An Investigation of Various Information Sources for Classifying Biological Names

Manabu Torii, Sachin Kamboj and K. Vijay-Shanker
Department of Computer and Information Sciences
University of Delaware
Newark, DE 19716
{torii,kamboj,vijay}@mail.eecis.udel.edu

Abstract

The classification task is an integral part of named entity extraction. This task has not received much attention in the biomedical setting, partly due to the fact that protein name recognition has been the focus of the majority of the work in this field. We study this problem and focus on different sources of information that can be utilized for the classification task and investigate the extent of their contributions for classification in this domain. However, while developing a specific algorithm for the classification of the names is not our main focus, we make use of some simple techniques to investigate different sources of information and verify our intuitions about their usefulness.

1 Introduction

In this paper, we investigate the extent to which different sources of information contribute towards the task of classifying the type of biological entity a phrase might refer to. The classification task is an integral part of named entity extraction. For this reason, name classification has been studied in solving the named entity extraction task in the NLP and information extraction communities (see, for example, (Collins and Singer, 1999; Cucerzan and Yarowsky, 1999) and various approaches reported in the MUC conferences (MUC-6, 1995)). However, many of these approaches do not distinguish the detection of the names (i.e., identifying a sequence of characters and words in text as a name) from that of its classification as separate phases. Yet, we believe that we will gain from examining the two as separate tasks as the classification task, the focus of this work, is sufficiently distinct from the name identification task. More importantly, from the perspective of the current work, we hope to show that the sources of information that help in solving the two tasks are quite distinct.

Similar to the approaches of name classification of (Collins and Singer, 1999; Cucerzan and Yarowsky, 1999), we investigate both name internal and external clues. However, we believe that the situation in the specialized domain of biomedicine is sufficiently distinct, that the clues for this domain need further investigation and that the classification task has not received the similar attention deserved. A large number of name extraction methods proposed in this specialized domain have focused on extracting protein names only (Fukuda et al., 1998; Franzen et al., 2002; Tanabe et al., 2002). Since only one class is recognized, the only task these methods directly address is that of identifying a string of characters and/or words that constitute a protein name. These methods do not, at least in an explicit manner, have to consider the classification task.

There are some important reasons to consider the detection of names of other types of entities of biological relevance. Information extraction need not be limited to protein-protein interactions, and extracting names of other types of entities will be required for other types of interactions. Secondly, classification of names can help improve the precision of the methods. For example, KEX (Fukuda et al., 1998) is a protein name recognizer and hence labels each name it detects as a protein. However, names of different types of entities share similar surface characteristics (including use of digits, special characters, and capitalizations). Due to this reason, KEX and other protein name recognizers can pick names of entities other than proteins (and label them as proteins). (Narayanaswamy et al., 2003) reports that by recognizing that some of these names as not those of proteins allows their method to improve the precision of protein name detection. Thirdly detecting names of different classes will help in coreference resolution, the importance of which is well recognized in the IE domain. In such specialized domains, the sortal/class information will play an important role for this task. In fact, the coreference resolution method described in (Castaño et al., 2002) seeks to use such information by using the UMLS system[1] and by applying type coercion. Finally, many information extraction methods are based on identifying or inducing patterns by which information (of the kind being extracted) is expressed in natural language text. If we can tag the text with occurrences of various types of names (or phrases that refer to biological entities) then better generalizations of patterns can be induced.

There are at least two efforts (Narayanaswamy et al., 2003; Kazama et al., 2002) that consider the recognition of names of different classes of biomedical relevance. Work reported in (Pustejovsky et al., 2002; Castaño et al., 2002) also seeks to classify or find the sortal information of phrases that refer to biological entities. However, classification was not the primary focus of these papers and hence the details and accuracy of the classification methods are not described in much detail. Other related works include those of (Hatzivassiloglou et al., 2001; Liu et al., 2001) that use external or contextual clues to disambiguate ambiguous expressions. While these works maybe viewed as similar to word sense disambiguation (WSD), the one reported in (Hatzivassiloglou et al., 2001) in particular is close to classification as well. In this work, using context of individual occurrences, names are disambiguated between gene, protein and RNA senses.

