

Language and Domain-Independent Named Entity Recognition: Experiment using SVM and High-Dimensional Features

Mona Soliman Habib and Jugal Kalita

Department of Computer Science, University of Colorado, 1420 Austin Bluffs Pkwy

Colorado Springs, CO 80917 USA

mhabib@eas.uccs.edu, kalita@eas.uccs.edu

Abstract— This paper presents the results of experiments aiming to explore a baseline solution for the named entity recognition problem using a language-independent, domain-independent approach. The first domain chosen for this experiment is the biomedical publications domain, especially selected due to its importance and inherent challenges. A supervised learning approach using Support Vector Machines (SVM) and high-dimensional features is used. Single-class and multi-class classification performance and results are examined. The approach used eliminates prior language knowledge such as part-of-speech or noun phrase tagging thereby allowing for its applicability across languages. No domain-specific knowledge is included. The initial results are comparable to those obtained using more complex approaches despite problems faced due to memory and computational power limitations. The multi-class $F_{\beta=1}$ score for the right boundary identification is 79.3%, for the left boundary identification is 69.9%, and the score for the semantic composite classification is 63.5%.

Index Terms— named entity recognition, support vector machines, language independence.

I. INTRODUCTION

NAMED entity recognition (NER) is one of the important tasks in information extraction, which involves the identification and classification of words or sequences of words denoting a concept or entity. Examples of named entities in general text are names of persons, locations, or organizations. Domain-specific named entities are those terms or phrases that denote concepts relevant to one particular domain. For example, protein and gene names are named entities which are of interest to the domain of molecular biology and medicine. The massive growth of textual information available in the literature and on the Web necessitates the automation of identification and management of named entities in text.

The task of identifying named entities in a particular language is often accomplished by incorporating knowledge about the language taxonomy in the method used. In the English language, such knowledge may include capitalization of proper names, known titles, common prefixes or suffixes, part of speech tagging, and/or identification of noun phrases in text. Techniques that rely on language-specific knowledge may not be suitable for porting to other languages. For example, the Arabic language does not use capitalization to identify proper names, and word variations are based on the

use of infixes in addition to prefixes and suffixes. Moreover, the composition of named entities in literature pertaining to specific domains follows different rules in each, which may or may not benefit from those relevant to general NER.

The following experiment attempts to eliminate language and domain-specific knowledge from the named entity recognition process when applied to the English biomedical entity recognition task, as a baseline for other languages and domains. The biomedical field NER remains a challenging task due to growing nomenclature, ambiguity in the left boundary of entities caused by descriptive naming, difficulty of manually annotating large sets of training data, strong overlap among different entities, to cite a few of the NER challenges in this domain.

II. EXPERIMENT AND RESULTS

A. Methods Used

The baseline experiment aims to identify biomedical named entities using a supervised learning approach. The training and testing data use the GENIA annotated corpus [5] of MEDLINE articles, where the names of proteins, cell lines, cell types, DNA and RNA entities are previously labeled. The named entities are often composed of a sequence of words. The training data includes 2,000 annotated abstracts (consisting of 492, 551 tokens). The testing data includes 404 abstracts (consisting of 101, 039 tokens) annotated for the same classes of entities: half of the test abstracts are from the same domain as the training data and the other half of them are from the super-domain of ‘blood cells’ and ‘transcription factors’. The testing data sets are grouped in four subsets, covering abstracts from different year ranges. The four subsets are: 1978-1989 set, 1990-1999 set, 2000-2001 set, and a subset that represents the super domain of ‘blood cells’ and ‘transcription factors’ from the years 1998-2001. The fraction of positive examples with respect to the total number of tokens in the training set varies from about 0.2% to about 6%. Basic statistics about the data sets as well as the absolute and relative frequencies for named entities within each set can be found in [6]. The approach employed in this experiment is the supervised machine learning using Support Vector Machines (SVM) [13], due to their ability to handle high-dimensional feature and input space.

The JNLPBA-04 shared task [6] is an open challenge task proposed at the “International Joint Workshop on Natural Language Processing in Biomedicine and its Application”. The task uses the GENIA corpus [5] described above. The systems participating in this task employ a variety of machine learning techniques such as Support Vector Machines (SVM), Hidden Markov Models (HMM), Maximum Entropy Markov Models (MEMMs) and Conditional Random Fields (CRFs). Five systems adopted SVMs either in isolation [7, 8], or in conjunction with other model [9, 11, 16]. The results of the experiment presented in this paper are compared to the five JNLPBA-04 task participating systems listed above, in addition to the results reported in [2]. Table I summarizes the performance comparison results.

