Phrase Chunking using Entropy Guided Transformation Learning

Ruy L. Milidiú
Departamento de Informática
PUC-Rio
Rio de Janeiro, Brazil
milidiu@inf.puc-rio.br

Cícero Nogueira dos Santos
Departamento de Informática
PUC-Rio
nogueira@inf.puc-rio.br

Julio C. Duarte
Centro Tecnológico do Exército
Rio de Janeiro, Brazil
jduarte@ctex.eb.br

Abstract

Entropy Guided Transformation Learning (ETL) is a new machine learning strategy that combines the advantages of decision trees (DT) and Transformation Based Learning (TBL). In this work, we apply the ETL framework to four phrase chunking tasks: Portuguese noun phrase chunking, English base noun phrase chunking, English text chunking and Hindi text chunking. In all four tasks, ETL shows better results than Decision Trees and also than TBL with hand-crafted templates. ETL provides a new training strategy that accelerates transformation learning. For the English text chunking task this corresponds to a factor of five speedup. For Portuguese noun phrase chunking, ETL shows the best reported results for the task. For the other three linguistic tasks, ETL shows state-of-the-art competitive results and maintains the advantages of using a rule based system.

1 Introduction

Phrase Chunking is a Natural Language Processing (NLP) task that consists in dividing a text into syntactically correlated parts of words. Theses phrases are non-overlapping, i.e., a word can only be a member of one chunk (Sang and Buchholz, 2000). It provides a key feature that helps on more elaborated NLP tasks such as parsing and information extraction.

Since the last decade, many high-performance chunking systems were proposed, such as, SVM-based (Kudo and Matsumoto, 2001; Wu et al., 2006), Winnow (Zhang et al., 2002), voted-perceptrons (Carreras and Màrquez, 2003), Transformation-Based Learning (TBL) (Ramshaw and Marcus, 1999; Megyesi, 2002) and Hidden Markov Model (HMM) (Molina and Pla, 2002), Memory-based (Sang, 2002). State-of-the-art systems for English base noun phrase chunking and text chunking are based in statistical techniques (Kudo and Matsumoto, 2001; Wu et al., 2006; Zhang et al., 2002).

TBL is one of the most accurate rule-based techniques for phrase chunking tasks (Ramshaw and Marcus, 1999; Ngai and Florian, 2001; Megyesi, 2002). On the other hand, TBL rules must follow patterns, called templates, that are meant to capture the relevant feature combinations. The process of generating good templates is highly expensive. It strongly depends on the problem expert skills to build them. Even when a template set is available for a given task, it may not be effective when we change from a language to another (dos Santos and Oliveira, 2005).

In this work, we apply Entropy Guided Transformation Learning (ETL) for phrase chunking. ETL is a new machine learning strategy that combines the advantages of Decision Trees (DT) and TBL (dos Santos and Milidiú, 2007a). The ETL key idea is to use decision tree induction to obtain feature combinations (templates) and then use the TBL algorithm to generate transformation rules. ETL produces transformation rules that are more effective than decision trees and also eliminates the need of a problem domain expert to build TBL templates.

We evaluate the performance of ETL over four phrase chunking tasks: (1) English Base Noun Phrase (NP) chunking; (2) Portuguese NP chunking; (3) English Text Chunking; and (4) Hindi Text Chunking. Base NP chunking consists in recognizing non-overlapping text segments that contain NPs. Text chunking consists in dividing a text into syntactically correlated parts of words. For these four tasks, ETL shows state-of-the-art competitive results and maintains the advantages of using a rule based system.

The remainder of the paper is organized as follows. In section 2, the ETL strategy is described. In section 3, the experimental design and the corresponding results are reported. Finally, in section 4, we present our concluding remarks.

2 Entropy Guided Transformation Learning

Entropy Guided Transformation Learning (ETL) is a new machine learning strategy that combines the advantages of Decision Trees (DT) and Transformation-Based Learning (TBL) (dos Santos and Milidiú, 2007a). The key idea of ETL is to use decision tree induction to obtain templates. Next, the TBL strategy is used to generate transformation rules. The proposed method is illustrated in the Fig. 1.