[1]The Unified Medical Language System (UMLS) was developed at National Library of Medicine, a National Institutes of Health at Bethesda, USA.

While our interest is in classification of phrases that refer to entities of biomedical significance, in this work we limit ourselves to name classification. In our investigations, we wish to use an annotated corpus for both inducing and evaluating features. We are unaware of any large corpus where phrases are annotated with their classes. However, large corpora for named entity extraction in this domain are being developed, and fortunately, corpora such as GENIA being developed at University of Tokyo are freely available. We make use of this corpus and hence investigate the classification of names only. However, we believe that the conclusions we draw in this regard will apply equally to classification of phrases other than names as well.

2 Sources of Information for Name Classification

To classify a name we consider both the words within the name (i.e., name internal) as well as the nearby words, the context of occurrences.

2.1 Using Name Internal Information

Methods for learning to identify names try to induce patterns of words and special characters that might constitute names. Hence the entire sequence of words in a name is important and necessary for name identification purposes. In contrast, for classification purposes, some parts of the names are more important than the others and some may play no role at all. For example, in the name *cyclic AMP response element-binding protein*, the last word, *protein*, is sufficient for its classification. Similarly, *Adherence-isolated monocytes*, can be classified on the basis of its last word, *monocytes*.

The fact that the last word of a name often bears the most information about the class of the name is not surprising. In English, often the type of object referred by a noun phrase is given by the head noun. Viewing a name as a noun phrase, the head noun is likely to determine its class. And in English noun phrases, the head noun is often the rightmost word because of the right-branching structure of English noun phrases. Quite often the nouns correspond to concepts (or classes) in an ontology. In such cases, we call these nouns *functional terms* or *f-terms*, following the terminology used in some name recognizers proposed for the biomedical domain.

2.1.1 F-terms

The notion of *f-terms*, was first introduced in the design of KEX (Fukuda et al., 1998). In this work, a set of words such as *proteins* and *receptors*, were manually selected as f-terms. In this protein name recognition system, as well as in Yapex (Franzen et al., 2002), f-terms are only used for locating names in text. On the other hand, the system reported in (Narayanaswamy et al., 2003), which identifies the names of other classes as well, generalizes them to also classify names as well. Thus, f-terms are identified with types/classes.

The existing methods that use f-terms rely on a manually selected list of f-terms. However, manual selection methods are usually susceptible to errors of omission. In Section 4.1, we consider a method that tries to automatically select a list of f-terms and the resultant word classes based on the GENIA corpus. We then use this generated list to test our intuitions about f-terms.

We also consider f-terms extended to consist of two consecutive words. We refer to these as *bigram* f-terms to differentiate them from single word only (*unigram*) f-terms. The use of bigrams will help us to classify names when the last word is not an f-term, but the last two words together can uniquely classify the name. For example, *Allergen -specific T cell clones* cannot be classified using the last word alone. However, a name ending with *cell clones* as the last bigram is likely to be a 'Source'.

2.1.2 Suffixes

Often the information about the class designated by a noun can be found in its suffix, particularly in a scientific domain. If f-terms can be viewed as words that designate a class of entities then note that suffixes also play the same role. For example, words ending with the suffix *-amine* are nitrogen compounds and those ending with *-cytes* are cells. Thus using suffixes results in a generalization at the word level. A method of selecting a list of suffixes and associating classes with them is described in Section 4.1.

2.1.3 Example-based Classification

Of course, not all names can be classified on the basis of f-terms and suffixes only. Sometimes names are chosen on a more ad hoc manner and do not reflect any underlying meaning. In such cases, matching with names found in a dictionary would be the only name-internal method possible.

