Bidirectional Sequence Classification for Tagging Tasks with Guided Learning

Andrea Gesmundo
Université de Genève, route de Drize 7, 1227 Genève
andrea.gesmundo@unige.ch

Résumé. Dans cet article nous présentons une série d'adaptations de l'algorithme du "cadre d'apprenstissage guidé" pour résoudre différentes tâches d'étiquetage. La spécificité du système proposé réside dans sa capacité à apprendre l'ordre de l'inférence avec les paramètres du classifieur local au lieu de la forcer dans un ordre pré-défini (de gauche à droite). L'algorithme d'entraînement est basé sur l'algorithme du "perceptron". Nous appliquons le système à différents types de tâches d'étiquetage pour atteindre des résultats au niveau de l'état de l'art en un court temps d'exécution.

Abstract. In this paper we present a series of adaptations of the Guided Learning framework to solve different tagging tasks. The specificity of the proposed system lies in its ability to learn the order of inference together with the parameters of the local classifier instead of forcing it into a pre-defined order (left-to-right). The training algorithm is based on the Perceptron Algorithm. We apply the system to different kinds of tagging tasks reaching state of the art results with short execution time.

Mots-clés : Bidirectionnel, Classification de Séquence, Apprentissage Guidé.

Keywords: Bidirectional, Sequence Classification, Guided Learning.

1 Introduction

The system described in this paper carries out tagging tasks with semi-supervised training.We extend to the Guided Learning (GL) framework presented in (Shen *et al.*, 2007). This approach has been applied in the past to POS tagging task with excellent results. One of the aims of this paper is to show that GL can be adapted to solve a wide set of tagging and chunking tasks obtaining good performances with short execution time. This framework is more complex than supervised learning. The system can learn the parameters for the local classifier from gold standard labels, but has no indications on the order of inference. Basing the learning algorithm on the Perceptron scheme allows one to keep a low system complexity and moderate execution time, without sacrificing learning capability and quality of the results. Compared to other systems that use a Perceptron algorithm, such as (Collins, 2002), GL introduces a bidirectional search strategy. Instead of forcing the order of the tagging in a left-to-right fashion, any tagging order is allowed. GL follows an easiest-first approach and incorporates the learning of the order of inference in the training phase. In this way right-context and bidirectional-context features can be used at little extra cost. In a direct comparison with (Collins, 2002) we show that it is possible to achieve better accuracy with shorter execution time allowing the inference order to be predicted by the system instead of using an exhaustive search strategy.

We test the effectiveness of this approach applying it to different tagging tasks, taking part in shared tasks or experimenting on widely used corpora, this allows us to make a comparison between our system and the state of the art. The tasks we focus on are : Part of Speech Tagging, Noun Phrase Chunking, and Named Entity Recognition. NP chunking and NER are defined as chunking tasks, but following the general guidelines of (Ramshaw & Marcus, 1995) we can solve these problems as tagging tasks. For the chunking tasks, we apply a voting system between multiple data representations of text chunks (Shen & Sarkar, 2005).

2 Bidirectional Guided Classification

The input of the Inference Algorithm is a sequence of tokens $t_1 t_2 \cdots t_n$. For each token t_i, we have to assign a label $l_i \in L$, with L being the label set. A subsequence $t_i \cdots t_j$ is called a span, and is denoted $[i, j]$. To each span

s is associated a set of hypotheses H_s, and each hypothesis $h_s \in H_s$ is composed by a sequence of length $|s|$ over L.

Spans are started and grown by means of tagging actions. Three kinds of actions are available : it is possible to start a new span by labeling a token which doesn't have any context, or expand an existing span by labeling a token adjacent to the span, or merging two spans by labeling the token between them. In this last case, the two originating spans become subsequences of the resulting span, and the labeling action of the token between the spans use both right and left context information. In our system a trigram model is used. So if the span $[i, j]$ has already been tagged, we can use its hypothesized two left boundary labels (l_i, l_{i+1}) as right context when choosing the label for the token t_{i-1}.

