Recognition of Protein/Gene Names from Text using an Ensemble of Classifiers and Effective Abbreviation Resolution

Zhou GuoDong [1], Shen Dan [1], Zhang Jie [1], Su Jian [1], Tan Soon Heng [1] and Tan Chew Lim [2]
[1] Institute for Infocomm Research, 21 Heng Mui Keng Terrace, Singapore 119613
[2] School of Computing, National Univ. of Singapore, Singapore 119610
[1] Email: zhougd, shendan, zhangjie, sujian, soonheng@i2r.a-star.edu.sg;
[2] Email: tancl@comp.nus.edu.sg

Abstract

In this paper, we propose an ensemble of classifiers for biomedical named entity recognition in which three classifiers (one SVM and two HMMs) are combined effectively using a simple majority voting strategy. In addition, we incorporate an abbreviation resolution module, a protein/gene name refinement module and a simple dictionary matching module into the system to further improve the performance. Evaluation shows that our system achieves best performance (F-measure 82.58) on the closed test of the BioCreative protein/gene name recognition task (Task 1A).

1 Feature Representation

In the competition, the following five features are applied to capture the special characteristics of protein/gene names:

- **Surface Word**: For example, if a word occurs in a vocabulary, one dimension in the feature vector of the SVM (corresponding to the position of the word in the vocabulary) is set to 1. The vocabulary is constructed by taking all the words in the training data (filtered with threshold 3).
- **Orthographic Feature**: This feature concerns about capitalization, digitalization and word formation information. Table 1 shows a complete list in the descending order of priority.

Table 1: Orthographic Features

Features 1-11	*e.g.*	Features 12-21	*e.g.*
Comma	*,*	OneCap	*T*
Dot	*.*	AllCaps	*CSF*
Parenthesis	*( ) [ ]*	CapLowAlpha	*All*
RomanDigit	*II*	CapMixAlpha	*IgM*
GreekLetter	*Beta*	LowMixAlpha	*kDa*
StopWord	*in, at*	AlphaDigitAlpha	*H2A*
ATCGsequence	*ACAG*	AlphaDigit	*T4*
OneDigit	*5*	DigitAlphaDigit	*6C2*
AllDigits	*60*	DigitAlpha	*19D*
DigitCommaDigit	*1,25*	Others	*Other*
DigitDotDigit	*0.5*		

- **POS**: A HMM-based POS tagger is trained to assign the POS feature. Throughout the competition, three versions of POS taggers are trained on different corpora:
  - GENIA-POS tagger, which is trained on GENIA V3.02p. This POS tagger is used in the SVM and HMM1 of the closed evaluation 1.
  - BioCreative-POS tagger, which is trained on the BioCreative corpus with the NEWGENE tag replaced by the NNP tag. This POS tagger is only used in the HMM2 of all the evaluations (closed evaluations 1, 2, 3 and open evaluation 1).
  - Refined-BioCreative-POS tagger, which is trained on a refined version of the BioCreative corpus. In the competition, the refined BioCreative corpus is created as follows: First, a BioCreative-POS tagger is trained as above; Second, the protein/gene names in the

BioCreative corpus is retagged using the BioCreative-POS tagger; Third, the words in the protein/gene names are all set to have the NNP tag when

- they are the head nouns of the protein/gene names
- they are capitalized
- they include dashes
- they include both alphas and digits

This POS tagger is used in the SVM and HMM1 of the closed evaluations 2, 3 and open evaluation 1.

- **Suffix**: Suffixes, such as ~ase, ~zyme, ~ome and ~gen, occur frequently in protein/gene names. In the meantime, some common words (58 in the competition) of these suffixes, such as *disease*, *base*, case and *come,* are filtered to reduce the noise.
- **Trigger**: We use two kinds of trigger words: *TW1* often occurs inside protein/gene names and *TW2* often occurs in the local context of protein/gene names. *TW1* is collected based on the *Task 1A Guideline*, such as *receptor*, *enhancer*, *mutant*, etc. *TW2* is extracted automatically from the training data, such as *activation*, *transcription and stimulation* using the tf-idf weighing scheme [Salton and Buckley 1990] to measure how specific a given trigger word is to protein/gene names. Here, the tf-idf value is used in the feature vector of the SVM. In the competition, 53 *TW1* and 51 *TW2* triggers are used.