We cannot simply use an "exact matching" algorithm since such a method would only work if the name was already present in our dictionary. As it is not reasonable at this time to have a dictionary that contains all possible names, we can attempt to use approximate matches to find similar names in the dictionary and use them for classification purposes. Such a method then can be thought of finding a way to generalize from the names in a dictionary, instead of relying on simple memorization.

However, assuming a large dictionary is not feasible at this time especially for all the classes. So our alternate is to look at examples from GENIA corpus. The candidate examples that we will use for classification would be the ones that most closely match a given name that needs to be classified. Hence, the method we are following here essentially becomes an example-based classification method such as k-nearest neighbor method. One approach to this task is described in Section 4.3.

2.2 Using Context

We now turn our attention to looking at clues that are outside the name being classified. Using context has been widely used for WSD and has also been applied to name classification (for example, in (Collins and Singer, 1999; Cucerzan and Yarowsky, 1999)). This approach has also been adopted for the biomedical domain as illustrated in the work of (Hatzivassiloglou et al., 2001; Narayanaswamy et al., 2003; Castaño et al., 2002)[2].

In the WSD work involving the use of context, we can find two approaches: one that uses few *strong* contextual evidences for disambiguation purposes, as exemplified by (Yarowsky, 1995); and the other that uses weaker evidences but considers a combination of a number of them, as exemplified by (Gale et al., 1992). We explore both the methods. In Section 4.4, we discuss our formulation and present a simple way of extracting contextual clues.

[2](Castaño et al., 2002) can be seen as using context in its type coercion rules.

3 Experimental Setup

3.1 Division of the corpus

We divided the name-annotated GENIA corpus (consisting of 2000 abstracts) into two parts—1500 abstracts were used to derive all the clues: f-terms, suffixes, examples (for matching) and finally contextual features. These derived sources of information were then used to classify the names found in the remaining 500 abstracts. The keys from the annotated corpus were then used to compute the precision and recall figures. We will call these two parts the training and test sections.

Since we pick the names from the test section and classify them, we are entirely avoiding the name identification task. Of course, this means that we do not account for errors in classification that might result from errors in identifying names. However, we believe that this is appropriate for two reasons. Our investigation focuses on how useful the above mentioned features are for classification and we felt that this might be slanted based on the name identifier we use and its characteristics. Secondly, most of the errors are due to not finding the correct extent of the name, either because additional neighboring words are included or because some words/characters are not included. In our experience, most of these errors happen at the beginning part of the name and, hence, should not unduly affect the classification.

3.2 Classes of Names

In our method, we classify names into one of the five classes that we call Protein, Protein Part, Chemical, Source and Others. We don't have any particularly strong reasons for this set of classes although we wish to point out that the first four in this choice corresponds to the classes used by the name recognizer of (Narayanaswamy et al., 2003). It must be noted that the class proteins not only include proteins but also protein families, and genes; all of which are recognized by many protein name recognizers. The GENIA class names were then mapped onto our class names.

3.3 Tokenization

After the assignment of classes, all the extracted names were tokenized. Noting that changing a digit by another, a Greek character by another, a Roman numeral by another rarely ever results in obtaining another name of a different class, our name tokenization marks these occurrences accordingly. To remove variability in naming, hyphens and extra spaces were removed. Also, as acronyms are not useful for detecting types, their presence is identified (in our case we use a simplistic heuristic that acronyms are words with 2 or more consecutive upper case characters).

3.4 Evaluation Methodology

We used an n-fold cross-validation to verify that the results and conclusions we draw are not slanted by a particular division of the 2000 abstracts. The corpus was divided into sets of 500 abstracts - the composition of each set being random - thus obtaining 4 different partitions. In the first partition, the first three sets were combined to form the Training Set and the last was used as the Test Set. In the second partition, the second, third and fourth sets formed the Training Set and the first was used as the Test Set and so on.

The overall results that we report in Section 5 were the average of results on the four partitions. However, the first partition was used for more detailed investigation.