For each hypothesis h associated with a span s, we maintain its most recent tagging action $a(h)$, and the hypotheses, if any, that have been used as left context $\hat{h}_l(h)$ and right context $\hat{h}_r(h)$. $\hat{h}_l(h)$ is the top hypothesis among the hypotheses H_l that are compatible with the left context used by the tagging action ; and similarly $\hat{h}_r(h)$ is the top hypothesis among the hypotheses H_r that are compatible with the right context used by the tagging action. The score function for hypotheses is computed in a recursive fashion, adding the score of the tagging action $U(a(h))$ to the scores of the left context $V(\hat{h}_l(h))$ and right context $V(\hat{h}_r(h))$ hypotheses :

$$V(h) = V(\hat{h}_l(h)) + V(\hat{h}_r(h)) + U(a(h)) \tag{1}$$

The score of the tagging action $U(a(h))$ is computed through a linear combination of the weight vector w and the feature vector of the action $f(a(h))$:

$$U(a(h)) = w \cdot f(a(h)) \tag{2}$$

We define the top hypothesis h^*_s for a span s to be the hypothesis in H_s with highest $V(h)$ score. So at each step of the algorithm we keep two kind of scores : $U(\cdot)$ the score of an action represents the confidence for the next labeling action, and $V(\cdot)$ the score of a hypothesis represents the overall quality of a partial result. The selection for the next tagging action directly depends on the score of the action. On the other hand, the score of the hypothesis is used to maintain the top partial results for each span. To reduce the search space explored during inference we apply a beam search strategy. For each span we consider only the B hypotheses with highest score $V(h)$. So in the worst case the computation of the top B hypotheses for a new span involves the scoring of every possible combination of the most recent action, left context, and right context, for a complexity of $O(B^2|L|)$.

Algorithm 1 Inference Algorithm
initialize the set of accepted spans S ;
initialize the queue of candidate spans Q ;
while $(Q \neq \emptyset)$ **do**
  span $s' \leftarrow \text{argmax}_{s \in Q} U(a(h^*_s))$;
  update S with s' ;
  update Q with s' and S ;

1	$[w_{-2}]$,$[w_{-1}]$,$[w_0]$,$[w_1]$,$[w_2]$,$[w_{-1},w_0]$,$[w_0,w_1]$
2	$[l_{-2}]$, $[l_{-1}]$, $[l_{-2},l_{-1}]$, $[l_{-2},w_0]$, $[l_{-1},w_0]$, $[l_{-2},l_{-1},w_0]$
3	$[l_1]$, $[l_2]$, $[l_1,l_2]$, $[l_1,w_0]$, $[l_2,w_0]$, $[l_1,l_2,w_0]$
4	$[l_{-1},l_1]$, $[l_{-1},l_1,w_0]$

TABLE 1: Context feature templates : 1) word features, 2) left context features, 3) right context features, 4) bidirectional features.

Algorithm 2 Guided Learning
for $(i \leftarrow 1; i \leq I; i{+}{+})$ **do**
  for $(r \leftarrow 1; r \leq R; r{+}{+})$ **do**
    initialize the set of accepted spans S ;
    initialize the queue of candidate spans Q ;
    while $(Q \neq \emptyset)$ **do**
      span $s' \leftarrow \text{argmax}_{s \in Q} U(a(h^*_s))$;
      if $h^*_{s'} = gold$ **then**
        update S with s' ;
        update Q with s' and S ;
      else
        $promote(w, f(gold))$;
        $demote(w, f(a(h)))$;

	TA	UWTA
(Dell'Orletta, 2009)	96.34%	91.07%
GL (Gesmundo, 2009b)	95.85%	91.41%

TABLE 2: Top two systems in Evalita 2009 POS task.

