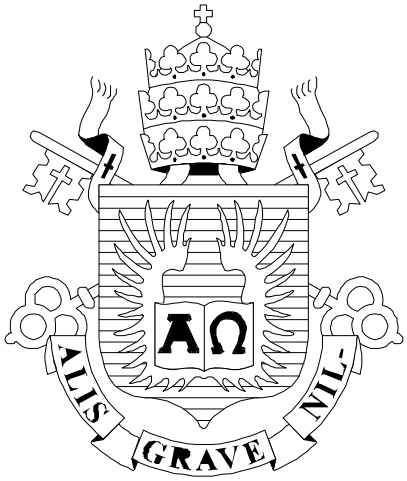

PUC

ISSN 0103-9741

Monografias em Ciência da Computação
nº 29/07

Entropy Guided Transformation Learning

Cícero Nogueira dos Santos
Ruy Luiz Milidiú

Departamento de Informática

PONTIFÍCIA UNIVERSIDADE CATÓLICA DO RIO DE JANEIRO
RUA MARQUÊS DE SÃO VICENTE, 225 - CEP 22451-900
RIO DE JANEIRO - BRASIL

Monografias em Ciência da Computação, No. 29/07 ISSN: 0103-9741
Editor: Prof. Carlos José Pereira de Lucena December, 2007

Entropy Guided Transformation Learning

Cícero Nogueira dos Santos, Ruy Luiz Milidiú

nogueira@inf.puc-rio.br, milidiu@inf.puc-rio.br

Abstract. This work presents a new machine learning strategy that combines the feature selection characteristics of Decision Trees (DT) and the robustness of Transformation Based Learning (TBL). The proposed method, Entropy Guided Transformation Learning (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 carry out experiments with three computational linguistic tasks: Portuguese noun phrase chunking, English base noun phrase chunking and text chunking. In all three tasks, ETL shows better results than Decision Trees and TBL with hand-crafted templates. ETL also provides a new training strategy that accelerates transformation learning by a factor of five. For Portuguese noun phrase chunking, ETL shows the best reported results for the task. For the other two linguistic tasks, ETL shows state-of-the-art competitive results and maintains the advantages of using a rule based system.

Keywords: Machine Learning, Decision Trees, Transformation Based Learning, Entropy Guided Transformation Learning.

Resumo. Este trabalho apresenta uma nova estratégia de aprendizado de máquina que combina as características de seleção de traços das Árvores de Decisão (DT) com a robustez do Aprendizado por Transformações (TBL). O método proposto, Aprendizado de Transformações Guiado pela Entropia (ETL), produz regras de transformação que são mais efetivas que árvores de decisão e elimina ainda a necessidade de um expecialista no domínio do problema para a construção dos gabaritos TBL. Desenvolvemos experimentos com três tarefas de linguística computacional: Extração de Sintagmas Nominais do português; Extração de Sintagmas Nominais Básicos do inglês; e análise sintática parcial do inglês. Nas três tarefas, o ETL apresenta melhores resultados do que as Árvores de Decisão e do que o TBL com gabaritos elaborados por humanos. O ETL também propicia uma nova estratégia que acelera por um fator de cinco o aprendizado de transformações. Para a Extração de Sintagmas Nominais do português, o ETL apresentou os melhores resultados reportados na literatura. Para as outras duas tarefas de linguística, o ETL apresentou resultados competitivos com o estado-da-arte, mantendo a vantagem de gerar um conjunto de regras.

Palavras-chave: Aprendizado de Máquina, Árvores de Decisão, Aprendizado Baseados em Transformações, Aprendizado de Transformações guiado por Entropia.

In charge of publications:

Rosane Teles Lins Castilho
Assessoria de Biblioteca, Documentação e Informação
PUC-Rio Departamento de Informática
Rua Marquês de São Vicente, 225 - Gávea
22451-900 Rio de Janeiro RJ Brasil
Tel. +55 21 3527-1516 Fax: +55 21 3527-1530
E-mail: bib-di@inf.puc-rio.br
Web site: http://bib-di.inf.puc-rio.br/techreports/

1 Introduction

Decision Trees (DT) and Transformation Based error-driven Learning (TBL) are widely used machine learning techniques. Both have the advantage that the outcome of their learning process is easily human interpretable. For Natural Language Processing (NLP) tasks, where the sparseness of the data is a constraint, TBL has proven to be more effective than DTs [Florian et al., 2000]. In [Brill, 1995], it is shown that TBL has some advantages over DTs, such as: (1) TBL rules are more readily interpretable; (2) TBL can be used as a postprocessor to any other classification system; and (3) TBL can access intermediate results of the classification process.

