Using Uneven Margins SVM and Perceptron for Information Extraction

Yaoyong Li, Kalina Bontcheva and Hamish Cunningham
Department of Computer Science, The University of Sheffield, Sheffield, S1 4DP, UK
{yaoyong,kalina,hamish}@dcs.shef.ac.uk

Abstract

The classification problem derived from information extraction (IE) has an imbalanced training set. This is particularly true when learning from smaller datasets which often have a few positive training examples and many negative ones. This paper takes two popular IE algorithms – SVM and Perceptron – and demonstrates how the introduction of an uneven margins parameter can improve the results on imbalanced training data in IE. Our experiments demonstrate that the uneven margin was indeed helpful, especially when learning from few examples. Essentially, the smaller the training set is, the more beneficial the uneven margin can be. We also compare our systems to other state-of-the-art algorithms on several benchmarking corpora for IE.

1 Introduction

Information Extraction (IE) is the process of automatic extraction of information about pre-specified types of events, entities or relations from text such as newswire articles or Web pages. IE is useful in many applications, such as information gathering in a variety of domains, automatic annotations of web pages for Semantic Web, and knowledge management.

A wide range of machine learning techniques have been used for IE and achieved state-of-the-art results, comparable to manually engineered IE systems. A learning algorithm usually learns a model from a set of documents which have been manually annotated by the user. Then the model can be used to extract information from new documents. Manual annotation is a time-consuming process. Hence, in many cases learning from small data sets is highly desirable. Therefore in this paper we also evaluate the performance of our algorithms on small amounts of training data and show their learning curve.

The learning algorithms for IE can be classified broadly into two main categories: rule learning and statistical learning. The former induces a set of rules from training examples. There are many rule based learning systems, e.g. SRV (Freitag, 1998), RAPIER (Califf, 1998), WHISK (Soderland, 1999), BWI (Freitag and Kushmerick, 2000), and $(LP)^2$ (Ciravegna, 2001). Statistical systems learn a statistical model or classifiers, such as HMMs (Freigtag and McCallum, 1999), Maximal Entropy (Chieu and Ng., 2002), the SVM (Isozaki and Kazawa, 2002; Mayfield et al., 2003), and Perceptron (Carreras et al., 2003). IE systems also differ from each other in the NLP features that they use. These include simple features such as token form and capitalisation information, linguistic features such as part-of-speech, semantic information from gazetteer lists, and genre-specific information such as document structure. In general, the more features the system uses, the better performance it can achieve.

This paper concentrates on classifier-based learning for IE, which typically converts the recognition of each information entity into a set of classification problems. In the framework discussed here, two binary classifiers are trained for each type of information entity. One classifier is used for recognising the entity's start token and the other – the entity's end token.

The classification problem derived from IE usually has imbalanced training data, in which positive training examples are vastly outnumbered by negative ones. This is particularly true for smaller data sets where often there are hundreds of negative training examples and only few positive ones. Two approaches have been studied so far to deal with imbalanced data in IE. One approach is to under-sample majority class or over-sample minority class in order to obtain a relatively balanced training data (Zhang and Mani, 2003). However, under-sampling can potentially remove certain important examples, and over-sampling can lead to over-fitting and a larger training set. Another approach is to divide the problem into several sub-problems in two layers, each of which has less imbalanced training set than the original one (Carreras et al., 2003; Sitter and Daelemans, 2003). The output of the classifier in the first layer is used as the input to the classifiers in the second layer. As a result, this approach needs more classifiers than the original problem. Moreover, the classification errors in the first layer will affect the performance of the second one.

In this paper we explore another approach to handle the imbalanced data in IE, namely, adapting the learning algorithms for balanced classification to imbalanced data. We particularly study two popular classification algorithms in IE, Support Vector Machines (SVM) and Perceptron.

SVM is a general supervised machine learning algorithm, that has achieved state of the art performance on many classification tasks, including NE recognition. Isozaki and Kazawa (2002) compared three commonly used methods for named entity recognition – the SVM with quadratic kernel, maximal entropy method, and a rule based learning system, and showed that the SVM-based system performed better than the other two. Mayfield et al. (2003) used a lattice-based approach to named entity recognition and employed the SVM with cubic kernel to compute transition probabilities in a lattice. Their results on CoNLL2003 shared task were comparable to other systems but were not the best ones.