Algorithm 1 describes the Inference Algorithm. The token sequence $t_1 \cdots t_n$, the beam width B and the weight vector w are provided as input. The algorithm works using two groups of spans : S is the list of accepted spans, and Q is the a queue of candidate spans. At the beginning of the inference algorithm, S is initialized with the empty set, and Q is filled with candidate spans $[i, i]$ for each token t_i ; to these spans are associated the B best hypotheses consisting of a single tagging action, with no context, associating a label $l \in L$ to t_i. This provides the set of starting hypotheses. The loop of the algorithm repeatedly selects a candidate span s' from Q, so that its top hypothesis $h^*_{s'}$ has the highest tagging action score $U(a(\cdot))$. Thus we pick the span that results from the next

tagging action we are most confident about. Then we use s' to update S and Q. First we update S, adding s' and removing the spans included in s'. Then let S^- be the set of spans removed from S. We update Q replacing each span adjacent to s' or which takes as context one of the spans in S^- with a new candidate span taking s' as its new context. The algorithm terminates when S contains a single span covering the whole token sequence and Q becomes empty. The loop is guaranteed to terminate since at each iteration a span is expanded or added in S, and considering that S cannot have overlapping spans we can conclude that the number of iterations needed is linear with the size of the token sequence.

Algorithm 2 is the pseudocode for the Guided Learning Algorithm that learns the weight vector w with a Perceptron-like Approach. A set of input samples $\{(T_r, L_r)\}_{1 \leq r \leq m}$ is provided as input. To each token sequence $T_r = (t_1, t_2, \cdots t_n)$ is paired a gold standard label sequence of the same length $L_r = (l_1, l_2, \cdots, l_n)$. We provide also the beam width B and the number of iterations I. Before processing every input sample (T_r, L_r), we initialize S and Q as we do in the inference algorithm. Then we iterate selecting s' from Q, so that its top hypothesis $h^*_{s'}$ has the highest tagging action score $U(a(\cdot))$. If the top hypothesis of s' matches the gold standard, we update S and Q as in the inference algorithm. Otherwise, we update the weight vector w by promoting the features of the gold standard, and demoting the features of $a(h^*_{s'})$. Then we use the updated weight vector w to compute the new scores of the candidate spans in Q. Note that the update of the weight vector w is done in an aggressive fashion. S will not be updated and the weights are repeatedly modified until a correct tagging action is chosen from the queue of candidate spans Q. In our implementation we have used the Averaged Perceptron (Collins, 2002) and Perceptron with margin (Krauth & Mézard, 1987).

3 Experiments

In this section we describe the set of experiments conducted for the different tagging tasks and report and discuss the results. For all the tasks we set the beam width $B = 3$, as a trade-off between speed and accuracy. In Table 1 we report a basic set of context feature templates that we use to exploit the bidirectional context window over the labels and words. The basic set of lexical features contains functions to detect the presence of special characters as digits or hyphenation, prefixes and suffixes up to length of 9 characters, and capitalization pattern of the word in relation to the capitalization on context words. We use this basic set of contextual and lexical features as base to be adapted for the different tasks.

For the chunking tasks (NER and NP Chunking), we applied a voting system between multiple data representation of text chunks, following (Shen & Sarkar, 2005). We consider 5 different data representations for text chunks : IOB1 ; IOB2 ; IOE1 ; IOE2 ; O+C. We generate 5 versions of the corpus, one for each text chunk representation. Then we train one instance of system on each of the five versions of the corpus. As final step we generate 5 different predictions for the test set from each of the five representation specific systems, and merge the predictions with a majority vote. Differently from (Shen & Sarkar, 2005) we associate the votes to chunks instead of associating them to the single labels. This enforces more consistency in the output label sequence resulting from the voting system, and improves accuracy of results.

For some tasks we apply the semi-automatic technique to generate new feature templates described in (Gesmundo, 2007). This technique is based on iterations. During each iteration we aim to select the feature template that added to the current set of features will give higher performance improvements than any other candidate. The heuristic function is based on short experiments of 3 training rounds on a development set for each candidate feature. The value returned by the heuristic is based on a linear combination of parameters resulting from this short experiments, like F_1, or entropy of the distribution of the new features generated. The search space of candidate feature templates is restricted with hand written constraints on the size of the feature template or on the number of different types of labels taken in consideration.

3.1 POS Tagging on Wall Street Journal Corpus

In (Shen *et al.*, 2007) the GL framework has been tested on the POS Tagging task on the Wall Street Journal corpus, annotated with Penn Treebank tag-set. This is the standard data-set for POS on English corpus. For this experiment we use the basic feature set. On this corpus the system achieves an error rate of 2.67%, this result is recognized to be the state of the art result for POS tagging on English.

