Use of Support Vector Machines in *Extended* Named Entity Recognition

Koichi Takeuchi and **Nigel Collier**
National Institute of Informatics
2-1-2 Hitotsubashi, Chiyoda-ku, Tokyo, 101-8430 Japan
E-mail:{koichi,collier}@nii.ac.jp

Abstract

This paper explores the use of Support Vector Machines (SVMs) for an extended named entity task. We investigate the identification and classification of technical terms in the molecular biology domain and contrast this to results obtained for traditional NE recognition on the MUC-6 data set. Furthermore we compare the performance of the SVM model to a standard HMM bigram model. Results show that the SVM utilizing a rich feature set of a ± 3 context window and POS features (MUC-6 only) had a significant performance advantage on both the MUC-6 and molecular biology data sets.

1 Introduction

Named entity (NE) extraction is now firmly established as a core technology for understanding low level semantics of texts. NE was formalized in the DARPA-sponsored Message Understanding Conference (MUC)-6 (MUC, 1995) and since then several methodologies have been widely explored:

- heuristics-based, using rules written by human experts after inspecting examples (Fukuda et al., 1998);
- supervised such as (Bikel et al., 1997) using labelled training examples;
- non-supervised methods such as (Collins and Singer, 1999).

NE's main role has been to identify expressions such as the names of people, places and organizations as well as date and time expressions. Such expressions are hard to analyze using traditional natural language processing (NLP) because they belong to the open class of expressions, i.e. there is an infinite variety and new expressions are constantly being invented.

The application of NE to non-news domains requires us to consider extending NE so that it can capture types, i.e. instances of conceptual classes as well as individuals. To distinguish between traditional NE and *extended* NE we refer to the later as *NE+*. There are several issues that may mean that NE+ is more challenging that NE. The most important is the number of variants of NE+ expressions due to graphical, morphological, shallow syntactic and discourse variations. For example the use of head sharing combined with embedded abbreviations in *unliganded (apo)- and liganded (holo)-LBD*. Such expressions will require syntactic analysis beyond simple noun phrase chunking if they are to be successfully captured. NE+ expressions may also require richer contextual evidence than is needed for regular NEs - for example knowledge of the head noun or the predicate. At the ontology level there are complex issues related to granularity when deciding on which class a possible NE+ expression should be assigned to.

NE+ expressions will typically belong to a much richer taxonomy than NE, opening up the possibility of combining information extraction (IE) with deep knowledge representations such as ontologies. This is an area we are currently exploring (Collier et al., 2002). Examples of

NE+ classes include, a person's name, a protein name, a chemical formula or a computer product code. All of these may be valid candidates for tagging depending on whether they are contained in the ontology.

NE+ can be viewed as a type of multiple classification task and there are several effective and well studied learning algorithms available for this such as Hidden Markov Models (HMMs) (Rabiner and Juang, 1986) and transformation-based error-driven learning (TBL) (Brill, 1995). Recently a new learning paradigm called support vector machines (SVMs) (Vapnik, 1995) has been the focus of intensive research in machine learning due to its capacity to learn effectively from large feature sets. SVMs have been applied very successfully in the past to several traditional classification tasks such as text classification. Promising results have been reported for NLP tasks such as part of speech tagging and chunking, e.g. (Kudoh and Matsumoto, 2000).

We have implemented and compared two learning methods (SVM, HMM) and tested them on two data sets. The comparison between these models is informative because of the different nature of the two learning methods. In the case of the HMM the learning approach is *generative*, i.e. it makes use of positive examples to build a model of NE classes and then evaluates each unseen sentence to see how well each of the words 'fits' the model. The SVM on the other hand is a *discriminative* approach and makes use of both positive and negative examples to learn the distinction between the two classes. Another major difference is that the SVM outputs a measure of distance from the classification function whereas the HMM uses the Viterbi algorithm (Viterbi, 1967) to decode using maximum likelihood probabilities. Basically we expect the models to have quite different strengths and weaknesses and hopefully these can be complementary, allowing us eventually to combine the approaches to achieve a composite model. The two models are described further below along with their performance.