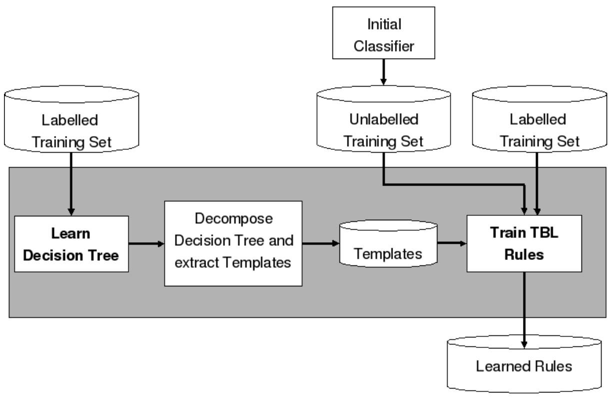

Figure 1: ETL - Entropy Guided Transformation Learning.

A combination of DT and TBL is presented in (Corston-Oliver and Gamon, 2003). The main difference between Corston-Oliver & Gamon work and the ETL strategy is that they extract candidate rules directly from the DT, and then use the TBL strategy to select the appropriate rules. Another difference is that they use a binary DT, whereas ETL uses a DT that is not necessarily binary.

An evolutionary approach based on Genetic Algorithms (GA) to automatically generate TBL templates is presented in (Milidiú et al., 2007). Using a simple genetic coding, the generated template sets have efficacy near to the handcrafted templates for the tasks: English Base Noun Phrase Identification, Text Chunking and Portuguese Named Entities Recognition. The main drawback of this strategy is that the GA step is computationally expensive. If we need to consider a large context window or a large number of features, it can be infeasible.

The remainder of this section is organized as follows. In section 2.1, we describe the DT learning algorithm. In section 2.2, the TBL algorithm is depicted. In section 2.3, we depict the process of obtaining templates from a decision tree decomposition. Finally, in section 2.4, we present a *template evolution scheme* that speeds up the TBL step.

2.1 Decision Trees

Decision tree learning is one of the most widely used machine learning algorithms. It performs a partitioning of the training set using principles of Information Theory. The learning algorithm executes a general to specific search of a feature space. The most informative feature is added to a tree structure at each step of the search. Information Gain Ratio, which is based on the data Entropy, is normally used as the informativeness measure. The objective is to construct a tree, using a minimal set of features, that efficiently partitions the training set into classes of observations. After the tree is grown, a pruning step is carried out in order to avoid overfitting.

One of the most used algorithms for induction of a DT is the C4.5 (Quinlan, 1993). We use Quinlan's C4.5 system throughout this work.

2.2 Transformation-Based Learning

Transformation Based error-driven Learning (TBL) is a successful machine learning algorithm introduced by Eric Brill (Brill, 1995). It has since been used for several Natural Language Processing tasks, such as part-of-speech (POS) tagging (Brill, 1995), English text chunking (Ramshaw and Marcus, 1999; dos Santos and Milidiú, 2007b), spelling correction (Mangu and Brill, 1997), Portuguese appositive extraction (Freitas et al., 2006), Portuguese named entity extraction (Milidiú et al., 2006) and Portuguese noun-phrase chunking (dos Santos and Oliveira, 2005), achieving state-of-the-art performance in many of them.

TBL uses an error correcting strategy. Its main scheme is to generate an ordered list of rules that correct classification mistakes in the training set, which have been produced by an initial classifier.

The requirements of the algorithm are:

- two instances of the training set, one that has been correctly labeled, and another that remains unlabeled;
- an initial classifier, the *baseline system*, which classifies the unlabeled training set by trying to apply the correct class for each sample. In general, the baseline system is based on simple statistics of the labeled training set; and
- a set of rule templates, which are meant to capture the relevant feature combinations that would determine the sample's classification. Concrete rules are acquired by instantiation of this predefined set of rule templates.
- a threshold value, that is used as a stopping criteria for the algorithm and is needed to avoid overfitting to the training data.