4 Classification Method

Given an unclassified name, we first tried to classify it on the basis of the f-terms and the suffixes. If that failed, we applied our string matcher to try to find a match and assign a category to the unknown name. Finally, we used context to assign classes to the names that were still left unclassified.

4.1 F-Term and Suffix Extraction

Since we consider f-terms to be nouns that appear at the end of a name and denote a type of entity, their presence in the name suffices for its classification. Hence, we use the last words of names found in the training set to see if they can uniquely identify the class. To generate a list of f-terms and their respective classes, we count each word or pair of words that is found at the end of any name. A unigram or bigram, w, was selected as an f-term if it appeared at least 5 times and if the conditional probability P(class$|$ w) for any class exceeds a threshold which we set at 0.95.

In the counting to estimate this conditional probability we ignore the presence of digits, Greek characters and Roman numerals as discussed in the Section 3.3. For example, in *latent membrane protein 1* the '1' at the end is ignored and '*protein*' will be selected as the unigram for the count.

The number of f-terms selected for chemicals was the lowest. This is not surprising considering chemical names have few words defining subtypes of chemicals. *acetate* was an example chosen for this class. Some other examples of extracted f-terms and their associated classes are: *cell, tissue, virus* (for Source); *kinase, plasmid* and *protein* (for Proteins); *subunit, site* and *chain* (for Protein Parts) and *bindings and defects* (for Others). A couple of surprising words were selected. Due to the limitations of our method, we do not check if a last name indeed denotes a class of entities but merely note that the name is strongly associated with a class. Hence, protein names like *Ras* and *Tax* were also selected.

For suffix extraction, we considered suffixes of length three, four and five. Since we argued earlier that the suffixes that we are considering play the same role as f-terms, we only consider the suffixes of the last word. This prevents the classification of *cortisol- dependent BA patients* (a 'Source') as a 'Chemical' on the basis of the suffix *-isol*. Also, like in the case of f-terms, digits, Greek characters etc at the end of a name were ignored. However, unlike f-terms, if the last word is an acronym the whole name is dropped, as taking the suffix of an acronym wouldn't result in any generalization. The probability of a class given a suffix is then calculated and only those suffixes which had a probability of greater than the probability threshold were selected.

When generating the list of suffixes, we have two possibilities. We could choose to consider names which ended with an f-term that was selected or not consider these names under the assumption that f-terms would be sufficient to classify such names. We found that considering the suffixes of the f-terms results in a significant increase in the recall with little or no change in precision. This rather surprising result can be understood if we consider the kinds of names that show up in the class *Others*. A suffix such as *ation* was selected because a number of names ending with selected f-terms like *transplantation, transformation*, and *association*. This suffix allows us to classify *AP-1 translocation* on the basis of the suffix despite the fact that *translocation* was not chosen as an f-term.

4.2 Classification based on f-terms and suffixes

Given a set of f-terms and suffixes, along with their associated classes, selected from the training part, names in the test portion were classified by looking at the words that end the names. If a name ended with a selected f-term, then the name was tagged as belonging to the corresponding class. If a match was not found, the suffix of the last word of the name was extracted and a match was attempted with the known list of suffixes. If no match was found, the name was left unclassified.

4.3 Classifying Names using Similar Examples

We had discussed earlier the use of similar examples to classify a new occurrence of a name. To find similar examples, standard string matching algorithms are often used which produce a similarity score that varies inversely with the number of edit operations needed to match two strings identically. However, we abandoned the use of standard string matching programs as their performance for classification purposes was rather poor. Primarily this was due to the fact that these algorithms do not distinguish between matches at the beginning and at the end of the name strings. As discussed before, for classification purposes the position of words is important and we noticed that matches at the beginning of the strings were hardly ever relevant unlike the case with those at the end. For this reason, we developed our own matching algorithm.

