Gene name recognition with systematic feature assessment

R.Lakshmi [1], T.Lavanya [2],
Assistant Professor [1],[2],
Department of CSE, SRK INSTITUTE OF TECHNOLOGY ENIKEPADU VIJAYAWADA
Mail Id : Vishnu laxmi5134@gmail.com, Mail id : lavanyathomandra@gmail.com,

Abstract

Task 1A of the Bio Creative assessment required the development of systems that can extract gene and protein names from free-text utterances. To solve this issue, we use a pattern-based post-processing to recognize phrases and a word classification system based on a sliding window method with a Support Vector Machine. Whether or whether the categorization approach takes into account factors like prefixes, suffixes, character n-grams, capitalization patterns, and word order is vital to the system's effectiveness. We provide a method based on recursive feature elimination, RFE, for comparing the efficacy of various feature sets. We can measure the effect of various feature sets on the outcomes of the word classification problem by gradually decreasing the number of features utilized by the system. We may then use this information to develop solutions that are quicker and simpler to comprehend by identifying descriptive characteristics and gaining insight into the underlying structure of the issue. We find that the SVM is resistant to the use of duplicated features. When compared to utilizing all of the features, RFE yields a 0.7% performance boost. In addition, employing less than 5% of the features may provide performance that is just 2.3% worse than this limit.

Introduction

Named entity recognition (NER) problems with gene and protein names [2] are at the heart of Task 1A of the Bio Creative assessment [1]. Participants were given a corpus of 10,000 words that included protein and gene names. We'll abbreviate "gene and protein" to "gene" for brevity, although these lines required the development of algorithms to identify gene and protein names in random text. The algorithms were evaluated using a dataset of 5,000 unread and untagged texts. The assessment was stringent, meaning that multi-name (phrase) gene names like "Drosophila shc gene product" had to be identified in their entirety, without any missing or extraneous portions. All of the organizer-provided statements have already been tokenized, or parsed into individual words and phrases. The training corpus also included embedded part-of-speech tags for easier comprehension. There are several publications available on the difficulties inherent in the NER process for gene names in biomedical literature (for examples, see [3,4]). Multiple names for genes, abbreviations galore, names that include numeric and special characters, and ambiguity over where a sentence ends all result from a lack of agreed-upon nomenclature standards. The recognition of multi-word phrases has shown to be a particularly challenging challenge if rigorous assessment criteria are used.

Methods

Our system consists of a two-stage process. To determine whether or not a given token is part of a gene name, a Support Vector Machine classifier [15-17] is taught to make that determination. In addition, our system activates a series of post-processing rules that are sensitive to context and classify specific neighbouring tokens as genes in order to identify compound nouns [18]. Examples are represented as vectors in the SVM. Each dimension in the vector space represents a feature of an example, and the values of those parameters are unique to each instance. Each token in both the training and testing data serves as an example in our situation. Each instance is either marked as "positive" (a component of a gene name) or "negative" (not part of a gene name). Support Vector Machines (SVMs) are capable of learning a hyperplane with a discriminating margin between two classes. Which side of the hyperplane a new example (from the test corpus) falls on then determines whether the example is positively or negatively labelled.

Connected Tasks

Our methods are used by several of the other groups in the 2003 Bio Creative assessment. Three further teams advocate for SVM-based token classifiers. The effectiveness of such methods varies considerably. We'll show you the systems that are most like ours and compare and contrast how they handle things.