This experiment is composed of two main parts – the first identifies protein named entities only, while the second locates and classifies all five named entities (protein, DNA, RNA, cell type, and cell line). The SVM^{light} software by T. Joachims [4] is used for the single-class part of the experiment, and the SVM^{Multiclass} software – also by T. Joachims – is used for the multi-class experiments. Tables II and III summarize the experiment results of both parts.

The training and test data pre-processing involves morphological and contextual features extraction only. In order to estimate a worse-case scenario of the approach used, no instance pruning or filtering is performed prior to learning and classification, thereby leaving the scarcity nature of the data intact. No language-specific pre-processing such as part-of-speech or noun phrases tagging is used. No dictionaries, gazetteers, or other domain-specific knowledge are used.

The initial results are promising and prove that the approach used is capable of recognizing and identifying the named entities successfully without making use of any language-specific or domain-specific prior knowledge. Performance measures of the experiments are reported in terms of recall, precision, and $F_{\beta=1}$ -score.

B. Features Selection

The training and testing data is preprocessed using the JFEX software [2] in order to extract morphological and contextual features that do not use language-specific knowledge such as part-of-speech or noun phrase tagging. The generated feature space is very large, including about a million different features. The feature extraction process is intentionally designed that way in order to test the scalability of the approach used and to allow the experiments to proceed in a language-independent and domain-independent fashion. All features are binary, i.e., each feature denotes whether the current token possesses this feature (one) or not (zero).

The morphological features extracted are:

- Capitalization: token begins with a capital letter.
- Numeric: token is a numeric value.
- Punctuation: token is a punctuation.
- Uppercase: token is all in uppercase.
- Lowercase: token is all in lowercase.

- Single character: token length is equal to one.
- Symbol: token is a special character.
- Includes hyphen: one of the characters is a hyphen.
- Includes slash: one of the characters is a slash.
- Letters and Digits: token is alphanumeric.
- Capitals and digits: token contains caps and digits.
- Includes caps: some characters are in uppercase.
- General regular expression summarizing the word shape, for e.g., $Xx+-X-x+$ describes a word starting with one capital letter followed by a number of lowercase letter, then a hyphen, one capital letter, another hyphen, and ending with a number of lowercase letters.

The morphological features extracted examine a token as a whole and do not include character n-grams features that detects in-word characteristics such as prefixes, suffixes, or infixes. Future work plans to extract these additional features in order to provide a finer level of word dissection. Morphological feature extraction was applied to the three tokens preceding and following the token being examined in addition to the token itself.

Each word appearing in the training text is considered its own feature. In addition, a consecutive collocation of tokens active over three positions around the token itself is used in order to provide a moving window of consecutive tokens which describes the context of the token relative to its surrounding.

Since biomedical named entities often are composed of more than one token, special labeling for the beginning, the middle, and the ending of a named entity sequence is often used. However, in this experiment, all tokens within a sequence use the same label during the SVM training phase. The output of the SVM classification phase is post-processed in order to label the beginning, middle, and end part of a sequence differently, as required by the JNLPBA-04 task evaluation scripts. Some minor errors in the post-processing labeling scripts caused some performance loss, however, the overall performance results are still indicative of the feasibility of the approach used.

C. Single-Class Results

Using SVM^{light} [4], a single-class support vector machine is trained to recognize protein name sequences. The trained machine is then used to classify proteins in the test data. Performance results of the protein classification task are summarized in table III. The hardware configuration used for single-class classification experiments is a single-processor Windows-based Pentium IV machine with 1 GB of RAM. This configuration was enough to complete each single-class experiment successfully within 20-30 minutes.

Since no pre-processing was performed on the training and testing data besides features extraction, the positive examples in the data sets remained scarce. As a result, we consider the performance results reported in table III to represent a worse-case indication of the potential performance.