The learning method is a mistake-driven greedy procedure that iteratively acquires a set of transformation rules. The TBL algorithm can be depicted as follows:

1. Starts applying the baseline system, in order to guess an initial classification for the unlabeled version of the training set;
2. Compares the resulting classification with the correct one and, whenever a classification error is found, all the rules that can correct it are generated by instantiating the templates. This template instantiation is done by capturing some contextual data of the sample being corrected. Usually, a new rule will correct some errors, but will also generate some other errors by changing correctly classified samples;
3. Computes the rules' scores (errors repaired - errors created). If there is not a rule with a score above an arbitrary threshold, the learning process is stopped;
4. Selects the best scoring rule, stores it in the set of learned rules and applies it to the training set;
5. Returns to step 2.

When classifying a new sample item, the resulting sequence of rules is applied according to its generation order.

2.3 DT Template Extraction

There are many ways to extract feature combinations from decision trees. In an path from the root to the leaves, more informative features appear first . Since we want to generate the most promising templates only, we just combine the more informative ones.

The process we use to extract templates from a DT includes a depth-first traversal of the DT. For each visited node, we create a new template that combines its parent node template with the feature used to split the data at that node. This is a very simple decomposition scheme. Nevertheless, it results into extremely effective templates. We also use pruned trees in all experiments shown in section 3.

Fig. 2 shows an excerpt of a DT generated for the English text chunking task[1]. Using the described method to extract templates from the DT shown in Fig. 2, we obtain the template set listed in the left side of Table 1. In order to generate more feature combinations, without largely increasing the number of templates, we extend the template set by including templates that do not have the root node feature. The extended template set for the DT shown in Fig. 2 is listed in the right side of the Table 1.

We have also tried some other strategies that extract a larger number of templates from a DT. However, the efficacy of the learned rules is quite similar to the one generated by the first method. This reinforces the conjecture that a DT generates informative feature combinations.

[1]CK[0] = Chunk tag of the current word (initial classifier result); CK[–1] = previous word Chunk tag; CK[1] = next word Chunk tag; POS[0] = current word POS tag; WRD[0] = current word.

Table 1: Text chunking DT Template set example

Template set	Extended template set	
CK[0]	CK[0]	
CK[0] CK[1]	CK[0] CK[1]	CK[1]
CK[0] CK[1] WRD[0]	CK[0] CK[1] WRD[0]	CK[1] WRD[0]
CK[0] CK[1] WRD[0] CK[–1]	CK[0] CK[1] WRD[0] CK[–1]	CK[1] WRD[0] CK[–1]
CK[0] CK[1] POS[0]	CK[0] CK[1] POS[0]	CK[1] POS[0]
CK[0] CK[–1]	CK[0] CK[–1]	CK[–1]

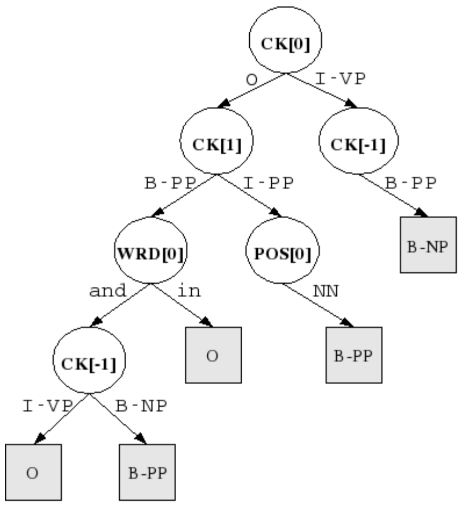

Figure 2: Text chunking decision tree excerpt.

2.4 Template Evolution Speedup

TBL training time is highly sensitive to the number and complexity of the applied templates. In (Curran and Wong, 2000), it is argued that we can better tune the *training time* vs. *templates complexity* trade-off by using an evolutionary template approach. The main idea is to apply only a small number of templates that evolve throughout the training. When training starts, templates are short, consisting of few feature combinations. As training proceeds, templates evolve to more complex ones that contain more feature combinations. In this way, only a few templates are considered at any point in time. Nevertheless, the descriptive power is not significantly reduced.

