Identifying Gene and Protein Mentions in Text Using Conditional Random Fields

Ryan McDonald and Fernando Pereira
Department of Computer and Information Science
University of Pennsylvania
Levine Hall, 3330 Walnut Street, Philadelphia, PA 19104
{ryantm,pereira}@cis.upenn.edu

1 Introduction

Applying information extraction techniques in the biological domain has been a growing research area over the past few years. Numerous large scale copora have been developed [10] or are being developed [4] to facilitate this process. Typically, the first step in most information extraction systems is to identify the named entities that are relevant to the concepts, relations and events described in the text. In molecular biology, named entities related to genes, proteins or other biologically-active molecules are especially important.

Approaches to biological entity detection span a broad range from linguistic rule-based [9] to pure machine learning [3], as well as hybrids such as Tanabe and Wilbur's [12] that integrate an initial stochastic part-of-speech tagger that identifies candidate genes using a special 'GENE' part-of-speech and with post-processing stage that uses a series of rules based on collected lexicons.

In this paper we present a method for identifying gene and protein mentions in text with conditional random fields (CRFs) [5], which are discussed in Section 2. Narayanaswamy et al. [9] suggest that rule-based systems make decisions on sets of textual indicator features and that these features may easily be exploited by supervised statistical approaches. Our model does just this by incorporating many of the post-processing steps of Tanabe and Wilbur [12] into features of our model. This aspect of the system will be discussed in Section 3. Results on the development and evaluation data are presented in Section 4. The model we present is general in that it may be extended to various other biological entities, providing the appropriate lexicons are available.

The training, development and evaluation data for our system was provided by the organizers of BioCreative 2004 [1].

2 Conditional Random Fields

The task of identifying gene mentions in text can be represented as a tagging task, in which each text token is labeled with a tag indicating whether the token begins, continues, or is outside of a gene mention. Conditional random fields are probabilistic tagging models that give the conditional probability of a possible tag sequence $\mathbf{t} = t_1, \ldots t_n$ given the input token sequence $\mathbf{o} = o_1, \ldots o_n$:

$$P(\mathbf{t}|\mathbf{o}) = \frac{e^{\sum_j \sum_i \lambda_i f_i(s_j, \mathbf{o}, j)}}{Z(\mathbf{o})}$$

In this definition, each f_i is a function that measures a feature relating the state s_j at position j with the input sequence around position j, λ_j is the corresponding feature weight, and the state $s_j = (t_{j-k+1}, \ldots, t_j)$ encodes the tag k-gram ending at position j, for a suitable choice of k. We use $k = 3$ in this work, but some feature functions may ignore t_{j-2} or t_{j-2} and t_{j-1} to provide a form of back-off for rarely occurring tag trigrams or bigrams.

We represent the input text with a sequence of sets of predicates. The jth set contains those predicates that hold of the jth token. For instance, if token j is the word "Kinase", then the jth set might contain the predicates *WORD=**kinase*** and *WordIsCapitalized*, among others. The tag sequence uses the possible tags *B-GENE*, *I-GENE* and *O*, representing the beginning, inside and outside of a gene mention respectively. As noted before, each feature function relates input properties and a k-gram of tags. We use only binary features, for instance:

$$f_i(s, \mathbf{o}, j) = \begin{cases} 1 \text{ if 'WORD=}\texttt{kinase}\text{'} \in o_j, \\ \quad \text{tag}_{-1}(s) = \text{B-GENE}, \text{tag}_0(s) = \text{I-GENE}; \\ 0 \text{ otherwise.} \end{cases}$$

where $\text{tag}_{-k}((\ldots, t_{j-k}, \ldots)) = t_{j-k}$.

The weight λ_i for each feature should ideally be highly positive for features that are on in correct taggings, highly negative for features that tend be off in correct taggings, and around zero for uninformative features. To achieve this, the model is trained so that the weights maximize the log-likelihood of the training data, $\mathcal{T}$:

$$\mathcal{L}(\mathcal{T}) = \sum_{(\mathbf{t},\mathbf{o}) \in \mathcal{T}} \log P(\mathbf{t}|\mathbf{o})$$

subject to a Gaussian penalty ($\sigma = 1.0$) on weights to control overfitting [2]. This is a standard approach for log-linear models like the present one; many iterative algorithms are known for finding the optimal weight setting [6]. Once the optimal weight setting is found, then one can find the tag sequence with highest probability for an unlabeled input sequence using Viterbi's algorithm [5].