3.2 POS Tagging on Evalita 2009 Corpus

For the Evalita 2009 POS Tagging shared task, we used a corpus of 4013 sentences extracted from an Italian news paper, and tagged with the Tanl tag-set consisting of 328 labels. Each label consists of a combination of lexical and morphological features.

Considering the fine grained structure of the tag-set used, we introduced a new kind of context feature that considers just the prefix of the actual POS tag, excluding the morphological information encoded in the last part of the label, this led to a 3% relative error reduction. We also separated the treatment of the capitalization pattern of the first word of the sentence obtaining a relative error reduction of 0.9%.

In Table 2 we report the results for the top two systems in the final rank of the EVALITA 2009 POS shared task. GL is the only single system among the top performers, the other models are based on the combination of multiple systems. For example, the top ranking (Dell'Orletta, 2009) proposed a combination of 6 different single models. We can also notice that the Guided Learning approach obtained the best result on the Unknown Words Tagging Accuracy (UWTA). We consider this as a consequence of the freedom in the inference order : as explained earlier, the Guided Learning approach follows a tagging order based on an easiest-first heuristic, so difficult labeling decisions (as is the case for unknown words) are postponed, and a label is assigned when more context is available.

We recorded short execution times despite the large tag-set. The 20 rounds of training were completed in 12 hours, and the prediction of the test set was done in 2 minutes. During the training phase 1M features were generated.

3.3 Named Entity Recognition on CoNLL2003 Corpus

The experiments for NER on English are executed on the CoNLL 2003 data-set, this dataset is based on the Reuters Corpus, consisting on a total of 20717 sentences extracted from news articles. The corpus is annotated with 4 NE classes : Person ; Organization ; Location ; Miscellaneous. As additional input data were provided POS and Syntactic Chunk labels.

To exploit the Syntactic Chunk tags we added 3 feature templates manually selected to the basic features set, these features are reported in Table 3 line 6, adding these features led to a relative F_1 improvement of 3.21%. Then we preprocessed the Syntactic Chunk tags removing all the chunks different from the Noun Phrase Chunks. With the idea that all the NE chunks are contained in a Noun Phrase, removing the useless information should reduce noise in the decoding phase. Our intuition was confirmed by a F_1 relative improvement of 0.38%. At this point we tried to add manually selected context features that exploit the POS tags information, but any candidate added to the feature set resulted in a performance decrement. So we decided to resort to the semi-automatic feature selection technique to find new feature templates that exploit POS tags and NP chunks tags. These automatically selected feature templates are reported in Table 3 line 7 and 8. Adding these feature resulted in a relative F_1 improvement of 1.09%.

As external data for the final experiment we used gazetteers with 10k names of organizations, 42k names of locations and cities, 38k English proper names, and 7k miscellaneous named entities. Comparing our performances with the CoNLL rank we can see that our system surpasses the third best result. In Table 6 we report our best result , along with the two systems that perform better at the CoNLL 2003 task. (Chieu & Ng, 2003) uses global features, extracting information about words in same document. (Florian *et al.*, 2003) applies a combination of 4 different NER systems. Even if our system uses only local features, small gazetteers and no combination of different systems, it was able to reach a competitive result compared to the CoNLL rank.

Recently (Ratinov & Roth, 2009) recorded the best result on this corpus applying to a standard model a rich set of calibrated features like : wide scope global features, context aggregation, large gazetteers and word clusters generated from unlabeled text. As extension of this work, it would be interesting to apply these effective features to the GL approach.

The 13 rounds of training were completed in 2 hours, and the prediction of the test set was done in 1 minute. During the training phase 425k features were generated.

5	$[l_0, c_0]$, $[l_0, l_{-1}, c_0, c_{-1}]$, $[l_0, l_1, c_0, c_1]$
6	$[l_0, c_1, c_2]$, $[l_0, l_{-1}, l_1, l_2, c_1]$, $[l_0, l_{-1}, l_1, l_2, c_{-2}, c_{-1}, c_1, c_2]$,
7	$[l_0, l_1, p_{-1}, p_1]$, $[l_0, l_1, l_2, p_2]$, $[l_{-2}, l_0, p_{-1}]$, $[l_0, p_{-2}, p_{-1}, p_0, p_1, p_2]$

TABLE 3: Features added for CoNLL 2003 NER.