2 Method

2.1 SVM

We developed our method using the Tiny SVM package from NAIST [1] which is an implementation of Vladimir Vapnik's SVM combined with an optimization algorithm (Joachims, 1999).

SVMs like other inductive-learning approaches take as input a set of training examples (given as binary valued feature vectors) and finds a classification function that maps them to a class. There are several points about SVM models that are worth summarizing here. The first is that SVMs are known to robustly handle large feature sets and to develop models that maximize their generalizability. This makes them an ideal model for the NE+ task. Generalizability in SVMs is based on statistical learning theory and the observation that it is useful sometimes to misclassify some of the training data so that the margin between other training points is maximized (Cortes and Vapnik, 1995). This is particularly useful for real world data sets that often contain inseparable data points. Secondly, although training is generally slow, the resulting model is usually small and runs quickly as only the support vectors need to be retained, i.e. the patterns that help define the function which separates positive from negative examples. Thirdly is that SVMs are binary classifiers and so we need to combine SVM models to obtain a multi-class classifier.

Formally then we can consider the purpose of the SVM to be to estimate a classification function $f : \chi \rightarrow \{\pm 1\}$ using training examples from $\chi \times \{\pm 1\}$ so that error on unseen examples is minimized. The classification function returns either $+1$ if the test data is a member of the class, or -1 if it is not. Although SVMs learn what are essentially linear decision functions, the effectiveness of the strategy is ensured by mapping the input patterns χ to a feature space Γ using a nonlinear mapping function $\Phi : \chi \rightarrow \Gamma$. Since the algorithm is well described in the literature cited earlier we will focus our description from now on the features we used for

[1] Tiny SVM is available from http:// http://cl.aist-nara.ac.jp/ taku-ku/software/ TinySVM/

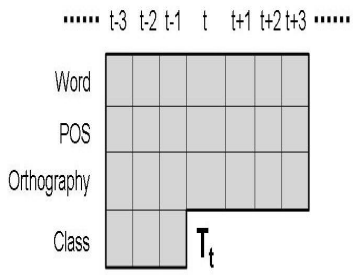

Figure 1: Lexical features and context considered by the SVM model when deciding the class tag T at the focus position t includes surface word forms, part of speech, orthographic features and previous word class tags.

training.

In our implementation each training pattern is given as a vector which represents certain lexical features. All models use a surface word, an orthographic feature (Collier et al., 2000) and previous class assignments, but our experiments with part of speech (POS) features (Brill, 1992) showed that POS features actually inhibited performance in the molecular biology data set which we present below. This is probably because the POS tagger was trained on news texts. Therefore POS features are used only for the MUC-6 news data set where we show a comparison with and without these features. The form of the vector is basically a bag of words, i.e. word positions or ordering are not recorded. In the experiments we report below we use feature vectors consisting of differing amounts of 'context' by varying the window around the focus word which is to be classified into one of the NE+ classes. The full window of context considered in these experiments is ± 3 about the focus word as shown in Figure 1. In pattern formation we took an IOB based approach to NE+ chunk identification in which each word was assigned a class tag from $\{I\_C_t, B\_C_t, O\}$ where C_t is the class, B stands for a beginning of chunk tag, I stands for an in-chunk tag, and O stands for outside of chunk, i.e. not a member of one of the given classes.

Due to the nature of the SVM as a binary classifier it is necessary in a multi-class task to consider the strategy for combining several classifiers. In our experiments with Tiny SVM the strategy used was one-against-one rather than one-against-the-rest. For example, if we have four classes A, B, C and D then Tiny SVM builds classifiers for (1) A against (B, C, D), (2) B against (C, D), and (3) C against D. The winning class is the one which obtains the most votes of the pairwise classifiers.