Our system uses the MALLET [8] implementation of CRFs trained with limited-memory quasi-Newton [11]. Furthermore, the model we use is trained using feature induction [7] which is described in greater detail in the next section.

3 Feature Set

Feature-based models like CRFs are attractive because they reduce each problem to that of finding a feature set that adequately represents the task at hand. As a good starting point, our feature set consisted of word as well as orthographic features (outlined in table 1). We then added character-n-gram features for $2 \leq n \leq 4$. These features help the system recognize informative substrings (e.g. 'homeo' or 'ase') in words that were not seen in training. In addition to the character-n-gram features, we also included word prefix and suffix features of the same lengths. This may seem redundant, but prefix and suffix features also take into account the position of the n-gram in the word. We include features that indicate whether the current token occurs within brackets or inside quotations. Finally, we made the window for all features {-1,1}. This means that features for token j would contain predicates about tokens j-1 and j+1.

Even with this very simple set of general features, performance on the development data was reasonable (see Section 4). In order to add expert knowledge to the model, we focused our attention on the gene and protein tagger *ABGene* [12]. ABGene is a hybrid model that uses a statistical part-of-speech tagger to identify candidate genes by labeling them with a special part-of-speech 'GENE'. Once the candidate genes are found a series of post processing rules are initiated to improve the results. Specifically, ABGene uses a set of lexicons to remove false positives and recover false negatives. These include general biological terms, amino acids, restriction enzymes, cell lines, organism names and non-biological terms meant to identify tokens that have been mislabeled as 'GENE'. To recover false negatives, ABGene utilizes large gene lexicons coupled with context lists to identify possible mentions. Another post-processing step identifies tokens that contain low frequency trigrams, compiled from MEDLINE, to identify possible gene candidates, since gene and proteins names often contain unusual character trigrams.

A straightforward method of integrating these post processing steps into our model is to create predicates indicating whether a token occurs in a one of the ABGene lexicons. For multi-token entries, we required that all tokens of the entry were matched. These features were also applied over a window of {-1,1}. Section 4 discusses the effect of adding these lexicons to our system.

3.1 Feature Induction

So far we have only described features over a single predicate. Often it is useful to create features based on the conjunction of several predicates. For instance, the following feature would be useful:

$$f_k(s, \mathbf{o}, j) = \begin{cases} 1 \text{ if 'WORD=}\texttt{p-53}\text{'} \in o_j, \\ \quad \text{'WORD=}\texttt{mutant}\text{'} \in o_{j+1} \\ \quad \text{tag}_0(s) = \text{B-GENE}; \\ 0 \text{ otherwise.} \end{cases}$$

This is because the token `p-53` can either be in a gene mention (when it is followed by the word 'mutant') or be in a mutation mention (when it is followed by the word 'mutations'). Hence the following feature would also be useful:

$$f_k(s, \mathbf{o}, j) = \begin{cases} 1 \text{ if 'WORD=}\texttt{p-53}\text{'} \in o_j, \\ \quad \text{'WORD=}\texttt{mutations}\text{'} \in o_{j+1}, \\ \quad \text{tag}_0(s) = \text{O}; \\ 0 \text{ otherwise.} \end{cases}$$

The system already has tens of thousands of singleton features, making it infeasible to create all such conjunctions. Even if it were computationally feasible to train a model with all conjunctions, it would be difficult to gather sufficient statistics on them since most conjunctions occur rarely if ever.

To solve this problem, McCallum [7] describes an implementation of feature induction for CRFs that automatically creates a set of useful features and feature conjunctions. Feature induction works by iteratively considering sets of candidate singleton and conjunction features that are created from the initially defined set of singleton features as well as the set of current model features. The log-likelihood gain on the training data is then measured for each feature individually. Only those candidates causing the highest gain are included into the current set of model features. Our experiments showed that using feature induction improved performace over just using all defined singleton features (Section 4).

4 Results and Discussion

Our system was initially trained on 7500 annotated MEDLINE sentences with a development set of 2500 sentences. Training with feature induction took approximately 22 hours, which is substantially longer than training without feature induction. Once trained, the system can annotate sentences in less than a second. For evaluation, we added the

Orthographic Feature	Reg. Exp.
Init Caps	[A-Z].*
Init Caps Alpha	[A-Z][a-z]*
All Caps	[A-Z]+
Caps Mix	[A-Za-z]+
Has Digit	.*[0-9].*
Single Digit	[0-9]
Double Digit	[0-9][0-9]
Natural Number	[0-9]+
Real Number	[-0-9]+[.,]+[0-9.,]+
Alpha-Num	[A-Za-z0-9]+
Roman	[ivxdlcm]+ or [IVXDLCM]+
Has Dash	.*-.*
Init Dash	-.*
End Dash	.*-
Punctuation	[,.;:?!-+'"']

Table 1: Orthographic features.