Previous research on using SVMs for IE adopts the standard form of the SVM, which treats positive and negative examples equally. As a result, they did not consider the difference between the balanced classification problems, where the SVM performs quite well, and the imbalanced ones. Li and Shawe-Taylor (2003) proposes an uneven margins version of the SVM and shows that the SVM with uneven margins performs significantly better than the standard SVM on document classification problems with imbalanced training data. Since the classification problem for IE is also imbalanced, this paper investigates the SVM with uneven margins for IE tasks and demonstrates empirically that the uneven margins SVM does have better performance than the standard SVM.

Perceptron is a simple, fast and effective learning algorithm, which has successfully been applied to named entity recognition (Carreras et al., 2003). The system uses a two-layer structure of classifiers to handle the imbalanced data. The first layer classifies each word as entity or non-entity. The second layer classifies the named entities identified by the first layer in the respective entity classes. Li et al. (2002) proposed another variant of Perceptron, the Perceptron algorithm with uneven margins (PAUM), designed especially for imbalanced data. In this paper we explore the application of PAUM to IE.

The rest of the paper is structured as follows. Section 2 describes the uneven margins SVM and Perceptron algorithms. Sections 3.1 and 3.2 discuss the classifier-based framework for IE and the experimental datasets we used, respectively. We compare our systems to other state-of-the-art systems on three benchmark datasets in Section 3.3. Section 3.4 discusses the effects of the uneven margins parameter on the SVM and Perceptron's performances. Finally, Section 4 provides some conclusions.

2 Uneven Margins SVM and Perceptron

Li and Shawe-Taylor (2003) introduced an uneven margins parameter into the SVM to deal with imbalanced classification problems. They showed that the SVM with uneven margins outperformed the standard SVM on document classification problem with imbalanced training data. Formally, given a training set $\mathbf{Z} = ((\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_m, y_m))$,where $\mathbf{x}_i$ is the n-dimensional input vector and y_i ($= +1$ or -1) its label, the SVM with uneven margins is obtained by solving the quadratic optimisation problem:

$$\min_{\mathbf{w},\, b,\, \xi} \langle \mathbf{w}, \mathbf{w} \rangle + C \sum_{i=1}^{m} \xi_i$$

$$\text{s.t. } \langle \mathbf{w}, \mathbf{x}_i \rangle + \xi_i + b \geq 1 \quad \text{if } y_i = +1$$

$$\langle \mathbf{w}, \mathbf{x}_i \rangle - \xi_i + b \leq -\tau \quad \text{if } y_i = -1$$

$$\xi_i \geq 0 \qquad \text{for } i = 1, ..., m$$

We can see that the uneven margins parameter τ was added to the constraints of the optimisation problem. τ is the ratio of negative margin to the positive margin of the classifier and is equal to 1 in the standard SVM. For an imbalanced dataset with a few positive examples and many negative ones, it would be beneficial to use larger margin for positive examples than for the negative ones. Li and Shawe-Taylor (2003) also showed that the solution of the above problem could be obtained by solving a related standard SVM problem by, for example, using a publicly available SVM package[1].

Perceptron is an on-line learning algorithm for linear classification. It checks the training examples one by one by predicting their labels. If the prediction is correct, the example is passed; otherwise, the example is used to correct the model. The algorithm stops when the model classifies all training examples correctly. The margin Perceptron not only classifies every training example correctly but also outputs for every training example a value (before thresholding) larger than a predefined parameter (margin). The margin Perceptron has better generalisation capability than the standard Perceptron. Li et al. (2002) proposed the Perceptron algorithm with uneven margins (PAUM) by introducing two margin parameters τ_+ and τ_- into the updating rules for the positive and negative examples, respectively. Similar to the uneven margins parameter in SVM, two margin parameters allow the PAUM to handle imbalanced datasets better than both the standard Perceptron and the margin Perceptron. Additionally, it is known that the Perceptron learning will stop after limited loops only on a linearly separable training set. Hence, a regularisation parameter λ is used in PAUM to guarantee that the algorithm would stop for any training dataset after some updates. PAUM is simple and fast and performed very well on document classification, in particularly on imbalanced training data.