We implemented two versions of the SVM. SVM^1 uses a ± 3 window about the focus word and is implemented with the polynomial (poly) kernel function. SVM^2 is used to directly compare the performance of the SVM with the HMM model described below and here we use only features for the focus word and previous word, i.e. a more limited context. Due to effects of data sparseness the HMM would be very difficult to train using a wider context window - this is one of the advantages we hope to test in SVM^1.

The kernel function $k : \chi \times \chi \rightarrow \mathbf{R}$ mentioned above basically defines the feature space f by computing the inner product of pairs of data points. For $x \in \chi$ we explored the simple polynomial function $k(x_i, x_j) = (x_i \cdot x_j + 1)^d$.

2.2 HMM

The HMM we implemented for comparison with the SVM was the version fully described in (Collier et al., 2000)[2]. Basically this is a linear interpolating HMM trained using maximum likelihood estimates from bigrams of the surface word and an orthographic feature which is deterministically chosen. No part of speech was used in the formulation of this model.

We consider words to be ordered pairs consisting of a surface word, W, and a word feature, F, given as $< W, F >$. As is common practice, we need to calculate the probabilities for a word sequence for the first word's name class C_0 and every other word differently since we have no initial name-class to make a transition from.

[2]For purposes of comparison we note that in further tests Nobata *et al.* (2000) found this HMM to be superior to the C4.5 decision tree rule learner.

Accordingly we use the following equation to calculate the initial name class probability,

$$\begin{aligned} Pr(C_0| < W_0, F_0 >) = \\ \sigma_0 f(C_0| < W_0, F_0 >) + \\ \sigma_1 f(C_0| < \_, F_0 >) + \\ \sigma_2 f(C_0) \end{aligned} \quad (1)$$

and for all other words and their name classes C_t as follows:

$$\begin{aligned} Pr(C_t| < W_t, F_t >, < W_{t-1}, F_{t-1} >, C_{t-1}) = \\ \lambda_0 f(C_t| < W_t, F_t >, < W_{t-1}, F_{t-1} >, C_{t-1}) + \\ \lambda_1 f(C_t| < \_, F_t >, < W_{t-1}, F_{t-1} >, C_{t-1}) + \\ \lambda_2 f(C_t| < W_t, F_t >, < \_, F_{t-1} >, C_{t-1}) + \\ \lambda_3 f(C_t| < \_, F_t >, < \_, F_{t-1} >, C_{t-1}) + \\ \lambda_4 f(C_t|C_{t-1}) + \\ \lambda_5 f(C_t) \end{aligned} \quad (2)$$

where $f(|)$ is calculated with maximum-likelihood estimates from counts on training data.

In our current system we set the constants λ_i and σ_i by hand and let $\sum \sigma_i = 1.0$, $\sum \lambda_i = 1.0$, $\sigma_0 \geq \sigma_1 \geq \ \sigma_2$, $\lambda_0 \geq \lambda_1 \ldots \geq \lambda_5$. The current name-class C_t is conditioned on the current word and feature, the previous name-class, C_{t-1}, and previous word and feature.

Once the state transition probabilities have been calculated according to Equations 1 and 2, the Viterbi algorithm (Viterbi, 1967) is used to search the state space of possible name class assignments in linear time to find the highest probability path, i.e. to maximize $Pr(W, C)$.

2.3 Data Sets

We used two data sets in our study one for NE+ and the other for traditional NE. The NE+ collection (Bio1) consists of 100 MEDLINE abstracts (23586 words) in the domain of molecular biology annotated for the names of genes and gene products (Tateishi et al., 2000). The second (MUC-6) is the collection of 60 executive succession texts (24617 words) used in MUC-6 for dryrun and testing. Details are shown in Tables 1 and 2.