SVM^{light} [4] offers the option of “boosting” the weight of the positive examples relative to the negative ones. We experimented with boosting factors of 2, 4, and 8 in order to counter the effect of positive data scarcity. The relative performance results are reported in table III. The overall recall, precision, and $F_{\beta=1}$ -score measures achieved with the different boosting factors is as follows:

No positive boosting	68.06 / 59.29 / 63.37
Positive boosting factor = 2	75.54 / 58.82 / 66.14
Positive boosting factor = 4	78.70 / 57.30 / 66.32
Positive boosting factor = 8	78.49 / 54.84 / 64.56

As expected, increasing the positive weight boosting factor led to an improved recall measure at the expense of the precision measure. The resulting $F_{\beta=1}$ -score improved with a boosting factor of 2 and 4 relative to the experiment without any positive weight boosting. However, a further increase of the positive boosting factor led to a decrease of the overall $F_{\beta=1}$ -score, as a result of the decreasing precision measure. A careful balance of the recall and precision results is required in order to maintain an overall $F_{\beta=1}$ -score, if such a measure is deemed to be the final performance indicator.

D. Multi-Class Results

The SVM^{Multiclass} implementation by T. Joachims is based on [1] and uses a different quadratic optimization algorithm described in [12]. The SVM^{Multiclass} implementation uses an “all-together” multi-classification approach, which is computationally more expensive yet usually more accurate than “one-against-all” or “one-against-one” multi-classification methods. Hsu and Lin [3] note that “as it is computationally more expensive to solve multi-class problems, comparisons of these methods using large-scale problems have not been seriously conducted. Especially for methods solving multi-class SVM in one step, a much larger optimization problem is required so up to now experiments are limited to small data sets.” The multi-class experiment presented in this paper is one such attempt at solving a large-scale problem using an “all-together” classification method.

Initial experiments for multi-class classification were unsuccessful, mostly due to hitting the processing power limits of the single processor machines. The same experiments were attempted on different machine configurations, and unreasonably long processing time was needed to finally complete one such experiment. The first successful experiment required a total learning and classification time of 17 days in order to complete using a serial algorithm on a quad-processor Pentium IV machine. The same experiment was repeated on a Linux machine with four Xeon 3.6 GHz processors and 896 Mbytes of main memory size and completed in 97 hours or four days and one hour.

The multi-class performance results are summarized in table II. The overall recall measure achieved is 62.43%, with a precision measure of 64.50%, and a final F-score of 63.45%. These results are compared to those obtained by the five

JNLPBA-04 participating systems which used support vector machines either in isolation or in combination with other models, as well as the results reported by [2] using the same task data. The performance comparison results are reported in table I. The language-independent approach used in this experiment performed very close to [8] and better than [7] which both used SVM as the only learning model. Park et al. [8] used character n-grams, orthographic information, word shapes, gene sequences prior knowledge, word variations, part-of-speech tags, noun phrase tags, and word triggers. Lee et al. [7] used lexical features, character n-grams, and part-of-speech tags in a two-phased model based on SVMs.

Rössler [9] adapted a NER-system for German to the biomedical field. The system used character n-grams, orthographic information, gene sequences prior knowledge, and word length as features. The overall performance of [9] is very close to that achieved in this experiment. The approach used by [9] is particularly interesting in demonstrating the applicability of some NER-system from one language to another by not incorporating language-specific features. However, Rössler [9] made use of domain-specific knowledge while applying the system to the biomedical domain.

Zhou and Su [16] developed the system that performed best in the JNLPBA-04 task. Their system performance reached an overall recall/precision/F-score of 76.0%, 69.4%, and 72.6% respectively. Zhou and Su [16] used SVM in conjunction with Hidden Markov Models in a more complex learning method. The systems also made use of many language-specific and domain-specific knowledge such as character n-grams, orthographic information, gene sequences, gazetteers, part-of-speech tags, word triggers, abbreviations, and cascaded entities. While this system performed better than the current multi-class experiment, its heavy use of language and domain-specific prior knowledge contradicts the promise of the approach presented in this paper.

Song et al. [11] used SVM in combination with Conditional Random Fields (CRFs) and included character n-grams, orthographic information, and other lexical features in addition to part-of-speech and noun phrase tagging. The overall performance of this system is comparable to that of the multi-class experiment results hereby presented. Giuliano et al. [2] also incorporated part-of-speech tagging and word features of tokens surrounding each analyzed token in addition to features similar to those used in this experiment. In addition, Giuliano et al. [2] pruned the data instances in order to reduce the dataset size by filtering out frequent words from the corpora because they are less likely to be relevant than rare words.