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 and strongly depends on the problem expert skills that builds them. When the number of features to be considered is large, the effort to manually create templates is extremely increased, becoming sometimes infeasible.

On the other hand, DT learning requires only a training set as input. In this sense, we can say that DT knowledge sources are cheap when compared to the expensive TBL templates. All information necessary for the decision trees induction is extracted from the training set. DT learning has the ability to automatically select good feature combinations.

A combination of DTs and TBL is presented in [Corston-Oliver and Gamon, 2003]. The main difference between Corston-Oliver & Gamon work and ours is that they extract candidate rules directly from the DT, and then use the TBL strategy to select the appropriate rules. Other difference is that they use binary DTs, whereas we use DTs that are 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.

In [Carberry et al., 2001], a randomized version of the TBL framework is shown. The idea is to use just a few templates, randomly chosen from the template set, when generating candidate rules for each error. This strategy speeds up the TBL training process, enabling the use of large template sets. On the other hand, in the experiments on Part-of-speech tagging, Carberry et al use handcrafted templates and variations of them, what implies that a template designer is still necessary.

Decision trees decomposition is used in [Hwang et al., 2003] to extract complex features for the Weighted Probabilistic Sum Model. This model is applied to English and Korean text chunking. Hwang et al work's is similar to ours in the sense that they use atomic feature combinations extracted from DTs as templates. On the other hand, their use of the extracted templates is totally different of ours.

In this work, we present a new machine learning strategy that combines the advantages of DTs and TBL. The key idea is to use decision tree induction to obtain feature combinations (templates) and then use the TBL strategy to generate transformation rules. The proposed method, Entropy Guided Transformation Learning (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.

The remainder of the paper is organized as follows. In section 2, the TBL algorithm is depicted. In section 3, we describe the DT learning algorithm. In section 4, we show the ETL strategy. In section 5, the experimental design and the corresponding results are reported. Finally, in section 6, we present our concluding remarks.

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, 2007], spelling correction [Mangu and Brill, 1997], portuguese appositive extraction [Freitas et al., 2006] and Portuguese noun-phrase chunking [dos Santos and Oliveira, 2005], achieving state-of-the-art performance in many of them.

In a classification problem setup, the application defines which feature is to be learnt. This feature is represented by a set of class labels Y. For instance, in the case of part-of-speech tagging, Y is the POS tagset.

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 guess.

The requirements of the algorithm are:

- two instances of the training set, one that has been correctly labeled with the Y's class labels, and another that remains unlabeled;
- an initial classifier, the baseline system, which classifies the unlabeled training set by trying to guess 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.

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 set, the resulting sequence of rules is applied according to its generation order.

3 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 in 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 DTs is the C4.5 [Quinlan, 1993]. We use Quinlan's C4.5 system throughout this work.

4 Entropy Guided Transformation Learning

Entropy Guided Transformation Learning (ETL) is a new machine learning strategy that combines the advantages of DTs and TBL. The key ETL idea 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.

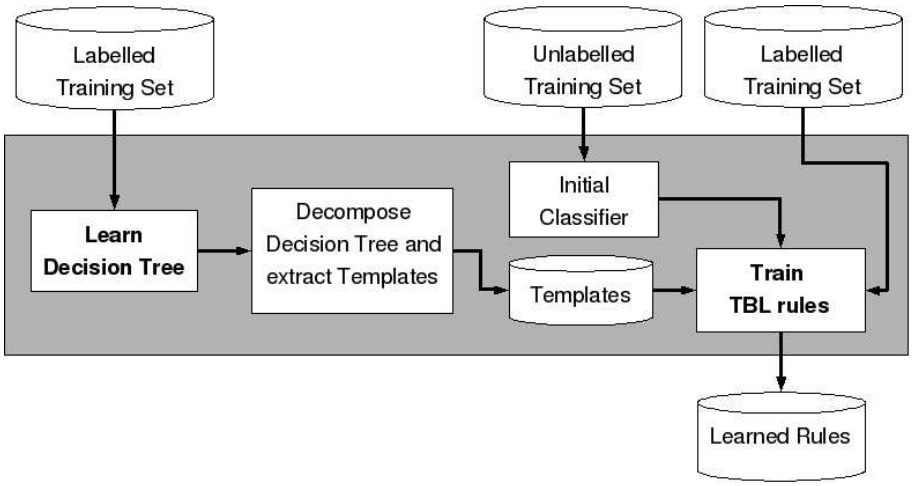

Figure 1: ETL - Entropy Guided Transformation Learning.

The remainder of this section is organized as follows. In section 4.1, we depict the process of obtaining templates from a decision tree decomposition. Finally, in section 4.2, we present a *template evolution scheme* that speeds up the TBL step.