The template evolution approach can be easily implemented by using template sets extracted from a DT. We implement this idea by successively training TBL models. Each model uses only the templates that contain feature combinations up to a given tree level. For instance, using the tree shown in Fig. 2, we have the following template sets for the three first training rounds[2]:

1. CK[0] CK[1];
   CK[0] CK[–1]

2. CK[0] CK[1] WRD[0];
   CK[0] CK[1] POS[0]

3. CK[0] CK[1] WRD[0] CK[–1]

Using the template evolution strategy, the training time is decreased by a factor of five for the English text chunking task. This is a remarkable reduction, since we use an implementation of the *fastTBL* algorithm (Ngai and Florian, 2001) that is already a very fast TBL version. The efficacy of the rules generated by the sequential training is quite similar to the one obtained by training with all the templates at the same time.

3 Experiments

This section presents the experimental setup and results of the application of ETL to four phrase chunking tasks. ETL results are compared with the results of DT and TBL using hand-crafted templates.

In the TBL step, for each one of the four chunking tasks, the initial classifier assigns to each word the chunk tag that was most frequently associated with the part-of-speech of that word in the training set.

The DT learning works as a feature selector and is not affected by irrelevant features. We have tried several context window sizes when training the classifiers. Some of the tested window sizes would be very hard to be explored by a domain expert using

[2]We ignore templates composed of only one feature test.

TBL alone. The corresponding huge number of possible templates would be very difficult to be managed by a template designer.

For the four tasks, the following experimental setup provided us our best results.

ETL in the ETL learning, we use the features *word*, *POS* and *chunk*. In order to overcome the sparsity problem, we only use the 200 most frequent words to induce the DT. In the DT learning, the chunk tag of the word is the one applied by the initial classifier. On the other hand, the chunk tag of neighbor words are the true ones. We report results for ETL trained with all the templates at the same time as well as using template evolution.

TBL the results for the TBL approach refers to TBL trained with the set of templates proposed in (Ramshaw and Marcus, 1999).

DT the best result for the DT classifier is shown. The features *word*, *POS* and *chunk* are used to generate the DT classifier. The chunk tag of a word and its neighbors are the ones guessed by the initial classifier. Using only the 100 most frequent words gives our best results.

In all experiments, the term WS=X subscript means that a window of size X was used for the given model. For instance, $\text{ETL}_{WS=3}$ corresponds to ETL trained with window of size three, that is, the current token, the previous and the next one.

3.1 Portuguese noun phrase chunking

For this task, we use the SNR-CLIC corpus described in (Freitas et al., 2005). This corpus is tagged with both POS and NP tags. The NP tags are: I, for in NP; O, for out of NP; and B for the leftmost word of an NP beginning immediately after another NP. We divided the corpus into 3514-sentence (83346 tokens) training set and a 878-sentence (20798 tokens) test set.

In Table 2 we compare the results[3] of ETL with DT and TBL. We can see that ETL, even with a small window size, produces better results than DT and TBL. The $F_{\beta=1}$ of the $\text{ETL}_{WS=7}$ classifier is 1.8% higher than the one of TBL and 2.6% higher than the one of the DT classifier.

[3]#T = Number of templates.

Table 2: Portuguese noun phrase chunking.

	Acc. (%)	Prec. (%)	Rec. (%)	$F_{\beta=1}$ (%)	# T
BLS	96.57	62.69	74.45	68.06	–
$\text{DT}_{WS=13}$	97.35	83.96	87.27	85.58	–
TBL	97.45	85.48	87.32	86.39	100
$\text{ETL}_{WS=3}$	97.61	86.12	87.24	86.67	21
$\text{ETL}_{WS=5}$	97.68	86.85	87.49	87.17	35
$\text{ETL}_{WS=7}$	**97.82**	**88.15**	88.20	**88.18**	34
$\text{ETL}_{WS=9}$	**97.82**	88.02	**88.34**	**88.18**	40

Table 3 shows the results[4] of ETL using template evolution. As we can see, for the task of Portuguese noun phrase chunking, the template evolution strategy reduces the average training time in approximately 35%. On the other hand, there is a decrease of the classifier efficacy in some cases.

