence (when any) is low. These curves look like relevant indicators of the behavior of the methods, and it would probably be worth monitoring these all along the AL process to make sure the learning exhibit a suitable behavior. This will be a future study that is out of the scope of this work.

6 Conclusion

In this paper we have presented several AL strategies tailored for the PL-CRFs in a pool-based scenario. We have tested the proposed strategies on four different datasets for four different sequence labeling tasks. Differently from other similar work in the field of AL, in this study we have shown that margin-based strategies constantly achieve good performance on four tasks with very different data characteristics. Furthermore, we have found that on datasets with certain characteristics a particular phenomenon that makes the entire

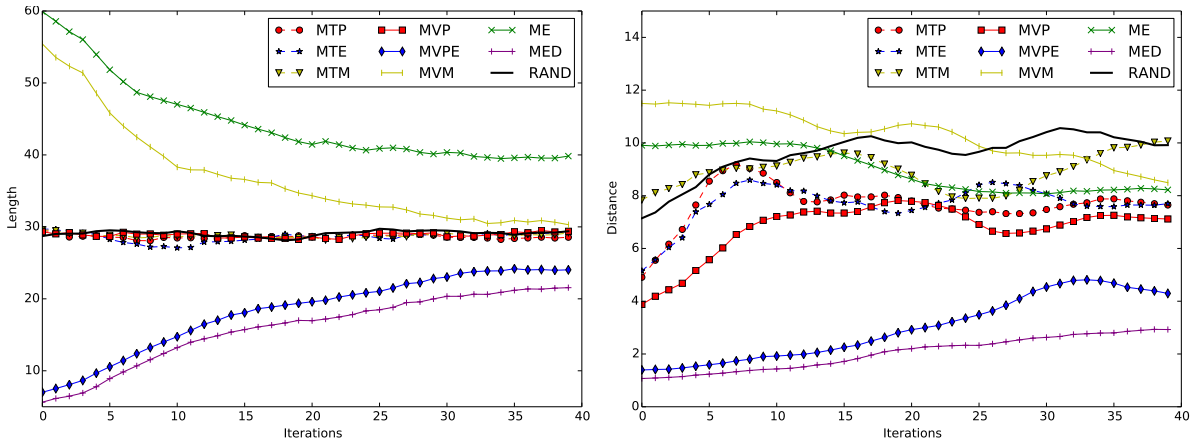

Figure 4: Behavior of the methods on CoNLL2000 dataset as a function of the number of the iterations (x-axis, from 1 to 40). Average length of the sentence the tokens that are selected by the AL strategy belong to (left) and average distance from a token that is selected to the closest already labeled token in the sentence, if any (right).

AL process highly unpredictable shows up. This phenomenon consists in random drops of accuracy of the classifiers learnt during the AL process. We have proposed a first solution for this problem that does not have a relevant impact on the human annotation effort.

Acknowledgments

We kindly thank Fabrizio Sebastiani and Andrea Esuli for their help and valuable comments, and Dittaya Wanvarie for providing us her implementation of partially-labeled CRFs.

References

Richard H. Byrd, Jorge Nocedal, and Robert B. Schnabel. 1994. Representations of quasi-Newton matrices and their use in limited memory methods. *Mathematical Programming*, 63:129–156.

Aron Culotta and Andrew McCallum. 2005. Reducing labeling effort for structured prediction tasks. In *Proceedings of the Twentieth National Conference on Artificial Intelligence and the Seventeenth Innovative Applications of Artificial Intelligence Conference, (AAAI 2005)*, pages 746–751, Pittsburgh, US.

Andrea Esuli and Fabrizio Sebastiani. 2010. Evaluating information extraction. In *Proceedings of the Conference on Multilingual and Multimodal Information Access Evaluation (CLEF 2010)*, pages 100–111, Padova, IT.

Andrea Esuli, Diego Marcheggiani, and Fabrizio Sebastiani. 2010. Sentence-based active learning strategies for information extraction. In *Proceedings of the First Italian Information Retrieval Workshop (IIR 2010)*, pages 41–45, Padua, Italy.

Jin-Dong Kim, Tomoko Ohta, Yoshimasa Tsuruoka, Yuka Tateisi, and Nigel Collier. 2004. Introduction to the bio-entity recognition task at JNLPBA. In *Proceedings of the International Joint Workshop on Natural Language Processing in Biomedicine and its Applications*, pages 70–75, Geneva, CH.

John Lafferty, Andrew McCallum, and Fernando Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the 18th International Conference on Machine Learning (ICML 2001)*, pages 282–289, Williamstown, US.

David D. Lewis and Jason Catlett. 1994. Heterogeneous uncertainty sampling for supervised learning. In *Proceedings of 11th International Conference on Machine Learning (ICML 1994)*, pages 148–156, New Brunswick, US.

Adwait Ratnaparkhi. 1996. A maximum entropy model for part-of-speech tagging. In *Proceedings of the conference on empirical methods in natural language processing*, volume 1, pages 133–142.

Sunita Sarawagi. 2008. Information extraction. *Foundations and Trends in Databases*, 1(3):261—377.

Tobias Scheffer, Christian Decomain, and Stefan Wrobel. 2001. Active hidden Markov models for information extraction. In *Proceedings of the 4th International Conference on Advances in Intelligent Data Analysis (IDA 2001)*, pages 309–318, Cascais, PT.

Burr Settles and Mark Craven. 2008. An analysis of active learning strategies for sequence labeling tasks. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP 2008)*, pages 1070–1079, Honolulu, US.

Burr Settles. 2012. *Active learning*. Synthesis Lectures on Artificial Intelligence and Machine Learning. Morgan & Claypool Publishers.

Dan Shen, Jie Zhang, Jian Su, Guodong Zhou, and Chew-Lim Tan. 2004. Multi-criteria-based active

learning for named entity recognition. In *Proceedings of the 42nd Meeting of the Association for Computational Linguistics (ACL 2004)*, pages 589–596, Barcelona, ES.

Erik F. Tjong Kim Sang and Sabine Buchholz. 2000. Introduction to the CoNLL-2000 shared task: Chunking. In *Proceedings of the 2nd Workshop on Learning Language in Logic and 4th Conference on Computational Natural Language Learning (LLL/CoNLL 2000)*, pages 127–132. Lisbon, PT.

Erik F. Tjong Kim Sang and Fien De Meulder. 2003. Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition. In *Proceedings of the 7th Conference on Natural Language Learning (CONLL 2003)*, pages 142–147, Edmonton, CA.

Katrin Tomanek and Udo Hahn. 2009. Semi-supervised active learning for sequence labeling. In *Proceedings of the 47th Annual Meeting of the Association for Computational Linguistics and the 4th International Joint Conference on Natural Language Processing of the AFNLP (ACL-IJCNLP 2009)*, pages 1039–1047, Singapore.

Katrin Tomanek, Florian Laws, Udo Hahn, and Hinrich Schütze. 2009. On proper unit selection in active learning: co-selection effects for named entity recognition. In *Proceedings of the NAACL HLT 2009 Workshop on Active Learning for Natural Language Processing*, pages 9–17, Boulder, US.

Yuta Tsuboi, Hisashi Kashima, Hiroki Oda, Shinsuke Mori, and Yuji Matsumoto. 2008. Training conditional random fields using incomplete annotations. In *Proceedings of the 22nd International Conference on Computational Linguistics (COLING 2008)*, pages 897–904, Manchester, UK.

Dittaya Wanvarie, Hiroya Takamura, and Manabu Okumura. 2011. Active learning with subsequence sampling strategy for sequence labeling tasks. *Information and Media Technologies*, 6(3):680–700.