4.1 DT Template Extraction

There are many ways to extract feature combinations from decision trees. More informative features appear first in an path from the root to the leaves. Since we want to generate

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

the most promising templates only, we just combine the more informative ones.

The process we use to extract templates from DTs 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 5.

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 DTs. However, the efficacy of the learned rules is quite similar to the one generated by the first method. This reinforces the conjecture that DTs generate informative feature combinations.

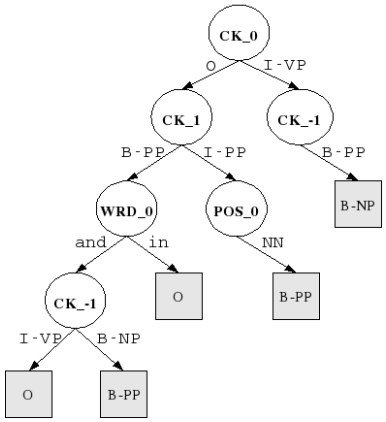

Figure 2: Text chunking decision tree excerpt.

[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.

4.2 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* tradeoff 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 DTs. 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 version of TBL. The efficacy of the rules generated by the sequential training is quite similar to the ones obtaining by training with all the templates at the same time.

5 Experiments

ETL effectiveness is illustrated through several experiments reported in this section. The experiments are performed on three tasks: Portuguese noun phrase chunking, English base noun phrase chunking and text chunking.

For each one of the three tasks, the base line system assigns to each word the chunk tag that was most frequently associated with the part-of-speech of the 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 using TBL alone, since the huge number of possible templates is very difficulty to be managed by a template designer.

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

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

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

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 neighbours 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 for the given model it was used a window of size X. (e.g. $\text{ETL}_{WS=3}$, ETL trained with window of size three, that is, the current token, the previous and the next one.)

5.1 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 of ETL with DTs and TBL. We can see that ETL, even with a small window size, produces better results than DTs and TBL. The $\text{F}_{\beta=1}$ 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.

Table 2: Portuguese noun phrase chunking.

	Accuracy (%)	Precision (%)	Recall (%)	$\text{F}_{\beta=1}$ (%)	# templates
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 of ETL using template evolution. As we can see, for the task of Portuguese noun phrase chunking, the template evolution strategy reduced the average training time in approximately 35%. On the other hand, there was a decrease of the classifier efficacy in some cases.

In [dos Santos and Oliveira, 2005], a special set of six templates is shown. These templates were designed to reduce classification errors of preposition within the task of Portuguese noun phrase chunking. These templates use very specific domain knowledge and are difficulty to DTs 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 $\text{F}_{\beta=1}$, 89.14%, is the best one ever reported for the SNR-CLIC corpus.

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

	Accuracy (%)	Precision (%)	Recall (%)	$F_{\beta=1}$ (%)	Training time reduction (%)
$ETL_{WS=3}$	97.61	86.22	87.27	86.74	20.65
$ETL_{WS=5}$	97.56	86.39	87.10	86.74	38.15
$ETL_{WS=7}$	97.69	87.35	87.89	87.62	37.02
$ETL_{WS=9}$	**97.76**	**87.55**	**88.14**	**87.85**	41.89

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

	Accuracy (%)	Precision (%)	Recall (%)	$F_{\beta=1}$ (%)	# templates
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

5.2 English base noun phrase chunking

Base noun phrase chunking consists in recognizing non-overlapping text segments that contain noun phrases (NPs). 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 pre-divided 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 DTs and TBL for the base NP chunking. We can see that ETL, even using a small window size, produces better results than DTs 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 reduced the average training time in approximately 61.6%. 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

Table 6: Base NP chunking.

	Accuracy (%)	Precision (%)	Recall (%)	$F_{\beta=1}$ (%)	# templates
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.

	Accuracy (%)	Precision (%)	Recall (%)	$F_{\beta=1}$ (%)	Training time reduction (%)
$ETL_{WS=3}$	97.58	92.07	92.74	92.41	53.85
$ETL_{WS=5}$	**97.63**	**92.66**	**93.16**	**92.91**	57.94
$ETL_{WS=7}$	97.61	92.56	93.04	92.80	65.13
$ETL_{WS=9}$	97.59	92.50	93.01	92.76	69.43

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 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**

5.3 English text chunking

Text chunking consists in dividing a text into syntactically correlated parts of words. The data used in the 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 best results for this task.

Table 10 shows the results of ETL using template evolution. The template evolution strategy reduces the average training time by approximately 81.39%. On the other hand,

Table 9: Text Chunking.