Class	#	Description
PROTEIN	2125	proteins, protein groups,families, complexes and substructures
DNA	358	DNAs, DNA groups, regions and genes
RNA	30	RNAs, RNA groups, regions and genes
SOURCE.cl	93	cell line
SOURCE.ct	417	cell type
SOURCE.mo	21	mono-organism
SOURCE.mu	64	multi-celled organism
SOURCE.vi	90	viruses
SOURCE.sl	77	sublocation
SOURCE.ti	37	tissue

Table 1: Markup classes used in Bio1 with the number of word tokens for each class.

3 Results and Analysis

Results are given as F-scores (van Rijsbergen, 1979) and calculated using the CoNLL evaluation script[3]. F-score is defined as $F = (2PR)/(P+R)$. where P denotes Precision and R Recall. P is the ratio of the number of correctly found NE chunks to the number of found NE chunks, and R is the ratio of the number of correctly found NE chunks to the number of true NE chunks.

Table 3 shows the overall F-score for the three models and two collections, calculated using 10-

[3]Available from http://lcg-www.uia.ac.be/conll2002/ner/bin/

Class	#
DATE	542
LOCATION	390
ORGANIZATION	1783
MONEY	423
PERCENT	108
PERSON	838
TIME	3

Table 2: Markup classes used in MUC-6 with the number of word tokens for class label.

fold cross validation on the total test collection. Due to the size of the collections we did not observe an optimal result for each model but we found a clear and sustained advantage by SVM[1] over the HMM for the NE task in MUC-6 and the NE+ task in Bio1. The only drawback we observed with SVM[2] was that it seemed to be quite weak for the very low frequency classes such as RNA, SOURCE.mo or TIME where the HMM usually proved to be more robust. SVM[2] was the weakest model that we tested and we can conclude that when trained with similar knowledge to the HMM the SVM has no particular performance advantage that we could observe. However by exploiting the SVMs capability to easily handle large feature sets including a wide context window and POS tags the results suggest that the SVM will perform at a significantly higher level than the HMM. A detailed break down of results by class is shown in Table 4.

What is not obvious from the tables is the effect we found of tokenization. In all the experiments reported for SVM in Table 3 we used the FDG parser (Tapanainen and Järvinen, 1997) which we found gave much better results for Bio1 than a simple tokenization strategy that simply split each word at spaces or punctuation marks. On MUC-6 the advantage was less clear and we concluded that the frequent and ambiguous use of hyphen in Bio1 was the key factor.

On the NE+ task in Bio1 we found that SVM[1] slightly but clearly outperformed the HMM. In analysis of SVM[1] results we identified several types of error. The first and perhaps most serious type was caused by local syntactic ambiguities such as head sharing in *39-kD SH2, SH3 domain* which should have been classed as a PROTEIN, but the SVM split it into two PROTEIN expressions *SH2* and *SH3 domain*. In particular the ambiguous use of hyphen, e.g. *14E1 single-chain (sc) Fv antibody* , and parentheses, e.g. *scFv (14E1)*, seemed to cause the SVM more difficulties than the HMM. It is likely that the limited contextual information we gave to the SVM was the cause of this and can be improved on using grammatical features such as head noun or main verb. HMM seems to gain an advantage through the Viterbi algorithm by being able to partially consider evidence over the entire sentence. A second minor type of error seemed to be the result of inconsistencies in the annotation scheme for Bio1 such as the inclusion of a definite description in a term name for *UT7/TPO cells, a thrombopoietin-dependent megakaryocytic cell line* which was all considered to be a SOURCE.ct expression.

4 Conclusion

There are many more kernel parameters to explore than can be dealt with here and we will be continuing our investigation by tuning the SVM parameters. The results do however provide an indication of performance trends: the first is that SVM will outperform the HMM by a significant margin on both the MUC-6 and Bio1 data sets if it is given a wide context window (± 3) and a rich feature set. The second is that the SVM lacked sufficient knowledge about complex structures in NE+ expressions to achieve its best performance on Bio1. We believe that with further tuning our SVM model will prove more useful in NE+ and allow us to combine evidence from large feature sets in order to model local structure and context. Furthermore, if a training set was developed for the POS tagger in the NE+ domain it seems likely that the SVM would strongly benefit from this. In its current configuration SVM[1] could be combined with the HMM on the Bio1 data set to achieve better performance in 5 of the 10 classes.