Given a name in the test corpus, we try to find how similar it is to candidate examples taken from the training portion. For each pair of names, we first try to pair together the individual words that make up the names allowing for some partial matching. These partial matches allow for certain kinds of substitutions that we do not believe will affect the classification. These include dropping a plural "s", substituting one Greek character by another, changing an uppercase character by the same character in lower case, changing an Arabic/Roman single digit by another, changing a Roman numeral by an Arabic one, and dropping digits. Each substitution draws a small penalty (although dropping digits incurs a slightly greater penalty) and only a perfect match receives a score of 1 for matching of individual words. Complete mismatches receive a score of 0.

We then try to assign a score to the whole pair of names. We begin by assigning position numbers to each pair of words (including matches, mismatches and drops) starting from the rightmost match which is assigned a position of zero. Mismatches to the right of the first match, if any, are assigned negative positions. We then use a weight table that gives more weightage to lower position numbers (i.e., towards the end of the strings rather than the beginning) to assign a weight to each pair of words depending on the position. Then the score of the entire match is given by a weighted sum of the match scores, normalized for length of the string. Assigning a score of 0 for a mismatch is tantamount to saying that a mismatch does not contribute towards the similarity score. A negative score for a mismatch would result in assigning a penalty.

We only consider those strings as candidate examples if their similarity score is greater than a threshold α. To assign a class to a name instance, we look at the **k** nearest neighbors, as determined by their similarity scores to the name being classified. To assign a class to the name, we weight the voting of each of the k (or fewer) candidates by their similarity score. A class is assigned only if the the ratio of the scores of the top two candidates exceeds a threshold, β. The precision should tend to increase with this ratio. [3]

4.4 Classifying Based on Context

To identify the best sources of contextual information for classifying names, we considered two possibilities — the use of a single strong piece of evidence and the use of a combination of weak evidences. For the former we made use Decision Lists similar to Yarowsky's method for Word Sense Disambiguation (WSD) (Yarowsky, 1995). However, we found that this method had a poor recall.[4]

Hence, we decided to use a combination of weak evidences and employ the Naive-Bayes assumption of independence between evidences, similar to the method described in (Gale et al., 1992). To do this, the words that occurred within a window and that matched some template pattern were selected as features if their scores [5] exceeded some threshold (which we name **a**). Also, unlike Decision Lists, all the features presented in the context of a name instance were involved in its classification and the probability that a name instance has a certain class was calculated by multiplying probabilities associated with all the features. As some of the evidences might be fairly weak, we wanted to classify only those cases where the combination of features strongly indicated a particular class. This is done by comparing the two probabilities associated with the best two classes for an instance. A class was assigned to a particular name instance only when the ratio of the two probabilities was beyond a certain threshold (which will call **b**). Together with the threshold, **a** for the feature selection, choice of this threshold could allow trade-off between precision and recall for classification accuracies.

5 Results and Evaluation

5.1 F-Terms and Suffixes

Table 1 gives the precision and recall values for the first partition for both f-terms and suffixes.[6] As can be seen, the recall for 'Chemical' is very low as compared to the other classes. This is due to two reasons—firstly most chemical names consist of only one word and secondly we found that chemical names do not end with an indicative word.

The number of f-terms and suffixes extracted by our program was considerably less for Chemicals and Protein Parts as compared to Proteins and Others. While this is consistent with the the explanation of poor recall for chemicals, it can be noted that the low number of f-terms and suffixes extracted for protein parts did not affect its recall in the same manner. As expected the precision remains high for all classes.

[3]As always, the reason for using a threshold is that it allows us to find the appropriate level of compromise between precision and recall. Given that there are different sources of information, there is no need to insist that particular method assign a class tag if we are not comfortable with the level of confidence that we have in such an assignment.

[4]Due to space limitations, we don't discuss why we might have obtained the poor recall that we got for the decision list method.

[5]The scores were simply the conditional probability of a class given a word

[6]The suffix list includes f-terms.