[1]The SVM^{light} package version 3.5, available from http://svmlight.joachims.org/, was used to learn the SVM classifiers in our experiments.

3 Experiments

3.1 Classifier-Based Framework for IE

In the experiments we adopted a classifier-based framework for applying the SVM and PAUM algorithms to IE. The framework consists of three stages: pre-processing of the documents to obtain feature vectors, learning classifiers or applying classifiers to test documents, and finally post-processing the results to tag the documents.

The aim of the preprocessing is to form input vectors from documents. Each document is first processed using the open-source ANNIE system, which is part of GATE[2] (Cunningham et al., 2002). This produces a number of linguistic (NLP) features, including token form, capitalisation information, token kind, lemma, part-of-speech (POS) tag, semantic classes from gazetteers, and named entity types according to ANNIE's rule-based recogniser.

Based on the linguistic information, an input vector is constructed for each token, as we iterate through the tokens in each document (including word, number, punctuation and other symbols) to see if the current token belongs to an information entity or not. Since in IE the context of the token is usually as important as the token itself, the features in the input vector come not only from the current token, but also from preceding and following ones. As the input vector incorporates information from the context surrounding the current token, features from different tokens can be weighted differently, based on their position in the context. The weighting scheme we use is the *reciprocal scheme*, which weights the surrounding tokens reciprocally to the distance to the token in the centre of the context window. This reflects the intuition that the nearer a neighbouring token is, the more important it is for classifying the given token. Our experiments showed that such a weighting scheme obtained better results than the commonly used equal weighting of features (Li et al., 2005).

The key part of the framework is to convert the recognition of information entities into binary classification tasks – one to decide whether a token is the start of an entity and another one for the end token.

After classification, the start and end tags of the

[2]Available from http://www.gate.ac.uk/

entities are obtained and need to be combined into one entity tag. Therefore some post-processing is needed to guarantee tag consistency and to try to improve the results by exploring other information. The currently implemented procedure has three stages. First, in order to guarantee the consistency of the recognition results, the document is scanned from left to right to remove start tags without matching end tags and end tags without preceding start tags. The second stage filters out candidate entities from the output of the first stage, based on their length. Namely, a candidate entity tag is removed if the entity's length (i.e., the number of tokens) is not equal to the length of any entity of the same type in the training set. The third stage puts together all possible tags for a sequence of tokens and chooses the best one according to the probability which was computed from the output of the classifiers (before thresholding) via a Sigmoid function.

3.2 The Experimental Datasets

The paper reports evaluation results on three corpora covering different IE tasks – named entity recognition (CoNLL-2003) and template filling or scenario templates in different domains (Jobs and CFP). The CoNLL-2003[3] provides the most recent evaluation results of many learning algorithms on named entity recognition. The Jobs corpus[4] has also been used recently by several learning systems. The CFP corpus was created as part of the recent Pascal Challenge for evaluation of machine learning methods for IE[5].

In detail, we used the English part of the CoNLL-2003 shared task dataset, which consists of 946 documents for training, 216 document for development (e.g., tuning the parameters in learning algorithm), and 231 documents for evaluation (i.e., testing), all of which are news articles taken from the Reuters English corpus (RCV1). The corpus contains four types of named entities — person, location, organisation and miscellaneous names. In the other two corpora domain-specific information was extracted into a number of slots. The Job corpus includes 300 computer related job advertisements and 17 slots encoding job details, such as title, salary, recruiter, computer language, application, and platform. The CFP corpus consists of 1100 conference or workshop call for papers (CFP), of which 600 were annotated. The corpus includes 11 slots such as workshop and conference names and acronyms, workshop date, location and homepage.

3.3 Comparison to Other Systems

Named Entity Recognition The algorithms are evaluated on the CoNLL-2003 dataset. Since this set comes with development data for tuning the learning algorithm, different settings were tried in order to obtain the best performance on the development set. Different SVM kernel types, window sizes (namely the number of tokens in left or right side of the token at the centre of window), and the uneven margins parameter τ were tested. We found that quadratic kernel, window size 4 and $\tau = 0.5$ produced best results on the development set. These settings were used in all experiments on the CoNLL-2003 dataset in this paper, unless otherwise stated. The parameter settings for PAUM described in Li et al. (2002), e.g. $\tau_+ = 50, \tau_- = 1$, were adopted in all experiments with PAUM, unless otherwise stated.