5	$[l_0, l_{-1}]$, $[n_{-2}, n_1, l_{-1}]$, $[n_{-2}, n_1, l_{-2}, l_{-1}, l_1]$

TABLE 4: Features added for EVALITA 2009 NER.

5	$[w_{-2}, w_{-1}]$, $[w_1, w_2]$
6	$[p_{-2}]$,$[p_{-1}]$,$[p_0]$,$[p_1]$,$[p_2]$
7	$[p_{-2}, p_{-1}]$,$[p_{-1}, p_0]$,$[p_0, p_1]$,$[p_1, p_2]$
8	$[p_{-2}, p_{-1}, p_0]$,$[p_{-1}, p_0, p_1]$,$[p_0, p_1, p_2]$

TABLE 5: Feature templates added for NP Chunking.

		F_1
1	(Florian *et al.*, 2003)	88.76
2	(Chieu & Ng, 2003)	88.31
3	GL with voting & gazetteers	87.53

TABLE 6: Results for the Conll2003 NER.

	F_1
(Zanoli *et al.*, 2009)	82.00
GL (Gesmundo, 2009a)	81.46
(Mehdad *et al.*, 2009)	81.09

TABLE 7: Top three systems at EVALITA 2009 NER.

	P. (%)	R. (%)	F_1
(Shen & Sarkar, 2005)	95.11	95.35	95.23
Guided Learning	94.78	94.34	94.56
(Sun *et al.*, 2008)	94.65	94.03	94.34

TABLE 8: Top three systems known for NP Chunking.

3.4 Named Entity Recognition on EVALITA 2009 I-CAB Corpus

The corpus for the Evalita 2009 NER shared task is composed of 11227 sentences extracted from an Italian newspaper, the corpus is labeled with 4 NE classes : Person ; Organization ; Geo-political entity ; Location. As input data were provided POS labels.

In Table 4 are reported the features obtained with the semi-automatic method for feature selection. Adding these features that use POS information led to a relative F_1 improvement of 1.1% on the development set. As external resources, we used gazetteers with 11k names of geographical locations, 49k Italian proper names and names, 14k organizations. The use of external resources led to a 6.89% relative F_1 improvement. After adding the voting system we recorded a relative F_1 improvement of 0.8% on the development set. In Table 7 we report the official score for the two best results for the EVALITA 2009 NER Shared Task. Also in this task GL reached the second position even if competing with more complex models that deploy greater amount of external resources.

The 8 rounds of training were completed in 1 hour and a half, and the tagging of the 4136 sentences of the test set took less then 2 minutes. During the training phase 500k features were generated.

3.5 Noun Phrase Chunking

The experiments for NP Chunking were executed on the Wall Street Journal Corpus, sections 15-18 (8936 sentences) for the training set and section 20 (2012 sentences) for testing. This dataset is the standard one for NP Chunking on English (Ramshaw & Marcus, 1995). For this task we used the same set of contextual feature used in (Collins, 2002), Collins also uses the same corpus to execute NP Chunking experiments. This allows a direct comparison between our GL and the HMM-Perceptron system proposed by Collins. To apply the same feature set we extend the base feature set with those features reported in Table 5. In his best system (Collins, 2002) records an F_1 of 93.53, with the same feature set GL obtains 94.44. Even if the Collin's perceptron deploys an exhaustive search strategy (Viterbi decoding) and our system applies a beam search approximate inference strategy, the latter is able to achieve better performance in shorter time. We believe this improvement is due to the ability of the GL approach to dynamically learn to predict the order of inference, instead of applying a monotonic order of inference (left-to-right) as in (Collins, 2002).