2 Support Vector Machine

Support Vector Machine (SVM) is a powerful machine learning method, which has been applied successfully in biomedical named entity recognition [Kazama et al. 2002; Lee et al. 2003]. SVM is a binary classifier and training a SVM classifier is to find the optimal hyper-plane that separates positive and negative data with the maximum margin. This optimal hyper-plane is then used to classify test data: those lying on one side of the hyper-plane are classified as the positive class, while others are classified as the negative class. Each instance in the training and test data is represented using a high-dimensional vector. Here, SVMLight [Joachims 1999] is used.

Since there is only one name class protein/gene (NEWGENE) in the BioCreative protein/gene recognition task (Task 1A), we simplify the traditional *BIO* representation and employ *IO* tags to represent the regional information of NEWGENE names. In this *IO* representation, *I* means that current word is a part of a protein/gene name, which corresponds to the SVM output 1; *O* means that current word is not a part of a protein/gene name, which corresponds to the SVM output -1. After the simplification, the protein/gene recognition task becomes a binary classification task. Although the *IO* representation cannot differentiate consecutive names, it simplifies the problem a lot since we can avoid the SVM multi-class problem. We find it is a worth tradeoff.

In SVM, each word is represented as a feature vector. A window of a target word *w* represents the local context of *w* and is used to make a decision on *w*. In this task, we set the window size to 7, which includes the previous 3 words and the next 3 words of the target word *w* including the target word *w* itself. In our system, all the five features as described above are applied for each of the 7 words in the window. Especially, when a word contains dashes, one additional overlapping orthographic feature is generated for each segment separated by dashes.

2 Hidden Markov Model

A Hidden Markov Model (HMM) is a model where a sequence of observation is generated in addition to the Markov state sequence. It is a latent variable model in the sense that only the observation sequence is known while the state sequence remains “hidden”. HMM has been widely used in named entity recognition [Bikel 1999; Zhou et al 2002; Zhang et al 2004; Zhou et al 2004]

Given an observation sequence $o_1^n = o_1 o_2 \cdots o_n$, the goal of a HMM is to find a stochastic optimal tag (state) sequence $s_1^n = t_1 t_2 \cdots t_n$ that maximizes $\log P(s_1^n \mid o_1^n)$: (Zhou et al 2002)

$$\begin{aligned} s^* &= \arg\max_{s_1^n} \{\log P(s_1^n \mid o_1^n)\} \\ &= \arg\max_{s_1^n} \{\log P(s_1^n) + MI(s_1^n, o_1^n)\} \end{aligned} \quad (1)$$

Obviously, the second term $MI(s_1^n, o_1^n)$ captures the mutual information between the state sequence s_1^n and the observation sequence o_1^n. To compute $MI(s_1^n, o_1^n)$ efficiently, we propose a novel mutual information independence assumption:

$$MI(s_1^n, o_1^n) = \sum_{i=1}^{n} MI(s_i, o_1^n) \text{ or } \log \frac{P(s_1^n, o_1^n)}{P(s_1^n) \cdot P(o_1^n)} = \sum_{i=1}^{n} \log \frac{P(s_i, o_1^n)}{P(s_i) \cdot P(o_1^n)} \quad (2)$$

That is, we assume a state is only dependent on the observation sequence o_1^n and independent on other states in the state sequence s_1^n. This assumption is reasonable because the dependence among the states in the state sequence s_1^n has been directly captured by the first term $\log P(s_1^n)$ in equation (1).