	Accuracy (%)	Precision (%)	Recall (%)	$F_{\beta=1}$ (%)	# templates
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

there is a small decrease of the classifier efficacy in all cases.

Table 10: Text Chunking using ETL with template evolution.

	Accuracy (%)	Precision (%)	Recall (%)	$F_{\beta=1}$ (%)	Training time reduction (%)
$ETL_{WS=3}$	**95.21**	**92.14**	**92.53**	**92.34**	77.23
$ETL_{WS=5}$	94.98	91.84	92.25	92.04	80.81
$ETL_{WS=7}$	95.03	91.89	92.28	92.09	83.04
$ETL_{WS=9}$	95.01	91.87	92.21	92.04	84.48

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: 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.

6 Conclusions

In this paper, we present Entropy Guided Transformation Learning (ETL), a learning method that 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 carry out experiments with three computational linguistic tasks: Portuguese noun phrase chunking, English base noun phrase chunking and text chunking. In all three tasks, ETL

Table 12: 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

shows better results than Decision Trees and TBL with hand-crafted templates. ETL also provides a new training strategy that accelerates transformation learning by a factor of five. For Portuguese noun phrase chunking, ETL shows the best reported results for the task. For the other two linguistic tasks, ETL shows competitive results and maintains the advantages of using a rule based system.

References

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

[Carberry et al., 2001] Carberry, S., Vijay-Shanker, K., Wilson, A., and Samuel, K. (2001). Randomized rule selection in transformation-based learning: a comparative study. *Natural Language Engineering*, 7(2):99–116.

[Corston-Oliver and Gamon, 2003] Corston-Oliver, S. and Gamon, M. (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.

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

[dos Santos and Milidiú, 2007] dos Santos, C. N. and Milidiú, R. L. (2007). Probabilistic classifications with tbl. In *Proceedings of Eighth International Conference on Intelligent Text Processing and Computational Linguistics – CICLing*, pages 196–207, Mexico City, Mexico.

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

[Florian et al., 2000] Florian, R., Henderson, J. C., and Ngai, G. (2000). Coaxing confidences from an old friend: Probabilistic classifications from transformation rule lists. In *Proceedings of Joint Sigdat Conference on Empirical Methods in NLP and Very Large Corpora*, Hong Kong University of Science and Technology.

[Freitas et al., 2005] Freitas, M. C., ao, M. G., Oliveira, C., dos Santos, C. N., and Silveira, M. (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.

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

[Hwang et al., 2003] Hwang, Y.-S., Chung, H.-J., and Rim, H.-C. (2003). Weighted probabilistic sum model based on decision tree decomposition for text chunking. *International Journal of Computer Processing of Oriental Languages*, (1):1–20.

[Kudo and Matsumoto, 2001] Kudo, T. and Matsumoto, Y. (2001). Chunking with support vector machines. In *Proceedings of the NAACL-2001*.

[Mangu and Brill, 1997] Mangu, L. and Brill, E. (1997). Automatic rule acquisition for spelling correction. In *Proceedings of The Fourteenth International Conference on Machine Learning, ICML 97*. Morgan Kaufmann.

[Milidiú et al., 2007] Milidiú, R. L., Duarte, J. C., and dos Santos, C. N. (2007). Tbl template selection: An evolutionary approach. In *Proceedings of Conference of the Spanish Association for Artificial Intelligence - CAEPIA*, Salamanca, Spain.

[Ngai and Florian, 2001] Ngai, G. and Florian, R. (2001). Transformation-based learning in the fast lane. In *Proceedings of North Americal ACL*, pages 40–47.

[Quinlan, 1993] Quinlan, J. R. (1993). *C4.5: programs for machine learning*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.

[Ramshaw and Marcus, 1999] Ramshaw, L. and Marcus, M. (1999). Text chunking using transformation-based learning. In Armstrong, S., Church, K., Isabelle, P., Manzi, S., Tzoukermann, E., and Yarowsky, D., editors, *Natural Language Processing Using Very Large Corpora*. Kluwer.

[Sang and Buchholz, 2000] Sang, E. F. T. K. and Buchholz, S. (2000). Introduction to the conll-2000 shared task: chunking. In *Proceedings of the 2nd workshop on Learning language in logic and the 4th conference on Computational Natural Language Learning*, pages 127–132, Morristown, NJ, USA. Association for Computational Linguistics.

[Wu et al., 2006] Wu, Y.-C., Chang, C.-H., and Lee, Y.-S. (2006). A general and multilingual phrase chunking model based on masking method. In *Proceedings of 7th International Conference on Intelligent Text Processing and Computational Linguistics*, pages 144–155.