In Table 8 we report metrics for our best result obtained with voting scheme, and in the same table we report also the score of the only system that recorded a better performance in NP Chunking. We believe that the difference with the top system (Shen & Sarkar, 2005) is due to the use of a specialization technique, that consist in changing the input and output of the function being learned, enriching the tagset adding POS and lexical information to the NP labels as proposed in (Molina & Pla, 2002). As an extension of this work it would be interesting to adapt such a technique to the GL approach.

The 25 rounds of training needed for convergence of the Perceptron were completed in 1 hour and 10 minutes, and the tagging of the 2012 sentences of the test set took 35 seconds. During the training phase 400k features were generated.

4 Conclusion

In this paper we extended the work on the Guided Learning approach, adapting it to a set of tagging tasks, and applying new features. The evaluation results show that top three results are reached for all the tasks tested, proving the ability of the GL framework to adapt successfully to different tasks and corpora in different languages. We have shown that GL effectively integrate the order of inference and local classification in the learning phase. This results in a good generalization behavior during the decoding phase and allows it to reach good unknown words tagging accuracy. We have also shown how the ability to learn and predict the inference order allows GL to produce better results in shorter time when compared to the exhaustive-search left-to-right Perceptron-like approach of (Collins, 2002). The use of simple but effective training and inference algorithms results in moderate execution time. We have also confirmed the validity of a voting system for different data representations for text chunking, applying and improving with success the work of (Shen & Sarkar, 2005).

Acknowledgments

We thank Giorgio Satta, Libin Shen and James Henderson for help and advice.

Références

CHIEU H. L. & NG H. T. (2003). Named Entity Recognition with a Maximum Entropy approach. In *Proceedings of CoNLL-2003*.

COLLINS M. (2002). Discriminative training methods for Hidden Markov Models : Theory and experiments with Perceptron algorithms. In *Proceedings of EMNLP-2002*.

DELL'ORLETTA F. (2009). Ensemble system for part-of-speech tagging. In *Proceedings of EVALITA-2009*.

FLORIAN R., ITTYCHERIAH A., JING H. & ZHANG T. (2003). Named Entity Recognition through classifier combination. In *Proceedings of CoNLL-2003*.

GESMUNDO A. (2007). Elaborazione del linguaggio naturale basata su features bidirezionali. In *Master Thesis, Università di Padova*.

GESMUNDO A. (2009a). Bidirectional Sequence Classification for Named Entities Recognition. In *Proceedings of Evaluation of NLP and Speech Tools for Italian*.

GESMUNDO A. (2009b). Bidirectional Sequence Classification for Part of Speech Tagging. In *Proceedings of Evaluation of NLP and Speech Tools for Italian*.

KRAUTH W. & MÉZARD M. (1987). Learning algorithms with optimal stability in neural networks. In *Journal of Physics A, 20 :745-752*.

MEHDAD Y., SCURTU V. & STEPANOV E. (2009). Italian named entity recognizer participation in NER task @ Evalita 09. In *Proceedings of EVALITA-2009*.

MOLINA A. & PLA F. (2002). Shallow Parsing using specialized HMMs. In *Journal of Machine Learning Research, 2 :595-613, March 2002*.

RAMSHAW L. A. & MARCUS M. P. (1995). Text Chunking using Transformation-Based Learning. In *Proceedings of the Third ACL Workshop on Very Large Corpora*.

RATINOV L. & ROTH D. (2009). Design challenges and misconceptions in named entity recognition. In *Proceedings of CoNLL 2009*.

SHEN H. & SARKAR A. (2005). Voting between multiple data representations for Text Chunking. In *Proceedings of the Eighteenth Meeting of the Canadian Society for Computational Intelligence*.

SHEN L., SATTA G. & JOSHI A. K. (2007). Guided Learning for Bidirectional Sequence Classification. In *Proceedings of ACL-2007*.

SUN X., MORENCY L.-P., OKANOHARA D. & TSUJII J. (2008). Modeling Latent-Dynamic in Shallow Parsing : A latent conditional model with improved inference. In *Proceedings of COLING-2008*.

ZANOLI R., PIANTA E. & GIULIANO C. (2009). Named Entity Recognition through Redundancy Driven Classifiers. In *Proceedings of EVALITA-2009*.