By applying the assumption (2) into the equation (1), we have:

$$s^* = \arg\max_{s_1^n} \{\sum_{i=2}^{n} MI(s_i, s_1^{i-1}) + \sum_{i=1}^{n} \log P(s_i \mid o_1^n)\} \quad (3)$$

The above model consists of two models: the state transition model $\sum_{i=2}^{n} MI(s_i, s_1^{i-1})$ which measures the state dependence of a state given the previous states, and the output model $\sum_{i=1}^{n} \log P(s_i \mid o_1^n)$ which measures the observation dependence of a state given the observation sequence in a discriminative way.

Computation of the above model consists of two parts. The first is to compute the state transition model: $\sum_{i=2}^{n} MI(s_i, s_1^{i-1})$. Here, the traditional ngram modeling (e.g. trigram) is used. The second is to estimate the output model: $\sum_{i=1}^{n} \log P(s_i \mid o_1^n)$. Here, a dynamic back-off modeling (Zhou et al 2004) is applied.

In the competition, only three features are used in the HMM: the orthographic feature, the POS feature and the surface word as described above. All the three features are combined and become an observation of HMM while each tag is structural indicating the position of the word, the features of the word and whether the word locates inside or outside a protein/gene name [Zhou et al 2004].

3. Ensemble of Classifiers

In the competition, an ensemble of classifiers is proposed for the BioCreative protein and gene name recognition task in which three classifiers (one SVM and two HMMs) are combined effectively using a simple majority voting strategy. The only difference between the two HMMs comes from the POS features which are trained on different corpora.

The main reason for integrating SVM, HMM1 and HMM2 as an ensemble is the finding during our investigation that they have quite different characteristics regarding the precision and the recall. Our evaluation on the dryrun data shows that SVM has high precision and low recall, the HMM1 using the GENIA-POS tagger has balanced precision and recall, and the HMM2 using the BioCreative-POS tagger has low precision and high recall. Such specialty means big complement

among SVM, HMM1 and HMM2, which shows the potential for significant performance improvement via an ensemble.

4. Post Processing

4.1 Abbreviation Resolution

Task 1A Guideline emphasizes that when an abbreviation is given in the sentence, it always implies certain words are necessary to be an entity and these words should be included in the protein/gene name. Therefore, a protein/gene name defining an abbreviation should have the right boundary. During our experimentation, we find that many classification errors occur in the words around parentheses which introduce an abbreviation. In order to solve this problem, we use a rule-based algorithm to identify the abbreviations [Schwartz et al 2003]. Abbreviation candidates are determined by their adjacency to parentheses. In this way, we can correct the boundary error of the long form which defines an abbreviation. In addition, we can classify the abbreviation according to the prediction of its long form, since we assume it is more accurate to classify the long form than the abbreviation.

4.2 Refinement of Protein/Gene Names

This module applies some heuristic rules to refine the recognized protein/gene names:

- Removing stop words, e.g. "by" and "or", from the recognized names, which have been wrongly regarded as a part of the names.
- Formalizing the recognition of slash and parentheses
- Extending recognized names by adding positive trailer words. For example, If only "p53" in "p53 mutant" is recognized as a protein/gene name, we will add "mutant" into the name since "mutant" is a positive trailer word. Similarly, we also shorten the recognized names by removing negative trailer words.
- Removing generic terms (all the words used in the name are too common), e.g. "protein kinase".
- Removing odd names, such as individual digits and Greek letter.
- Recovering errors caused by wrong tokenization of "." in the recognized protein/gene names, e.g. "UL3 **.** 5" and "E **.** coli RNase H".
- Removing generic adjective words, e.g. "new" and "novel" in the beginning of recognized protein/gene names

4.3 Simple Dictionary Matching

In our closed system, the dictionary is constructed by extracting all protein/gene names from the training data. In our open system, the dictionary adds more protein/gene names (~700,000 entries) from public resources (e.g. Swissport). Then, the dictionary is filtered using some criteria in order to reduce noise. For examples, if a name consists of only one word, the length of the word must be greater than 3. Finally, we use the dictionary to match the test data and correct the output of the ensemble.