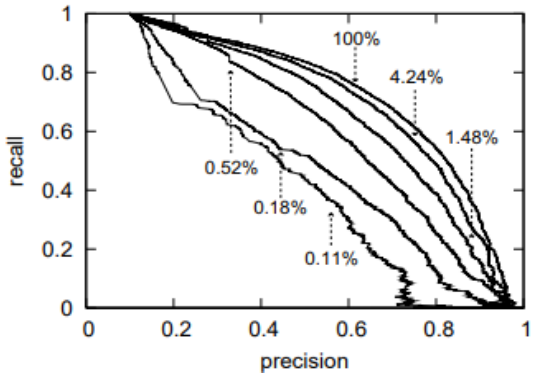

Figure 2Impact of the Recursive Figure 2 e Feature Elimination Impact of the Recursive Feature Elimination. Impact of removing 10% of the features with the lowest weight vctor in each round. After 30 iterations, with only 4.28% of all features remaining, the f-measure has dropped only by 2%. The underlying evaluation method only considers the recognation of single tokens rather than whole phrases. The bottom line (65 iterations) shows the impact of the remaining 0.11% of all features. All values are evaluated without the post-expansion step (see text). Dependence of the f-measure on the number of features

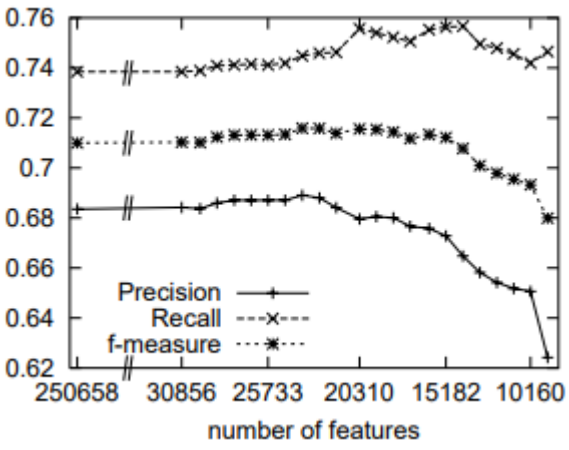

Figure 3 The f-measure's dependence on the feature count. Precision, recall, and f-measure results for a range of feature counts. As we start with the whole feature set, the features with the lowest weight vector are eliminated using recursive feature elimination, and our performance is evaluated after each round.

PosBIOTM-Ner

Song et al. [9] utilize a variant of the BIO-markup with classes that isolate gene names from adjacent text. Not only do they classify tokens as being either inside or outside of a gene name (I/O), but they also introduce a new category for tokens that come before the first letter of a compound gene name (B). In our own system, we solely make use of I and O. Significantly expanding the feature space using an edit-distance-based gazetteer search for tokens and phrases. The edit-distance algorithm takes into account linguistic shifts. Our method involves using precise matches of candidate phrases against a gazetteer that also includes alternative spellings. Song et al. make an effort to artificially expand the training sample. Sentences from the training corpus are recycled by the system with new gene names inserted. Tokens other than noun-phrases and determiners are disregarded entirely on the assumption that all proper nouns are limited to these constructions. The method's f-measure drops from 73.8% without the extra training examples to 66.7% with them, showing that limiting it to noun-phrases does not boost its performance. Using fabricated data increases recall by 1-2% at the expense of a 17% decrease in accuracy. The primary differences between our system and PosBIOTM-NER are the usage of BIO-classes and the invocation of a gazetteer, and PosBIOTM-NER achieves roughly 1% higher f-measure, mostly due to the improved accuracy of its predictions (80% precision at 68.5% recall).

Yam Cha

The method described above is similar to one used by Matsumura et al. [10], which extends BIO classes. The approach also makes use of a context window;however, this has no positive effect on our system's efficiency. We also tried using various sized context windows (both symmetrical and asymmetrical) to see whether they had any effect on performance, but were ultimately disappointed. However, keep in mind that our post-processing stage does employ context in a subtle way. The Yam Cha gazetteer function makes use of data from both SWISS-PROT and TrEMBL. Up to an f-measure of 78.1%, performance is improved by 3 percentage points. Yam Cha also uses the gazetteer in a different manner than we do, and its entries come from a different source. Token n-grams are used to find terms in the gazetteer that are a precise match. Yam Cha is around 5% more efficient than our approach, despite not include a name expansion phase. This comes as a shock since our system's performance drops by around 15% when not post-processed.