While acknowledging the danger of drawing broad conclusions about the NE+ task from one domain-based data set, pending further analysis, we can cautiously say that performance on the two data sets has shown that the MUC-6 NE task is somewhat easier than the Bio1 NE+ task. Despite a few investigations into the nature of the NE task (Palmer and Day, 1997) (Nobata et al., 2000) the information theoretical aspects of the knowledge required for the task are still not well understood and this must be considered as a key area for future research. In order to test our method more accurately and develop a composite model we are now building a more

Data set	Model					
	HMM	SVM^1(poly) degree d=				SVM^2 degree d=
		1	2	3	4	2
Bio1†	**70.97**	71.33	**71.78**	68.54	65.09	**65.63**
MUC-6†	**70.38**	72.86	**73.21**	69.22	65.12	**65.94**
MUC-6‡	–	**74.80**	74.66	72.68	67.92	**68.83**

Table 3: Overall F-scores for each of the learning methods on the two test sets using 10-fold cross validation on all data. SVM^1(poly) denotes the SVM trained using a polynomial kernel function and a ±3 context window; SVM^2 results when a context window of -1 and the focus were used so that direct comparisons with the HMM can be made. † Results for models using surface word and orthographic features but no part of speech features; ‡ Results for models using surface word, orthographic and part of speech features.

realistic data set for molecular biology from full journal articles.

Acknowledgements

This work was supported in part by the Japanese Ministry of Education and Science (grant no. 14701020). We would like to thank Jun-ichi Tsujii for providing the data set Bio1 and to the anonymous referees whose comments have helped to improve the paper.

References

D. Bikel, S. Miller, R. Schwartz, and R. Wesichedel. 1997. Nymble: a high-performance learning name-finder. In *Proceedings of the Fifth Conference on Applied Natural Language Processing (ANLP'97), Washington D.C., USA.*, pages 194–201, 31 March – 3 April.

E. Brill. 1992. A simple rule-based part of speech tagger. In *Third Conference on Applied Natural Language Processing – Association for Computational Linguistics, Trento, Italy*, pages 152–155, 31st March – 3rd April.

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

N. Collier, C. Nobata, and J. Tsujii. 2000. Extracting the names of genes and gene products with a hidden Markov model. In *Proceedings of the 18th International Conference on Computational Linguistics (COLING'2000), Saarbrucken, Germany*, July 31st–August 4th.

N. Collier, K. Takeuchi, C. Nobata, J. Fukumoto, and N. Ogata. 2002. Progress on multi-lingual named entity annotation guidelines using RDF(S). In *Proceedings of the Third International Conference on Language Resources and Evaluation (LREC'2002), Las Palmas, Spain*, pages 2074–2081, May 27th – June 2nd.

M. Collins and Y. Singer. 1999. Unsupervised models for named entity classification. In *Proceedings of the 1999 Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora*.

C. Cortes and V. Vapnik. 1995. Support-vector networks. *Machine Learning*, 20:273–297, November.

K. Fukuda, T. Tsunoda, A. Tamura, and T. Takagi. 1998. Toward information extraction: identifying protein names from biological papers. In *Proceedings of the Pacific Symposium on Biocomputing'98 (PSB'98)*, pages 707–718, January.

T. Joachims. 1999. Making large-scale SVM learning practical. In B. Scholkopf, C. Burges, and A. Smola, editors, *Advances in Kernel Methods - Support Vector Learning*. MIT Press.