Table 1 presents the results of our system using three learning algorithms, the uneven margins SVM, the standard SVM and the PAUM on the CONLL-2003 test set, together with the results of three participating systems in the CoNLL-2003 shared task: the best system (Florian et al., 2003), the SVM-based system (Mayfield et al., 2003) and the Perceptron-based system (Carreras et al., 2003).

Firstly, our uneven margins SVM system performed significantly better than the other SVM-based system. As the two systems are different from each other in not only the SVM models used but also other aspects such as the NLP features and the framework, in order to make a fair comparison between the uneven margins SVM and the standard SVM, we also present the results of the two learning algorithms implemented in our framework. We can see from Table 1 that, under the same experimental settings, the uneven margins SVM again performed better than the standard SVM.

Secondly, our PAUM-based system performed slightly better than the system based on voted Perceptron, but there is no significant difference between them. Note that they adopted different mechanisms to deal with the imbalanced data in IE (refer

[3]See http://cnts.uia.ac.be/conll2003/ner/

[4]See http://www.isi.edu/info-agents/RISE/repository.html.

[5]See http://nlp.shef.ac.uk/pascal/.

Table 1: Comparison to other systems on CoNLL-2003 corpus: F-measure(%) on each entity type and the overall micro-averaged F-measure. The 90% confidence intervals for results of other three systems are also presented. The best performance figures for each entity type and overall appear in bold.

	System	LOC	MISC	ORG	PER	Overall
Our Systems	SVM with uneven margins	89.25	77.79	82.29	90.92	86.30
	Standard SVM	88.86	77.32	80.16	88.93	85.05
	PAUM	88.18	76.64	78.26	89.73	84.36
Participating Systems	Best one	**91.15**	**80.44**	**84.67**	**93.85**	**88.76($\pm$0.7)**
	Another SVM	88.77	74.19	79.00	90.67	84.67($\pm$1.0)
	Voted Perceptron	87.88	77.97	80.09	87.31	84.30($\pm$0.9)

to Section 1). The structure of PAUM system is simpler than that of the voted Perceptron system.

Finally, the PAUM system performed worse than the SVM system. On the other hand, training time of PAUM is only 1% of that for the SVM and the PAUM implementation is much simpler than that of SVM. Therefore, when simplicity and speed are required, PAUM presents a good alternative.

Template Filling On *Jobs corpus* our systems are compared to several state-of-the-art learning systems, which include the rule based systems Rapier (Califf, 1998), $(LP)^2$ (Ciravegna, 2001) and BWI (Freitag and Kushmerick, 2000), the statistical system HMM (Freitag and Kushmerick, 2000), and the double classification system (Sitter and Daelemans, 2003). In order to make the comparison as informative as possible, the same settings are adopted in our experiments as those used by $(LP)^2$, which previously reported the highest results on this dataset. In particular, the results are obtained by averaging the performance in ten runs, using a random half of the corpus for training and the rest for testing. Only basic NLP features are used: token form, capitalisation information, token types, and lemmas.

Preliminary experiments established that the SVM with linear kernel obtained better results than SVM with quadratic kernel on the Jobs corpus (Li et al., 2005). Hence we used the SVM with linear kernel in the experiments on the Jobs data. Note that PAUM always uses linear kernel in our experiments.

Table 2 presents the results of our systems as well as the other six systems which have been evaluated on the Jobs corpus. Note that the results for all the 17 slots are available for only three systems, Rapier, $(LP)^2$ and double classification, while the results for some slots were available for the other three systems. We computed the macro-averaged F_1 (the mean of the F_1 of all slots) for our systems as well as for the three fully evaluated systems in order to make a comparison of the overall performance.

Firstly, the overall performance of our two systems is significantly better than the other three fully evaluated systems. The PAUM system achieves the best performance on 5 out of the 17 slots. The SVM system performs best on the other 3 slots. Secondly, the double classification system had much worse overall performance than our systems and other two fully evaluated systems. HMM was evaluated only on two slots. It achieved best result on one slot but was much worse on the other slot than our two systems and some of the others. Finally, somewhat surprisingly, our PAUM system achieves better performance than the SVM system on this dataset. Moreover, the computation time of PAUM is about 1/3 of that of the SVM. Hence, the PAUM system performs quite satisfactory on the Jobs corpus.