Powerline

Ensemble approaches using two HMM and one SVM classifier are proposed by Zhou et al. [5,6]. The SVM employs a feature set similar to our own. Instead,then using a single dimension to describe a candidate token's occurrence in a vocabulary, Powerline uses a separate dimension for each each

word. Table 3 Orthographic Characteristics The post-expansion phase's guiding principles. Certain POS tags are converted to NEWGENE tags according to the regulations. In the extension of noun phrases, we leave off 372/222 nouns and add just 778 specific adjectives. The notation NN* denotes nouns, proper nouns, and plurals; JJ denotes adjectives; CD denotes cardinal numbers; DT denotes determiners; and '/' indicates that the token is itself a noun.

Former POS pattern	Expanded pattern	Limitation
NEWGENE NN*	NEWGENE NEWGENE	all but 372 particular nouns
NN* NEWGENE	NEWGENE NEWGENE	all but 222 particular nouns
JJ NEWGENE	NEWGENE NEWGENE	only 778 particular adjectives
NEWGENE JJ	NEWGENE NEWGENE	only 778 particular adjectives
NEWGENE DT NN*	NEWGENE NEWGENE NEWGENE	
NEWGENE CD	NEWGENE NEWGENE	
NN* / NEWGENE	NEWGENE NEWGENE NEWGENE	
NEWGENE / NN*	NEWGENE NEWGENE NEWGENE	

similar to ours in terms of token representation, but furthermore include tests for parentheses, punctuation, and stop words. We don't know the details of Powerbaseone's stop word list, but our own version uses the top 10,000 most frequent English terms (out of a total of 100, 000). Zhou et al. differentiate between two types of triggers that indicate whether or not a token is included in two distinct word lists. The first group comprises terms that are often found within gene names, whereas the second group contains words that are frequently found in the local context of gene names. Our own approach calculates a single feature's value based on its distance from these keywords. The authors offer an ensemble technique; hence it is unclear how their system performs in comparison to ours in terms of prediction accuracy. Similarly, the impact of various feature sets varies. Despite what some of the other participants may claim, our prediction ability is not improved by some of the factors they find helpful. We've recreated everything that wasn't in our first system, including ldl, ddd, and DNA (see Table 1), but these additions don't help us much. Because of their poor discriminative strength, practically all of them are eliminated in early rounds by the recursive feature removal.

Feature classes and feature definitions

We construct a collection of features from the token and its context for each token in the training and test corpora. For example, instead of using just the characteristics "al", "lap", and "ph.", we may switch to using all character 2-grams. Feature engineering, therefore, is concerned with determining which kinds of features are most useful (for examples, see Table 1). The features are created using the following categories and their respective definitions: Invisible token Each token in the training set is treated as a separate feature. In cases where they are relevant, the values from tf.idf are used. It is our goal to train a model that can easily include novel tokens. One binary feature ("unseen") is included to accomplish this goal; it is set to 0 if the token is part of the training set and 1 otherwise. Tokens are assumed to be invisible at random throughout training. This implies that we don't properly set the tf.idf value for the single token feature and instead set the "unseen" feature to 1 with a minimal prob ability.

Results

We employed a technique where feature classes were selected via trial and error for the BioCreAtIvE evaluation. The outcomes of this system will be presented initially. Once the competition was over, we used a more methodical approach to zero in on the best possible feature set. Starting with a model that included every possible feature class, we then iteratively eliminated the ones with the lowest weights in the SVM-learned model. In this chapter's second half, the findings are discussed. We conclude by demonstrating the results of our post-processing phase, which was the same for both scenarios. Table 1 lists all characteristics analysed by both methods, along with their effect on a default classifier.