Table 3: Portuguese noun phrase chunking using ETL with template evolution.

	Acc. (%)	Prec. (%)	Rec. (%)	$F_{\beta=1}$ (%)	TTR (%)
$\text{ETL}_{WS=3}$	97.61	86.22	87.27	86.74	20.7
$\text{ETL}_{WS=5}$	97.56	86.39	87.10	86.74	38.2
$\text{ETL}_{WS=7}$	97.69	87.35	87.89	87.62	37.0
$\text{ETL}_{WS=9}$	**97.76**	**87.55**	**88.14**	**87.85**	41.9

In (dos Santos and Oliveira, 2005), a special set of six templates is shown. These templates are designed to reduce classification errors of preposition within the task of Portuguese noun phrase chunking. These templates use very specific domain knowledge and are difficult to DT and TBL to extract. Table 4 shows the results of an experiment where we include these six templates into the Ramshaw&Marcus template set and also into the template sets generated by ETL. Again, ETL produces better results than TBL.

Table 5 shows the results of using a committee composed by the three best ETL classifiers. The classification is done by selecting the most popular tag among all the three committee members. The achieved $F_{\beta=1}$, 89.14% is the best one ever reported for the SNR-CLIC corpus.

[4]TTR = Training time reduction.

Table 4: Portuguese noun phrase chunking using six additional hand-crafted templates.

	Acc. (%)	Prec. (%)	Rec. (%)	$F_{\beta=1}$ (%)	# T
BLS	96.57	62.69	74.45	68.06	–
TBL	97.60	86.79	88.12	87.45	106
$ETL_{WS=3}$	97.73	86.95	88.40	87.67	27
$ETL_{WS=5}$	97.87	88.35	89.02	88.68	41
$ETL_{WS=7}$	97.91	88.12	**89.22**	88.67	40
$ETL_{WS=9}$	**97.93**	**88.53**	89.11	**88.82**	46

Table 5: Committee with the classifiers $ETL_{WS=5}$, $ETL_{WS=7}$ and $ETL_{WS=9}$, shown in Table 4.

	Results (%)
Accuracy	97.97
Precision	88.62
Recall	89.67
$F_{\beta=1}$	89.14

3.2 English base noun phrase chunking

The data used in the base NP chunking experiments is the one by Ramshaw & Marcus (Ramshaw and Marcus, 1999). This corpus contains sections 15-18 and section 20 of the Penn Treebank, and is predivided into 8936-sentence (211727 tokens) training set and a 2012-sentence (47377 tokens) test. This corpus is tagged with both POS and chunk tags.

Table 6 compares the results of ETL with DT and TBL for the base NP chunking. We can see that ETL, even using a small window size, produces better results than DT and TBL. The $F_{\beta=1}$ of the $ETL_{WS=9}$ classifier is 0.87% higher than the one of TBL and 2.31% higher than the one of the DT classifier.

Table 7 shows the results of ETL using template evolution. The template evolution strategy reduces the average training time in approximately 62%. Differently from the Portuguese NP chunking, we observe an increase of the classifier efficacy in almost all the cases.

Table 8 shows the results of using a committee composed by the eight ETL classifiers reported in this section. Table 8 also shows the results for a committee of SVM models presented in (Kudo and Matsumoto, 2001). SVM's results are the state-of-the-art for the Base NP chunking task. On the other hand, using a committee of ETL classifiers, we produce very competitive results and maintain the advantages of using a rule based system.

Table 6: Base NP chunking.

	Acc. (%)	Prec. (%)	Rec. (%)	$F_{\beta=1}$ (%)	# T
BLS	94.48	78.20	81.87	79.99	–
$DT_{WS=11}$	97.03	89.92	91.16	90.53	–
TBL	97.42	91.68	92.26	91.97	100
$ETL_{WS=3}$	97.54	91.93	92.78	92.35	68
$ETL_{WS=5}$	97.55	92.43	92.77	92.60	85
$ETL_{WS=7}$	97.52	92.49	92.70	92.59	106
$ETL_{WS=9}$	**97.63**	**92.62**	**93.05**	**92.84**	122

Table 7: Base NP chunking using ETL with template evolution.