Our systems were also evaluated by participating in a Pascal challenge – Evaluating Machine Learning for Information Extraction. The evaluation provided not only the *CFP corpus* but also the linguistic features for all tokens by pre-processing the documents. The main purpose of the challenge was to evaluate machine learning algorithms based on the same linguistic features. The only compulsory task is task1, which used 400 annotated documents for training and other 200 annotated documents for testing. See Ireson and Ciravegna (2005) for a short overview of the challenge. The learning methods explored by the participating systems included LP^2, HMM, CRF, SVM, and a variety of combinations

Table 2: Comparison to other systems on the jobs corpus: F_1 (%) on each entity type and overall performance as macro-averaged F_1. Standard deviations for the MA F_1 of our systems are presented in parenthesis. The highest score on each slot and overall performance appears in bold.

Slot	SVM	PAUM	$(LP)^2$	Rapier	DCs	BWI	HMM	semi-CRF
Id	97.7	97.4	**100**	97.5	97	100	–	–
Title	49.6	53.1	43.9	40.5	35	50.1	**57.7**	40.2
Company	77.2	**78.4**	71.9	70.0	38	78.2	50.4	60.9
Salary	**86.5**	86.4	62.8	67.4	67	–	–	–
Recruiter	78.4	**81.4**	80.6	68.4	55	–	–	–
State	92.8	93.6	84.7	90.2	**94**	–	–	–
City	**95.5**	95.2	93.0	90.4	91	–	–	–
Country	96.2	**96.5**	81.0	93.2	92	–	–	–
Language	86.9	87.3	**91.0**	81.8	33	–	–	–
Platform	80.1	78.4	**80.5**	72.5	36	–	–	–
Application	70.2	69.7	**78.4**	69.3	30	–	–	–
Area	46.8	**54.0**	53.7	42.4	17	–	–	–
Req-years-e	**80.8**	80.0	68.8	67.2	76	–	–	–
Des-years-e	81.9	85.6	60.4	**87.5**	47	–	–	–
Req-degree	87.5	**87.9**	84.7	81.5	45	–	–	–
Des-degree	59.2	62.9	65.1	**72.2**	33	–	–	–
Post date	99.2	99.4	**99.5**	**99.5**	98	–	–	–
MA F_1	80.8($\pm$1.0)	**81.6($\pm$1.1)**	77.2	76.0	57.9	–	–	–

of different learning algorithms. Firstly, the system of the challenge organisers, which is based on LP^2 obtained the best result for Task1, followed by one of our participating systems which combined the uneven margins SVM and PAUM (see Ireson and Ciravegna (2005)). Our SVM and PAUM systems on their own were respectively in the fourth and fifth position among the 20 participating systems. Secondly, at least six other participating system were also based on SVM but used different IE framework and possibly different SVM models from our SVM system. Our SVM system achieved better results than all those SVM-based systems, showing that the SVM models and the IE framework of our system were quite suitable to IE task. Thirdly, our PAUM based system was not as good as our SVM system but was still better than the other SVM based systems. The computation time of the PAUM system was about 1/5 of that of our SVM system.

Table 3 presents the per slot results and overall performance of our SVM and PAUM systems as well as the system with the best overall result. Compared to the best system, our SVM system performed better on two slots and had similar results on many of other slots. The best system had extremely good results on the two slots, C-acronym and C-homepage. Actually, the F_1 values of the best system on the two slots were more than double of those of every other participating system.

3.4 Effects of Uneven Margins Parameter

A number of experiments were conducted to investigate the influence of the uneven margins parameter on the SVM and Perceptron's performances. Table 4 show the results with several different values of uneven margins parameter respectively for the SVM and the Perceptron on two datasets – CoNLL-2003 and Jobs. The SVM with uneven margins ($\tau < 1.0$) had better results than the standard SVM ($\tau = 1$). We can also see that the results were similar for the τ between 0.6 and 0.4, showing that the results are not particularly sensitive to the value of the uneven margins parameter. The uneven margins parameter has similar effect on Perceptron as on the SVM. Table 4 shows that the PAUM had better results than both the standard Perceptron and the margin Perceptron

Table 3: Results of our SVM and PAUM systems on CFP corpus: F-measures(%) on individual entity type and the overall figures, together with the system with the highest overall score. The highest score on each slot appears in bold.