Unique functionality packages

We created many feature classes and tested out various on/off class use combinations for the Bio Creative submission. In Table 1, the feature classes highlighted with a "*" were the ones we submitted. Simply using the tokens as features provides a baseline f-measure of 54.1% for the system. The system reaches, say, 68.2% following the addition of character 1-, 2-, and 3-grams. When all the bells and whistles are turned on, performance suffers. We did this by identifying a subset of surface cues that gave us the highest attainable f-measure. This subset covers checks for tokens consisting of a single or double capital letter, all capital letters, all lowercase letters, all uppercase letters, special characters, and numerical combinations. Not all of these characteristics enhanced performance when selected alone, but only in conjunction with others, maybe because of the connection of feature classes. For instance, the performance benefits of using special characters and letter/digit combinations were seen only when the two were utilized in tandem.

The competition was split into two categories: open, where participants could build gazetteers using any available data, and closed, where participants could only use the training data given by the tournament's organizers. Because of the greater expected influence of gazetteers, this distinction was made. Contrary to expectations, this has not materialized. When used with gazetteers produced from the

training data, automatically constructed gazetteers had a rather little effect. Performance drops slightly when only tokens are used as features compared to when the gazetteer constructed from training data and external sources (synonym lists for gene names from mouse, yeast, and fruit fly (all from Biotreat tIvE Task 1B), and human [12]) are used as features (see Table 1). We saw a little boost when combining gazetteers with other feature classes, therefore we made sure to include them in our submissions for both regions. The aggregate data indicated that the gap between open and closed division was just 1%. When compared to the findings of Task 1B, where the top-performing systems relied heavily on human curated dictionaries, this little improvement in performance is surprising. Part-of-speech tags are another intriguing feature class. Table 1 demonstrates that reducing analysis to tokens and the given part-of-speech tags allows the f-measure to approach zero. In this scenario, we employed a feature for each part-of-speech tag (a total of 12), and a feature for each token in the training corpus. Only the feature representing the token itself and the part-of-speech tag of the token are 1s in this vector. The SVM seems to be entirely bamboozled by this data. However, several systems have shown that POS may boost NER performance when trained on a domain-specific corpus and combined with other features [5,6].

Following the enlargement to full gene names

A set of criteria (see Table 3) was manually developed to convert the SVM's predictions of single tokens into whole gene name phrases. Our rules' structure is described in further depth in the "Methods" section. These guidelines either remove tags from individual tokens or expand a series of tokens into a whole sentence. When the SVM assigns a positive tag to a sequence of tokens longer than one that has been labelled as a gene name, that tag is never deleted. Our system improves by 12% in accuracy and 10% in recall once post-processing is applied compared to its unaltered state. More than 20% of the original incorrect negative predictions have been rectified (a total of around 300 gene names). To further reduce false positives, 120 single-word terms like "protein" are removed (representing 10% of the total). When it comes to the latter, we determined via thorough examination that the post-processing is always accurate. Figure 4 shows the difference in recall and accuracy after the expansion was implemented.

Discussion

Our technique for identifying protein and gene names in written text is composed of the following basic building blocks: To begin, we establish a set of attributes that might be useful for the standalone categorization of words. Second, we use recursive feature elimination to narrow down the feature space and zero in on the features that provide the most performance boost to the underlying machine learning classifier. Finally, single-word classifications are combined into phrase classifications by rule-based post processing. Starting with the effects of the RFE stage, we'll go through each of these key issues in turn. In the section under "Related Work," we will compare our system to others in order to analyze its broad set of characteristics and its effect.

Conclusion

An approach to extracting gene and protein names from text has been demonstrated. Our method utilizes a sliding window strategy in tandem with a Support Vector Machine to categorize words. We use a pattern-based post-processing step after making this forecast of candidate tokens in order to recognize compound names. Tokens, character n-grams, a gazetteer, and orthographic traits were identified as the most useful features and feature classes for differentiating gene names from other text. The latter group includes checks for Greek words and patterns including capitalization, numeric characters, and special symbols. The collection was finished off by distances to the keywords. We demonstrated that the performance may be greatly improved by careful selection and testing of feature classes utilized for the vector space model representation of tokens. It is essential to collect many feature classes and examine each one for potential biases. Prediction quality was affected by feature selection for the NER task at hand. Using recursive feature removal, we found that features and feature classes that negatively impacted performance could be identified. When used to detection, RFE is able to identify a subset of characteristics that is just as effective as the entire set.