Class	F-Term and suffix				String Matching				Context			
	F-Term		Suffix		Alone		After Suffix		a = 5, b = 2		a = 2, b = 5	
	Prec.	Rec.	Prec.	Rec.	Prec.	Rec.	Prec.	Rec.	Prec.	Rec.	Prec.	Rec.
Chemical	0.97	0.05	0.98	0.19	0.89	0.54	0.90	0.59	0.85	0.06	0.55	0.10
Protein	0.97	0.35	0.98	0.55	0.92	0.81	0.93	0.81	0.70	0.31	0.53	0.76
Protein Part	0.98	0.40	0.98	0.33	0.86	0.75	0.85	0.76	0.75	0.05	0.37	0.12
Source	0.98	0.61	0.97	0.62	0.95	0.87	0.94	0.89	0.83	0.10	0.78	0.10
Others	0.99	0.69	0.97	0.71	0.96	0.87	0.96	0.91	0.80	0.05	0.74	0.03
Total	0.98	0.49	0.98	0.57	0.93	0.81	0.93	0.84	0.72	0.17	0.53	0.36

Table 1: **Results for the various stages of our method.**

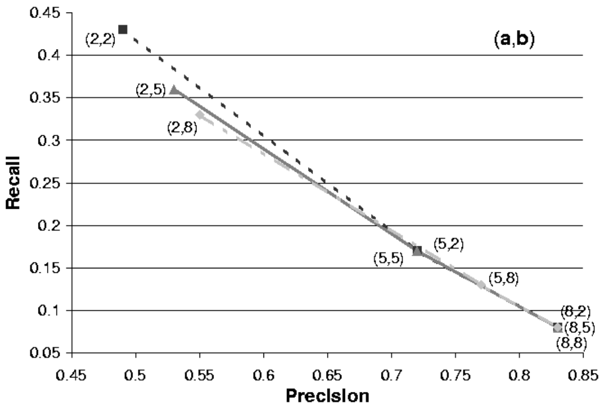

Figure 1: **Precision-Recall Tradeoff**

5.2 Using Examples

For the string matching, we tried three different set of values for the parameters α, β and k,that is (0.3, 2, 3), (0.7, 2, 1) and (0.7, 2, 5). We found that the results were marginally better for the set (0.3,2,3) and, hence, show the results for this set only. Table 1 shows the results of applying the string matching to the first partition — all by itself and on names not classified after the suffix stage. As can be seen, the recall is higher than the previous stages but with a slight reduction in precision.

5.3 Results for Context

We ran the context classifier for different values of the parameters **f**, **a** and **b** but finally chose a value of 5 for **f** because choosing a higher frequency threshold does not improve the precision but hurts the recall. Figure 1 shows the precision plotted against the recall for different choice of **a** and **b**.

The values of the precision and recall on the first partition for each individual class and the two sets of thresholds are shown in Table 1. The first set, that considers stronger evidences (since a is higher), achieves higher precision but recall is not satisfactory. Most of the word evidences chosen tended to indicate a classification of proteins and hence the higher recall for this class. Allowing weaker evidences (because $a = 2$) means more contextual evidences were selected and hence a higher recall was obtained (particularly for protein). But precision is lowered except for Source and Others (which incidentally don't show an increase in recall).

Class	Precision	Recall
Chemical	0.87	0.62
Protein	0.84	0.90
Protein Part	0.86	0.79
Source	0.94	0.87
Others	0.96	0.90
Total	0.90	0.87

Table 2: **Overall Results**

5.4 Overall Results

Table 2 shows the precision and recall for all the different classes, averaging it out for the 4 different partitions. We observed very little variance between the results for the different partitions.

6 Conclusions and Future Work

We have considered a few name internal and external sources of information that help in the classification of names. Despite using fairly simple methods to classify the names, we have obtained encouraging results which we take to suggest that that our intuitions about them are on the right track. We feel that the effectiveness of f-terms and suffixes that generalize the idea of f-terms, the matching algorithm that places more emphasis on partial matches of words to the right vindicates our stance that the classification of names is a task sufficiently distinct from the name identification process and warrants an independent investigation. Even the use of context is different for the two tasks as in the latter task only the immediately neighboring words are used and that too for purpose of demarking the extremities of the name string.