SLOT	PAUM	SVM	Best one
W-name	51.9	**54.2**	35.2
W-acronym	50.4	60.0	**86.5**
W-date	67.0	69.0	**69.4**
W-homepage	69.6	70.5	**72.1**
W-location	60.0	**66.0**	48.8
W-submission	70.2	69.6	**86.4**
W-notification	76.1	85.6	**88.9**
W-camera-ready	71.5	74.7	**87.0**
C-name	43.2	47.7	**55.1**
C-acronym	38.8	38.7	**90.5**
C-homepage	7.1	11.6	**39.3**
Micro-average	61.1	64.3	**73.4**

Our conjecture was that the uneven margins parameter was more helpful on small training sets, because the smaller a training set is, the more imbalanced it could be. Therefore we carried out experiments on a small numbers of training documents. Table 5 shows the results of the SVM and the uneven margins SVM on different numbers of training documents from CoNLL-2003 and Jobs datasets. The performance of both the standard SVM and the uneven margins SVM improves consistently as more training documents are used. Moreover, compared to the results one large training sets shown in Table 4, the uneven margins SVM obtains more improvements on small training sets than the standard SVM model. We can see that the smaller the training set is, the better the results of the uneven margins SVM are in comparison to the standard SVM.

4 Conclusions

This paper studied the uneven margins versions of two learning algorithms – SVM and Perceptron – to deal with the imbalanced training data in IE. Our experiments showed that the uneven margin is helpful, in particular on small training sets. The smaller the training set is, the more beneficial the uneven margin could be. We also showed that the systems based on the uneven margins SVM and Perceptron were comparable to other state-of-the-art systems.

Table 4: The effects of uneven margins parameter of the SVM and Perceptron, respectively: macro averaged $F_1(\%)$ on the two datasets CoNLL-2003 (development set) and Jobs. The standard deviations for the Jobs dataset show the statistical significances of the results. In bold are the best performance figures for each dataset and each system.

τ	1.0	0.8	0.6	0.4	0.2
Conll	89.0	89.6	**89.7**	89.2	85.3
Jobs	79.0	79.9	**81.0**	80.8	79.0
	±1.4	±1.2	±**0.9**	±1.0	±1.3
(τ_+, τ_-)	(0,0)	(1,1)	(50,1)		
Conll	83.5	83.9	**84.4**		
Jobs	74.1	78.8	**81.6**		
	±1.5	±1.0	±**1.1**		

Our SVM system obtained better results than other SVM-based systems on the CoNLL-2003 corpus and CFP corpus respectively, while being simpler than most of them. This demonstrates that our SVM system is both effective and efficient.

We also explored PAUM, a simple and fast learning algorithm for IE. The results of PAUM were somehow worse (about 0.02 overall F-measure lower) than those of the SVM on two out of three datasets. On the other hand, PAUM is much faster to train and easier to implement than SVM. It is also worth noting that PAUM outperformed some other learning algorithms. Therefore, even PAUM on its own would be a good learning algorithm for IE. Moreover, PAUM could be used in combination with other classifiers or in the more complicated framework such as the one in Carreras et al. (2003).

Since many other tasks in Natural Language Processing, like IE, often lead to imbalanced classification problems and the SVM has been used widely in Natural Language Learning (NLL), we can expect that the uneven margins SVM and PAUM are likely to obtain good results on other NLL problems as well.

Acknowledgements

This work is supported by the EU-funded SEKT project (http://www.sekt-project.org).

Table 5: The performances of the SVM system with small training sets: macro-averaged $F_1(\%)$ on the two datasets CoNLL-2003 (development set) and Jobs. The uneven margins SVM ($\tau = 0.4$) is compared to the standard SVM model with even margins ($\tau = 1$). The standard deviations are presented for results on the Jobs dataset.

size	10	20	30	40	50
	$\tau = 0.4$				
Conll	60.6	66.4	70.4	72.2	72.8
Jobs	51.6 ±2.7	60.9 ±2.5	65.7 ±2.1	68.6 ±1.9	71.1 ±2.5
	$\tau = 1$				
Conll	46.2	58.6	65.2	68.3	68.6
Jobs	47.1 ±3.4	56.5 ±3.1	61.4 ±2.7	65.4 ±1.9	68.1 ±2.1

References

M. E. Califf. 1998. *Relational Learning Techniques for Natural Language Information Extraction*. Ph.D. thesis, University of Texas at Austin.