T. Kudoh and Y. Matsumoto. 2000. Use of support vector learning for chunk identification. In *Proceedings of the Fourth Conference on Natural Language Learning (CoNLL-2000), Lisbon, Portugal*, pages 142–144.

DARPA. 1995. *Proceedings of the Sixth Message Understanding Conference(MUC-6)*, Columbia, MD, USA, November. Morgan Kaufmann.

C. Nobata, N. Collier, and J. Tsujii. 2000. Comparison between tagged corpora for the named entity

Class	Model					
	SVM[1](poly)[†] d=2			HMM		
	p	r	$F_{\beta=1}$	p	r	$F_{\beta=1}$
Data set Bio1						
PROTEIN	75.99	78.87	77.40	78.84	78.81	**78.82**
DNA	69.32	48.88	**57.33**	48.80	45.79	47.25
RNA	80.00	13.79	23.53	41.67	16.67	**23.81**
SOURCE.cl	75.00	48.91	**59.21**	50.00	38.71	43.64
SOURCE.ct	78.89	54.68	64.59	70.60	64.51	**67.42**
SOURCE.mo	0.00	0.00	0.00	77.78	35.00	**48.28**
SOURCE.mu	72.50	45.31	**55.77**	60.00	32.81	42.42
SOURCE.vi	91.80	60.87	**73.20**	78.08	62.64	69.51
SOURCE.sl	72.41	54.55	**62.22**	65.67	57.14	61.11
SOURCE.ti	50.00	2.70	5.13	47.06	21.62	**29.63**
All	75.89	68.09	**71.78**	73.10	58.95	70.97
Data set MUC-6						
DATE	84.52	62.56	**71.90**	74.47	61.67	67.47
LOCATION	67.78	48.22	**56.35**	66.23	40.32	50.12
ORG.	70.17	81.19	**75.28**	65.33	66.01	65.67
MONEY	83.11	76.88	**79.87**	77.71	80.62	79.14
PERCENT	75.68	51.85	61.54	85.48	98.15	**91.38**
PERSON	88.62	76.70	82.23	87.06	80.46	**83.63**
TIME	0.00	0.00	0.00	0.00	0.00	0.00
All	76.13	73.24	**74.66**	73.09	67.87	70.38

Table 4: Recall, precision and F-scores by class for Bio1 and MUC-6. Results were calculated using 10-fold cross validation on all the available data. [†] Results are shown for the best performing SVM models, i.e. without POS features for Bio1 and with POS features for MUC-6.

task. In A. Kilgarriff and T. Berber Sardinha, editors, *Proceedings of the Association for Computational Linguistics (ACL'2000) Workshop on Comparing Corpora, Hong Kong*, pages 20–27, October 7th.

D. Palmer and D. Day. 1997. A statistical profile of the named entity task. In *Proceedings of the Fifth Conference on Applied Natural Language Processing (ANLP'97), Washington D.C., USA.*, 31 March – 3 April.

L. Rabiner and B. Juang. 1986. An introduction to hidden Markov models. *IEEE ASSP Magazine*, pages 4–16, January.

P. Tapanainen and T. Järvinen. 1997. A non-projective dependency parser. In *Proceedings of the 5th Conference on Applied Natural Language Processing, Washington D.C., Association of Computational Linguistics*, pages 64–71.

Y. Tateishi, T. Ohta, N. Collier, C. Nobata, K. Ibushi, and J. Tsujii. 2000. Building an annotated corpus in the molecular-biology domain. In *COLING'2000 Workshop on Semantic Annotation and Intelligent Content, Luxemburg*, 5th–6th August.

C. J. van Rijsbergen. 1979. *Information Retrieval*. Butterworths, London.

V. N. Vapnik. 1995. *The Nature of Statistical Learning Theory*. Springer-Verlag, New York.

A. J. Viterbi. 1967. Error bounds for convolutions codes and an asymptotically optimum decoding algorithm. *IEEE Transactions on Information Theory*, IT-13(2):260–269.