E. Challenges and Problems

The Java-based JFEX feature extraction system provided by [2] provides a scripted approach to define which features are to be extracted, which adds flexibility to the feature extraction process. However, the system memory requirements are very high, especially with a large dataset and a selection of several complex features. The memory requirements can be reduced

by breaking down the datasets into smaller sets grouped in a folder, yet similar memory needs reduction was not possible when complex features were to be extracted. For the sake of this experiment, word shape features for surrounding tokens could not be included due to the memory limitation. Also, extraction of character n-grams was not included in this system. The resulting set of features used in this experiment mostly includes simple orthographic information, contextual features, and the words themselves.

Different machine configurations were tried during the course of this research. Desktops and laptops with Pentium IV processors and 1-2 GB of RAM were used for single-class classification experiments. Attempts to run multi-class experiments failed on single processor machines due to the CPU-intensive nature of the classification process. The later experiment was tried on a Pentium III quad-processor Linux machine. The process did not complete in 22 days and was aborted. Only one successful multi-class experiment that was run on a Pentium IV quad-processor Linux-based machine completed in 17 days.

Some initial parallelization single-class experiments were attempted successfully on a cluster of Linux machines. The parallel SVM software developed by Zanni et al. [15] based on [10, 14] was used. However, the initial performance measures achieved by the serial SVM software were higher than those attained by the parallel algorithm, with no

considerable gain in computational time. Further experiments are needed in order to determine the best use of the parallel SVM software in [15] or other parallel algorithms.

III. CONCLUSION AND FUTURE WORK

Initial experiment results presented in this paper show that the proposed language and domain-independent approach is capable of successfully recognizing and classifying named entities with reasonable performance measures. These measures may be further improved by the inclusion of character n-grams and more robust post-processing scripts.

Recognizing named entities in a large set of sparse data using either classification or clustering techniques is a CPU and memory-intensive process. For future work, we would like to explore the parallelization of the multi-class classification approach in order to improve its scalability and performance, while taking advantage of the multiple classes and the possibility of data and process partitioning.

An extension of this research aims to develop an unsupervised learning solution for the named entity recognition problem, in order to overcome the inherent problems with manual annotation of large training data sets used for supervised learning solutions. A new technique for clustering data using support vectors is currently being explored.

TABLE I
PERFORMANCE OF JNLPBA SHARED TASK PARTICIPANT SYSTEMS VS. EXPERIMENT RESULTS

	1978-1989 Set	1990-1999 Set	2000-2001 Set	S/1998-2001 Set	Total
Zhou [16]	75.3 / 69.5 / 72.3	77.1 / 69.2 / 72.9	75.6 / 71.3 / 73.8	75.8 / 69.5 / 72.5	76.0 / 69.4 / 72.6
Song [11]	60.3 / 66.2 / 63.1	71.2 / 65.6 / 68.2	69.5 / 65.8 / 67.6	68.3 / 64.0 / 66.1	67.8 / 64.8 / 66.3
Rosslar [9]	59.2 / 60.3 / 59.8	70.3 / 61.8 / 65.8	68.4 / 61.5 / 64.8	68.3 / 60.4 / 64.1	67.4 / 61.0 / 64.0
Park [8]	62.8 / 55.9 / 59.2	70.3 / 61.4 / 65.6	65.1 / 60.4 / 62.7	65.9 / 59.7 / 62.7	66.5 / 59.8 / 63.0
Lee [7]	42.5 / 42.0 / 42.2	52.5 / 49.1 / 50.8	53.8 / 50.9 / 52.3	52.3 / 48.1 / 50.1	50.8 / 47.6 / 49.1
Baseline [6]	47.1 / 33.9 / 39.4	56.8 / 45.5 / 50.5	51.7 / 46.3 / 48.8	52.6 / 46.0 / 49.1	52.6 / 43.6 / 47.7
Giuliano [2]	--	--	--	--	64.4 / 69.8 / 67.0
Habib	53.2 / 70.8 / 60.7	63.7 / 63.6 / 63.7	64.2 / 65.4 / 64.8	63.0 / 63.2 / 63.1	62.4 / 64.5 / 63.5