	Acc. (%)	Prec. (%)	Rec. (%)	$F_{\beta=1}$ (%)	TTR (%)
$ETL_{WS=3}$	97.58	92.07	92.74	92.41	53.9
$ETL_{WS=5}$	**97.63**	**92.66**	**93.16**	**92.91**	57.9
$ETL_{WS=7}$	97.61	92.56	93.04	92.80	65.1
$ETL_{WS=9}$	97.59	92.50	93.01	92.76	69.4

Table 8: Base NP chunking using a committee of eight ETL classifiers.

	Accuracy (%)	Precision (%)	Recall (%)	$F_{\beta=1}$ (%)
ETL	97.72	92.87	93.34	93.11
SVM	–	**94.15**	**94.29**	**94.22**

3.3 English text chunking

The data used in the English text chunking experiments is the CoNLL-2000 corpus, which is described in (Sang and Buchholz, 2000). It is composed by the same texts as the Ramshaw & Marcus (Ramshaw and Marcus, 1999) corpus.

Table 9 compares the results of ETL with DTs and TBL for English text chunking. ETL, even using a small window size, produces better results than DTs and TBL. The $F_{\beta=1}$ of the $ETL_{WS=3}$ classifier is 0.28% higher than the one of TBL and 2.17% higher than the one of the DT classifier. It is an interesting linguistic finding that the use of a window of size 3 (the current token, the previous token and the next token) provides the current best results for this task.

Table 9: English text Chunking.

	Acc. (%)	Prec. (%)	Rec. (%)	$F_{\beta=1}$ (%)	# T
BLS	77.29	72.58	82.14	77.07	–
$DT_{WS=9}$	94.29	89.55	91.00	90.27	–
TBL	95.12	92.05	92.28	92.16	100
$ETL_{WS=3}$	**95.24**	**92.32**	**92.56**	**92.44**	105
$ETL_{WS=5}$	95.12	92.19	92.27	92.23	167
$ETL_{WS=7}$	95.13	92.24	92.32	92.28	183
$ETL_{WS=9}$	95.07	92.10	92.27	92.19	205

Table 10 shows the results of ETL using template evolution. The template evolution strategy reduces the average training time by approximately 81%. On the other hand, there is a small decrease of the classifier efficacy in all cases.

Table 10: English text chunking using ETL with template evolution.

	Acc. (%)	Prec. (%)	Rec. (%)	$F_{\beta=1}$ (%)	TTR (%)
$ETL_{WS=3}$	**95.21**	**92.14**	**92.53**	**92.34**	77.2
$ETL_{WS=5}$	94.98	91.84	92.25	92.04	80.8
$ETL_{WS=7}$	95.03	91.89	92.28	92.09	83.0
$ETL_{WS=9}$	95.01	91.87	92.21	92.04	84.5

Table 11 shows the results of using a committee composed by the eight ETL classifiers reported in this section. Table 11 also shows the results for a SVM model presented in (Wu et al., 2006). SVM's results are the state-of-the-art for the Text chunking task. On the other hand, using a committee of ETL classifiers, we produce very competitive results and maintain the advantages of using a rule based system.

Table 11: English text Chunking using a committee of eight ETL classifiers.

	Accuracy (%)	Precision (%)	Recall (%)	$F_{\beta=1}$ (%)
ETL	95.50	92.63	92.96	92.79
SVM	–	**94.12**	**94.13**	**94.12**

Table 12 shows the results, broken down by chunk type, of using a committee composed by the eight ETL classifiers reported in this section.

Table 12: English text chunking results, broken down by chunk type, for the ETL committee.