References

[1]. BioCreAtIvE Challenge Cup 2003 [http://www.pdg.cnb.uam.es/ BioLINK/BioCreative.eval.html].

[2]. Yeh A, Morgan A, Colosimo M, Hirschman L: BioCreAtIvE task 1A: gene mention finding evaluation. BMC Bioinformatics 2005, 6(Suppl 1):S2.

[3]. de Bruijn B, Martin J: Literature mining in molecular biology. Proc EFMI Workshop on Natural Language Processing in Biomedical Applications, Nicosia, Cyprus 2002:1-5.

[4]. Shatkay H, Feldman R: Mining the Biomedical Literature in the Genomic Era: An Overview. Journal of Computational Biology 2003, 10(6):821-856.

[5]. Zhou G, Shen D, Zhang J, Su J, Soon TH, Tan CL: Recognition of Protein/Gene Names from Text using an Ensemble of Classifiers and Effective Abbreviation Detection. BioCreAtIvE Workshop, Granada, Spain 2004.

[6]. Zhou G, Zhang J, Su J, Shen D, Tan CL: Recognizing names in biomedical texts: a machine learning approach. Bioinformatics 2004, 20(7):1178-1190.

[7]. Kinoshita S, Ogren P, Cohen KB, Hunter L: Entity identification in the molecular biology domain with a stochastic POS tagger: the BioCreative task. BioCreAtIvE Workshop, Granada, Spain 2004.

[8]. McDonald R, Pereira F: Identifying Gene and Protein Mentions in Text Using Conditional Random Fields. BioCreAtIvE Workshop, Granada, Spain 2004.

[9]. Song Y, Yi E, Kim E, Lee GG: POSBIOTM-NER: A Machine Learning Approach. BioCreAtIvE Workshop, Granada, Spain 2004.

[10] . Mitsumori T, Fation S, Murata M, Doi K, Doi H: Gene/protein name recognition using Support Vector Machine after dictionary matching. BioCreAtIvE Workshop, Granada, Spain 2004.

[11]. Guyon I, Weston J, Barnhill S, Vapnik VN: Gene Selection for Cancer Classification using Support Vector Machines. Machine Learning 2002, 46(1–3):389-422.

[12]. Wain H, Lush M, Ducluzeau F, Povey S: Genew: The Human Nomenclature Database. Nuc Acids Res 2002, 30:169. 13. Chang JT, Schütze H, Altman RB: GAPSCORE: finding gene and protein names one word at a time. Bioinformatics 2004, 20(2):216-225.

[14] . Seki K, Mostafa J: A Probabilistic Model for Identifying Protein Names and their Name Boundaries. Proceedings of the Computational Systems Bioinformatics Conference (CSB) 2003.

[15] . Vapnik VN: The Nature of Statistical Learning Theory New York: SpringerVerlag; 1995.

[16]. Cristianini N, Shawe-Taylor J: An Introduction to Support Vector Machines and other kernel-based learning methods Cambridge: Cambridge University Press; 2000.

[17]. Joachims T: Text Categorization with Support Vector Machines: Learning with Many Relevant Features. Proceedings of ECML-98, 10th European Conference on Machine Learning, Springer 1998.

[18]. Bickel S, Brefeld U, Faulstich L, Hakenberg J, Leser U, Plake C, Scheffer T: A Support Vector Classifier for Gene Name Recognition. BioCreAtIvE Workshop, Granada, Spain 2004.

[19]. Brill E: A simple rule-based part of speech tagger. Proceedings of ANLP-92, 3rd Conference on Applied Natural Language Processing, Trento, Italy 1992.

[20]. Marcus MP, Santorini B, Marcinkiewicz MA: Building a large annotated corpus of English: the Penn Treebank. Computational Linguistics 1993, 19:313-330.