X. Carreras, L. Màrquez, and L. Padró. 2003. Learning a perceptron-based named entity chunker via online recognition feedback. In *Proceedings of CoNLL-2003*, pages 156–159. Edmonton, Canada.

H. L. Chieu and H. T. Ng. 2002. A Maximum Entropy Approach to Information Extraction from Semi-Structured and Free Text. In *Proceedings of the Eighteenth National Conference on Artificial Intelligence*, pages 786–791.

F. Ciravegna. 2001. $(LP)^2$, an Adaptive Algorithm for Information Extraction from Web-related Texts. In *Proceedings of the IJCAI-2001 Workshop on Adaptive Text Extraction and Mining*, Seattle.

H. Cunningham, D. Maynard, K. Bontcheva, and V. Tablan. 2002. GATE: A Framework and Graphical Development Environment for Robust NLP Tools and Applications. In *Proceedings of the 40th Anniversary Meeting of the Association for Computational Linguistics (ACL'02)*.

R. Florian, A. Ittycheriah, H. Jing, and T. Zhang. 2003. Named Entity Recognition through Classifier Combination. In *Proceedings of CoNLL-2003*, pages 168–171. Edmonton, Canada.

D. Freigtag and A. K. McCallum. 1999. Information Extraction with HMMs and Shrinkage. In *Proceesings of Workshop on Machine Learnig for Information Extraction*, pages 31–36.

D. Freitag and N. Kushmerick. 2000. Boosted Wrapper Induction. In *Proceedings of AAAI 2000*.

D. Freitag. 1998. *Machine Learning for Information Extraction in Informal Domains*. Ph.D. thesis, Carnegie Mellon University.

N. Ireson and F. Ciravegna. 2005. Pascal Challenge The Evaluation of Machine Learning for Information Extraction. In *Proceedings of Dagstuhl Seminar Machine Learning for the Semantic Web (http://www.smi.ucd.ie/Dagstuhl-MLSW/proceedings/)*.

H. Isozaki and H. Kazawa. 2002. Efficient Support Vector Classifiers for Named Entity Recognition. In *Proceedings of the 19th International Conference on Computational Linguistics (COLING'02)*, pages 390–396, Taipei, Taiwan.

Y. Li and J. Shawe-Taylor. 2003. The SVM with Uneven Margins and Chinese Document Categorization. In *Proceedings of The 17th Pacific Asia Conference on Language, Information and Computation (PACLIC17)*, Singapore, Oct.

Y. Li, H. Zaragoza, R. Herbrich, J. Shawe-Taylor, and J. Kandola. 2002. The Perceptron Algorithm with Uneven Margins. In *Proceedings of the 9th International Conference on Machine Learning (ICML-2002)*, pages 379–386.

Y. Li, K. Bontcheva, and H. Cunningham. 2005. SVM Based Learning System For Information Extraction. In *Proceedings of Sheffield Machine Learning Workshop*, Lecture Notes in Computer Science. Springer Verlag.

J. Mayfield, P. McNamee, and C. Piatko. 2003. Named Entity Recognition Using Hundreds of Thousands of Features. In *Proceedings of CoNLL-2003*, pages 184–187. Edmonton, Canada.

A. De Sitter and W. Daelemans. 2003. Information extraction via double classification. In *Proceedings of ECML/PRDD 2003 Workshop on Adaptive Text Extraction and Mining (ATEM 2003)*, Cavtat-Dubrovnik, Croatia.

S. Soderland. 1999. Learning information extraction rules for semi-structured and free text. *Machine Learning*, 34(1):233–272.

J. Zhang and I. Mani. 2003. KNN Approach to Unbalanced Data Distributions: A Case Study Involving Information Extraction. In *Proceedings of the ICML'2003 Workshop on Learning from Imbalanced Datasets*.