	Precision (%)	Recall (%)	$F_{\beta=1}$ (%)
ADJP	75.59	72.83	74.19
ADVP	82.02	79.56	80.77
CONJP	35.71	55.56	43.48
INTJ	00.00	00.00	00.00
LST	00.00	00.00	00.00
NP	92.90	93.08	92.99
PP	96.53	97.63	97.08
PRT	66.93	80.19	72.96
SBAR	86.50	85.05	85.77
VP	92.84	93.58	93.21
Overall	92.63	92.96	92.79

3.4 Hindi text chunking

The data used in the Hindi text chunking experiments is the SPSAL-2007 corpus, which is described in (Bharati and Mannem, 2007). This corpus is pre-divided into a 20000-tokens training set, a 5000-tokens development set and a 5000-tokens test set. This corpus is tagged with both POS and chunk tags.

To fairly compare our approach with the ones presented in the SPSAL-2007, the POS tags of the test corpus were replaced by the ones predicted by an ETL-based Hindi POS Tagger. The description of our ETL pos tagger is beyond the scope of this work. Since the amount of training data is very small (20000 tokens), the accuracy of the ETL Hindi POS tagger is low, 77.50% for the test set.

The results are reported in terms of chunking accuracy, the same performance measure used in the SPSAL-2007. Table 13 compares the results of ETL with DT and TBL for Hindi text chunking. ETL produces better results than DT and achieves the same performance of TBL using 60% less templates. We believe that ETL performance is not as good as in the other tasks mainly because of the small amount of training data, which increases the sparsity problem.

We do not use template evolution for Hindi text chunking. Since the training corpus is very small, the training time reduction is not significant.

Table 13: Hindi text Chunking.

	Accuracy (%)	# Templates
BLS	70.05	–
$\text{DT}_{WS=5}$	78.20	–
TBL	**78.53**	100
$\text{ETL}_{WS=5}$	**78.53**	30

Table 14 compares the results of ETL with the two best Hindi text chunkers at SPSAL-2007 (Bharati and Mannem, 2007). The first one is a combination of Hidden Markov Models (HMM) and Conditional Random Fields (CRF) (PVS and Gali, 2007). The second is based in Maximum Entropy Models (MaxEnt) (Dandapat, 2007). ETL performs better than MaxEnt and worst than HMM+CRF. It is important to note that the accuracy of the POS tagger used by (PVS and Gali, 2007) (78.66%) is better than ours (77.50%). The POS tagging quality directly affects the chunking accuracy.

Table 14: Comparison with best systems of SPSAL-2007

	Accuracy (%)
HMM + CRF	**80.97**
$\text{ETL}_{WS=5}$	78.53
MaxEnt	74.92

4 Conclusions

In this paper, we approach the phrase chunking task using Entropy Guided Transformation Learning (ETL). We carry out experiments with four phrase chunking tasks: Portuguese noun phrase chunking, English base noun phrase chunking, English text chunking and Hindi text chunking. In all four tasks, ETL shows better results than Decision Trees and also than TBL with hand-crafted templates. ETL provides a new training strategy that accelerates transformation learning. For the English text chunking task this corresponds to a factor of five speedup. For Portuguese noun phrase chunking, ETL shows the best reported results for the task. For the other three linguistic tasks, ETL shows competitive results and maintains the advantages of using a rule based system.

References

Akshar Bharati and Prashanth R. Mannem. 2007. Introduction to shallow parsing contest on south asian languages. In *Proceedings of the IJCAI and the Workshop On Shallow Parsing for South Asian Languages (SPSAL)*, pages 1–8.

Eric Brill. 1995. Transformation-based error-driven learning and natural language processing: A case study in part-of-speech tagging. *Comput. Linguistics*, 21(4):543–565.

Xavier Carreras and Lluís Màrquez. 2003. Phrase recognition by filtering and ranking with perceptrons. In *Proceedings of RANLP-2003*, Borovets, Bulgaria.

Simon Corston-Oliver and Michael Gamon. 2003. Combining decision trees and transformation-based learning to correct transferred linguistic representations. In *Proceedings of the Ninth Machine Tranlsation Summit*, pages 55–62, New Orleans, USA. Association for Machine Translation in the Americas.

J. R. Curran and R. K. Wong. 2000. Formalisation of transformation-based learning. In *Proceedings of the Australian Computer Science Conference - ACSC*, pages 51–57, Canberra, Australia.