System	Precision	Recall	F-Measure
No Lexicons	0.830	0.773	0.801
Lexicons	0.864	0.787	0.824

Table 2: System performance on eval. data.

development set to the training data and evaluated on 5000 new unannotated sentences. The results are shown in table 2. Entities were correctly identified by the system if and only if all and only the tokens of the entity were correctly detected.

Adding the ABGene lexicons made a significant improvement to both precision and recall. This is a very good indicator that additional domain knowledge may help to further improve the accuracy of the system. To determine which lexicons gave the best performance, we conducted experiments examining the effect of adding each type of lexicon individually to the model and tested the model on the development data. These results are outlined in table 3. Each list made a small improvement to the overall accuracy of the system, with the gene lexicon contributing the largest improvement. Table 3 also shows the performance of the system without lexicons and feature induction.

Overall, our experiments show that CRF models with carefully designed features can identify gene and protein mentions with fairly high accuracy even without features containing domain specific knowledge. However, such features, which in our case take the form of lexicon membership, can lead to improved system performance.

Acknowledgments

The authors would like to thank our collaborators Mark Liberman, Andy Schein, Pete White and Scott Winters for useful discussions and suggestions. We would also like to thank Lorraine Tanabe for making the ABGene lexicons available to us. Finally we are particularly appreciative of Andrew McCallum for providing us with an early version of MALLET.

System	Precision	Recall	F-Measure
No Lex, No Feat. Ind.	0.793	0.731	0.761
No Lexicons	0.807	0.744	0.774
Trigrams	0.811	0.759	0.784
Non-gene Lexicons	0.818	0.743	0.778
Gene Lexicons	0.812	0.775	0.793
All Lexicons	0.817	0.782	0.799

Table 3: Effects of feature induction and lexicons on system performance for devel. data.

References

[1] A critical assessment of text mining methods in molecular biology workshop, 2004. http://www.pdg.cnb.uam.es/BioLINK/workshop_BioCreative_04/

[2] S. F. Chen and R. Rosenfeld. A Gaussian prior for smoothing maximum entropy models. Technical Report CMU-CS-99- 108, Carnegie Mellon University, 1999.

[3] J. Kazama, T. Makino, Y. Ohta and J. Tsujii. Tuning support vector machines for biomedical named entity recognition. In *the Proceedings of the Natural Language Processing in the Biomedical Domain*, ACL, 2002.

[4] S. Kulick, A. Bies, M. Liberman, M. Mandel, R. McDonald, M. Palmer, E. Pancoast, A. Schein, L. Ungar, P. White and S. Winters. Integrated annotation for biomedical information extraction. To appear at *Biolink 2004*, 2004.

[5] John Lafferty, Andrew McCallum, and Fernando Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of ICML-01*, 2001.

[6] R. Malouf. A comparison of algorithms for maximum entropy parameter estimation. In Proceedings of the *Sixth Conference on Natural Language Learning*, 2002.

[7] Andrew McCallum. Efficiently inducing features of conditional random fields. *Conference on Uncertainty in Artificial Intelligence*, 2003.

[8] Andrew McCallum. "MALLET: A Machine Learning for Language Toolkit." http://mallet.cs.umass.edu 2002.

[9] M. Narayanaswamy, K. E. Ravikumar, K. Vijay-Shanker. A biological named entity recognizer. *Pacific Symposium on Biocomputing*, 2003.

[10] T. Ohta, Y. Tateisi, J. Kim, S. Lee and J. Tsujii. GENIA corpus: A semantically annotated corpus in molecular biology domain. In *the Proceedings of the ninth International Conference on Intelligent Systems for Molecular Biology*, 2001.

[11] F. Sha and F. Pereira. Shallow parsing with conditional random fields. In *Proceedings of HLT-NAACL 2003*, pages 213–220. Association for Computational Linguistics, 2003.

[12] L. Tanabe, W. J. Wilbur. Tagging gene and protein names in biomedical text. *Bioinformatics* 18(8), 2002.