While the high precision of f-terms and suffix based classification was expected, the recall of these methods was higher than expected. It is also clear that these methods do not help much with the chemicals class. We believe that in addition to suffix, the knowledge of other chemical root forms (such as "meth"), e.g., used in (Narayanaswamy et al., 2003), would be useful.

We would like to focus more on the matching part of the work. In particular, rather than hand-coding our intuitions in terms of weights for the different parameters, we would like to automatically, e.g., using a held-out validation set, have these set and see to what extent the automated choice of parameters show the bias for the rightmost words in the matching. We would also like to generalize our work further by not limiting the classes to the ones chosen here but allow a wider range of classes. To do this, we would like to consider the GENIA classes and collapse classes at various levels of their ontology and try to see at what level of fine-grained distinctions can classification still be done satisfactorily. In regards to the use of the contextual method, while we have some preliminary ideas, we need to investigate further why the use of a single strong clue, as exemplified by the decision list method, does not work as well as it seems to for the WSD task.

References

J. Castaño, M. Zhang, and J. Pustejovsky. 2002. Anaphora Resolution in Biomedical Literature. In *Proc. of International Symposium on Reference Resolution*.

M. Collins and Y. Singer. 1999. Unsupervised Models for Named Entity Classification. In *Proc. of EMNLP 1999*.

S. Cucerzan and D. Yarowsky 1999. Language Independent Named Entity Recognition Combining Morphological and Contextual Evidence. In *Proc. of Joint SIGDAT Conference on Empirical Methods in NLP and Very Large Corpora*, 90–99.

W. Gale, K. W. Church, and D. Yarowsky. 1992. A method for disambiguating word senses in a large corpus. *Computers and the Humanities*, 26:415–439.

K. Fukuda, T. Tsunoda, A. Tamura, and T. Takagi. 1998. Toward information extraction: identifying protein names from biological papers. *Proc. of ISMB 1998*, 707–18.

K. Franzén, G. Eriksson, F. Olsson, L. Asker, P. Lidén, and J. Cöster. 2002. Protein names and how to find them. *International Journal of Medical Informatics special issue on Natural Language Processing in Biomedical Applications*, 67:49–61.

V. Hatzivassiloglou, P. A. Duboue, and A. Rzhetsky. 2001. Disambiguating proteins, genes, and RNA in text: a machine learning approach. *Bioinformatics*, 17 Suppl 1: S97–S106.

J. Kazama, T. Makino, Y. Ota, and J. Tsujii. 2002. Tuning Support Vector Machines for Biomedical Named Entity Recognition. In *Proc. of the ACL-02 Workshop on Natural Language Processing in the Biomedical Domain*, 1–8.

H. Liu, Y. Lussier, and C. Friedman. 2001. Disambigutting Biomedical Terms in Biomedical Narrative Text: an Unsupervised. *Journal of Biomedical Informatics*, 34 (4): 249-61.

Proc. of the Sixth Message Understanding Conference (MUC-6). 1995. Morgan Kaufmann.

M. Narayanaswamy, K. E. Ravikumar, and K. Vijay-Shanker. 2003. A Biological Named Entity Recognizer. In *Proc. of PSB 2003*. 8.

J. Pustejovsky, J. Castaño, J. Zhang, M. Kotecki, and B. Cochran. 2002. Robust Relational Parsing Over Biomedical Literature: Extracting Inhibit Relations. In *Proc. of PSB 2002*, 7:362–373.

L. Tanabe and W. J. Wilbur 2002. Tagging gene and protein names in full text articles. In *Proc. of the ACL-02 Workshop on Natural Language Processing in the Biomedical Domain*, 9–13.

D. Yarowsky. 1995. Unsupervised Word Sense Disambiguation Rivaling Supervised Methods. In *Proc. of ACL 1995*, 189–196.