Sandipan Dandapat. 2007. Part of speech tagging and chunking with maximum entropy model. In *Proceedings of the IJCAI and the Workshop On Shallow Parsing for South Asian Languages (SPSAL)*, pages 29–32.

Cícero N. dos Santos and Ruy L. Milidiú. 2007a. Entropy guided transformation learning. Technical Report 29/07, Departamento de Informtica, PUC-Rio.

Cícero N. dos Santos and Ruy L. Milidiú. 2007b. Probabilistic classifications with tbl. In *Proceedings of Eighth International Conference on Intelligent Text Processing and Computational Linguistics – CICLing*, pages 196–207, Mexico City, Mexico, February.

Cícero N. dos Santos and Claudia Oliveira. 2005. Constrained atomic term: Widening the reach of rule templates in transformation based learning. In *EPIA*, pages 622–633.

M. C. Freitas, M. Garrao, C. Oliveira, C. N. dos Santos, and M. Silveira. 2005. A anotação de um corpus para o aprendizado supervisionado de um modelo de sn. In *Proceedings of the III TIL / XXV Congresso da SBC*, São Leopoldo - RS - Brasil.

M. C. Freitas, J. C. Duarte, C. N. dos Santos, R. L. Milidiú, R. P. Renteria, and V. Quental. 2006. A machine learning approach to the identification of appositives. In *Proceedings of Ibero-American AI Conference*, Ribeirão Preto, Brazil, October.

T. Kudo and Y. Matsumoto. 2001. Chunking with support vector machines. In *Proceedings of the NAACL-2001*.

Lidia Mangu and Eric Brill. 1997. Automatic rule acquisition for spelling correction. In *Proceedings of The Fourteenth International Conference on Machine Learning, ICML 97*. Morgan Kaufmann.

Beáta Megyesi. 2002. Shallow parsing with pos taggers and linguistic features. *Journal of Machine Learning Research*, 2:639–668.

Ruy L. Milidiú, Julio C. Duarte, and Roberto Cavalcante. 2006. Machine learning algorithms for portuguese named entity recognition. In *Proceedings of Fourth Workshop in Information and Human Language Technology (TIL'06)*, Ribeirão Preto, Brazil.

Ruy L. Milidiú, Julio C. Duarte, and Cícero N. dos Santos. 2007. Tbl template selection: An evolutionary approach. In *Proceedings of Conference of the Spanish Association for Artificial Intelligence - CAEPIA*, Salamanca, Spain.

Antonio Molina and Ferran Pla. 2002. Shallow parsing using specialized hmms. *J. Mach. Learn. Res.*, 2:595–613.

Grace Ngai and Radu Florian. 2001. Transformation-based learning in the fast lane. In *Proceedings of North Americal ACL*, pages 40–47, June.

Avinesh PVS and Karthik Gali. 2007. Part-of-speech tagging and chunking using conditional random fields and transformation based learning. In *Proceedings of the IJCAI and the Workshop On Shallow Parsing for South Asian Languages (SPSAL)*, pages 21–24.

J. Ross Quinlan. 1993. *C4.5: programs for machine learning*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.

Lance Ramshaw and Mitch Marcus. 1999. Text chunking using transformation-based learning. In S. Armstrong, K.W. Church, P. Isabelle, S. Manzi, E. Tzoukermann, and D. Yarowsky, editors, *Natural Language Processing Using Very Large Corpora*. Kluwer.

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 the 4th CONLL*, pages 127–132, Morristown, NJ, USA. Association for Computational Linguistics.

Erik F. Tjong Kim Sang. 2002. Memory-based shallow parsing. *J. Mach. Learn. Res.*, 2:559–594.

Yu-Chieh Wu, Chia-Hui Chang, and Yue-Shi Lee. 2006. A general and multi-lingual phrase chunking model based on masking method. In *Proceedings of 7th International Conference on Intelligent Text Processing and Computational Linguistics*, pages 144–155.

Tong Zhang, Fred Damerau, and David Johnson. 2002. Text chunking based on a generalization of winnow. *J. Mach. Learn. Res.*, 2:615–637.