5. Evaluation

All the final open and closed systems are trained on the combined training and dryrun data (10000 sentences). Evaluation on the formal test data (5000 sentences) of the protein/gene name recognition task (Task 1A) shows that our closed system performs best among all the closed systems with F-measure of 82.58 which is 0.4 and 2.2 higher than the second and third best system.

Table 2 shows the different configurations for all the evaluations and the contributions of the differences (in Bold, compared with the left configuration) among the configurations. It shows that the POS tagger trained on the refined version of the BioCreative corpus works better (+0.40) than the POS tagger trained on GENIA V3.02 (closed-2 vs. closed-1). This may be because the refined BioCreative corpus is more task-oriented than the GENIA V3.02. It also shows that the protein/gene name refinement increases the F-measure by 2.35 (closed-3 vs. closed-2). However, Evaluation of the

open system shows that extra protein/gene names from public resources decrease the performance by 4.52 (open-1 vs. closed-3). This is largely due to the short time spent on the open system (half day). This indicates that proper handling of public resources is important for performance improvement.

Table 2: Configurations of all the evaluations in the protein/gene name recognition task (Task 1A)

Modules	Closed-1	Closed-2	Closed-3	Open-1
SVM	Surface word, orthographic feature, suffix, trigger			
	GENIA-POS	**Refined-BioCreative-POS**	Refined-BioCreative-POS	Refined-BioCreative-POS
HMM1	Surface word, orthographic feature,			
	GENIA-POS	**Refined-BioCreative-POS**	Refined-BioCreative-POS	Refined-BioCreative-POS
HMM2	Surface word, orthographic feature, BioCreative-POS			
Ensemble	Majority Voting			
Abbreviation Res.	Abbreviation Resolution based on the parentheses structure			
Refinement of protein/gene names	N/A	N/A	**YES**	N/A
Dictionary Matching	Closed dictionary	Closed dictionary	Closed dictionary	**Open Dictionary**
Overall Performance	P79.97 R80.15 F80.23	P80.46 R80.80 F80.63(**+0.40**)	P82.00 R83.17 F82.58(**+2.35**)	P75.10 R81.26 F78.06(**-4.52**)

Reference

Bikel M.D., Schwartz R. and Weischedel M.R. 1999. An Algorithm that Learns What's in a Name. *Machine Learning (Special Issue on NLP)*. 34(3). 211-231.

Joachims T. 1999. Making large-Scale SVM Learning Practical. *Advances in Kernel Methods - Support Vector Learning*, B. Schukopf and C. Burges and A. Smola (ed.), MIT-Press.

Kazama J., Makino T., Ohta Y., and Tsujii J. 2002. Tuning Support Vector Machines for Biomedical Named Entity Recognition. In *Proc. of the Workshop on Natural Language Processing in the Biomedical Domain (at ACL'2002)*, 1-8.

Lee K.J. Hwang Y.S. and Rim H.C. Two-phase biomedical NE Recognition based on SVMs. In *Proceedings of the ACL'2003 Workshop on Natural Language Processing in Biomedicine*. pp.33-40. Sapporo, Japan.

Salton G. and Buckley C. 1990. Improving retrieval performance by relevance feedback. *Journal of American Society for Information Systems*. 41: 288-297.

Schwartz A.S. and Hearst M.A. 2003. A simple algorithm for identifying abbreviation definitions in biomedical text. in *Proceedings of the Pacific Symposium on Biocomputing (PSB 2003)* Kauai.

Zhou G.D. and Su J. 2002. Named Entity Recognition using an HMM-based Chunk Tagger. In *Proc. of the 40th Annual Meeting of the Association for Computational Linguistics (ACL)*, 473-480.

Zhou GuoDong, Zhang Jie, Su Jian, Shen Dan and Tan ChewLim, 2004. Recognizing Names in Biomedical Texts: a Machine Learning Approach. *Bioinformatics* (to appear)

Zhang Jie, Shen Dan, Zhou GuoDong, Su Jian and Tan Chew Lim, 2004. Enhancing HMM-based Biomedical Named Entity Recognition by Studying Special Phenomena, *Journal of Biomedical Informatics* (to appear).