TABLE II
MULTI-CLASS SVM EXPERIMENT RESULTS (RECALL / PRECISION / F-SCORE)

Named Entity	1978-1989 Set	1990-1999 Set	2000-2001 Set	S/1998-2001 Set	Total
protein	58.62 / 70.83 / 64.15	72.68 / 63.43 / 67.74	70.83 / 62.03 / 66.14	71.28 / 60.19 / 65.27	70.37 / 62.00 / 65.92
DNA	61.61 / 65.71 / 63.59	52.21 / 63.01 / 57.10	52.55 / 70.59 / 60.25	47.11 / 69.60 / 56.19	51.00 / 67.64 / 58.16
RNA	0.00 / 0.00 / 0.00	55.10 / 57.45 / 56.25	50.00 / 74.29 / 59.77	50.00 / 62.50 / 55.56	51.16 / 63.77 / 57.02
cell type	51.79 / 73.55 / 60.78	51.42 / 72.84 / 60.28	52.94 / 82.89 / 64.62	50.09 / 81.31 / 61.99	59.56 / 78.00 / 67.55
cell line	32.39 / 67.86 / 43.85	50.00 / 50.60 / 50.30	56.94 / 55.03 / 55.97	53.53 / 43.75 / 48.15	47.72 / 51.64 / 49.61
Overall	53.18 / 70.79 / 60.73	63.68 / 63.63 / 63.66	64.15 / 65.39 / 64.76	62.97 / 63.16 / 63.06	62.43 / 64.50 / 63.45
Correct Right	71.55 / 95.25 / 81.72	79.44 / 79.38 / 79.41	78.95 / 80.47 / 79.70	78.40 / 78.64 / 78.52	78.05 / 80.65 / 79.33
Correct Left	56.12 / 74.72 / 64.10	70.33 / 70.28 / 70.31	70.95 / 72.31 / 71.63	69.68 / 69.90 / 69.79	68.76 / 71.06 / 69.89

TABLE III
EFFECT OF POSITIVE EXAMPLES BOOSTING ON SINGLE-CLASS (PROTEIN) SVM RESULTS

	1978-1989 Set	1990-1999 Set	2000-2001 Set	S/1998-2001 Set	Total
No Boosting Complete	57.47 / 53.35 / 55.34	71.69 / 62.76 / 66.93	68.35 / 59.60 / 63.68	68.27 / 58.63 / 63.08	68.06 / 59.29 / 63.37
Right	73.56 / 68.29 / 70.83	81.76 / 71.58 / 76.33	79.04 / 68.92 / 73.63	79.47 / 68.25 / 73.43	79.30 / 69.09 / 73.84
Left	59.61 / 55.34 / 57.39	77.82 / 68.13 / 72.65	76.70 / 66.88 / 71.45	76.08 / 65.34 / 70.30	75.24 / 65.55 / 70.06
Boost Factor = 2 Complete	65.35 / 50.83 / 57.18	78.59 / 60.88 / 68.61	76.33 / 60.36 / 67.41	75.58 / 58.40 / 65.89	75.54 / 58.82 / 66.14
Right	80.62 / 62.71 / 70.55	87.75 / 67.98 / 76.61	86.28 / 68.23 / 76.20	86.25 / 66.65 / 75.19	86.09 / 67.04 / 75.38
Left	67.98 / 52.87 / 59.48	85.70 / 66.39 / 74.82	83.67 / 66.16 / 73.89	83.21 / 64.30 / 72.54	82.57 / 64.30 / 72.30
Boost Factor = 4 Complete	71.43 / 48.49 / 57.77	81.55 / 59.32 / 68.68	78.99 / 59.03 / 67.57	78.63 / 57.04 / 66.11	78.70 / 57.30 / 66.32
Right	84.73 / 57.53 / 68.53	89.93 / 65.42 / 75.74	88.58 / 66.20 / 75.77	88.64 / 64.30 / 74.53	88.55 / 64.46 / 74.61
Left	74.38 / 50.50 / 60.16	88.66 / 64.50 / 74.67	86.10 / 64.35 / 73.65	86.00 / 62.39 / 72.31	85.58 / 62.31 / 72.11
Boost Factor = 8 Complete	71.43 / 44.52 / 54.85	81.41 / 57.14 / 67.15	78.76 / 56.91 / 66.08	78.34 / 54.74 / 64.45	78.49 / 54.87 / 64.59
Right	85.06 / 53.02 / 65.32	90.14 / 63.27 / 74.35	88.21 / 63.74 / 74.00	88.42 / 61.78 / 72.73	88.41 / 61.81 / 72.76
Left	73.89 / 46.06 / 56.75	88.59 / 62.18 / 73.08	86.47 / 62.48 / 72.54	86.44 / 60.39 / 71.11	85.83 / 60.00 / 70.63

REFERENCES

- [1] K. Crammer and Y. Singer, "On the Algorithmic Implementation of Multi-class SVMs," *Journal of Machine Learning Research*, vol. 2, pp. 265–292, 2001.
- [2] C. Giuliano, A. Lavelli, et al., "Simple Information Extraction (SIE)," ITC-irst, Istituto per la Ricerca Scientifica e Tecnologica, 2005.
- [3] C.-W. Hsu and C.-C. Lin, "A Comparison of Methods for Multi-Class Support Vector Machines," *IEEE Transactions on Neural Networks*, vol. 13, pp. 415-425, 2002.
- [4] T. Joachims, *Learning to Classify Text Using Support Vector Machine*. Norwell, MA: Kluwer Academic, 2002.
- [5] J. D. Kim, T. Ohta, et al., "GENIA Corpus--Semantically Annotated Corpus for Bio-Textmining," *Bioinformatics*, vol. 19 Suppl 1, pp. 180-182, 2003.
- [6] J.-D. Kim, T. Ohta, et al., "Introduction to the Bio-Entity Recognition Task at JNLPBA," in Proc. of the 2004 Joint Workshop on Natural Language Processing in Biomedicine and its Applications (JNLPBA'2004), Geneva, Switzerland, 2004.
- [7] K. J. Lee, Y. S. Hwang, et al., "Biomedical Named Entity Recognition using Two-Phase Model Based on SVMs," *Journal of Biomedical Informatics*, vol. 37, pp. 436-447, 2004.
- [8] K.-M. Park, S.-H. Kim, et al., "Incorporating Lexical Knowledge into Biomedical NE Recognition," in Proc. of the 2004 Joint Workshop on Natural Language Processing in Biomedicine and its Applications (JNLPBA'2004), Geneva, Switzerland, 2004.
- [9] M. Rössler, "Adapting an NER-System for German to the Biomedical Domain," in Proc. of the 2004 Joint Workshop on Natural Language Processing in Biomedicine and its Applications (JNLPBA'2004), Geneva, Switzerland, 2004.
- [10] T. Serafini, G. Zanghirati, et al., "Gradient Projection Methods for Large Quadratic Programs and Applications in Training Support Vector Machines," *Optimization Methods and Software*, vol. 20, pp. 353-378, 2005.
- [11] Y. Song, E. Kim, et al., "POSBIOTM-NER in the Shared Task of BioNLP/NLPBA 2004," in Proc. of the 2004 Joint Workshop on Natural Language Processing in Biomedicine and its Applications (JNLPBA'2004), Geneva, Switzerland, 2004.
- [12] I. Tsochantaridis, T. Hofmann, et al., "Support Vector Learning for Interdependent and Structured Output Spaces," in Proc. of the 21st International Conference on Machine Learning (ICML), Alberta, Canada, 2004.
- [13] V. N. Vapnik, *The Nature of Statistical Learning Theory*: Springer, 1995.
- [14] G. Zanghirati and L. Zanni, "A Parallel Solver for Large Quadratic Programs in Training Support Vector Machines," *Parallel Computing*, vol. 29, pp. 535-551, 2003.
- [15] L. Zanni, T. Serafini, et al., "Parallel Software for Training Large Scale Support Vector Machines on Multiprocessor Systems," *Journal of Machine Learning Research*, vol. 7, pp. 1467-1492, 2006.
- [16] G. Zhou and J. Su, "Exploring Deep Knowledge Resources in Biomedical Name Recognition," in Proc. of the 2004 Joint Workshop on Natural Language Processing in Biomedicine and its Applications (JNLPBA'2004), Geneva, Switzerland, 2004.