Ensemble Based Active Annotation for Biomedical Named Entity Recognition

A Dissertation submitted in the partial fulfilment for the award of

MASTER OF TECHNOLOGY

IN

COMPUTER SCIENCE AND ENGINEERING

by

Ritu Srivastava

Roll no. 2K11/CSE/12

Under the Esteemed Guidance of

Mridula Verma

Department of Computer Engineering

Delhi Technological University

New Delhi

2012-2013

<u>DECLARATION</u>

I hereby declare that the thesis entitled **"Ensemble Based Active Annotation for Biomedical Named Entity Recognition"** which is being submitted to the **Delhi Technological University**, in partial fulfillment of the requirements for the award of degree of **Master of Technology in Computer Science and Engineering** is an authentic work carried out by me. The material contained in this thesis has not been submitted to any university or institution for the award of any degree.

Ritu Srivastava

Department of Computer Engineering

Delhi Technological University,

Delhi.

CERTIFICATE

DELHI TECHNOLOGICAL UNIVERSITY

(Govt. of National Capital Territory of Delhi)

BAWANA ROAD, DELHI-110042

Date: ____________________

This is to certify that the thesis entitled **"Named Entity recognition on bio medical domains"** submitted by **Ritu Srivastava (Roll Number: 2K11/CSE/12),** in partial fulfillment of the requirements for the award of degree of Master of Technology in Computer Science and Engineering, is an authentic work carried out by her under my guidance. The content embodied in this thesis has not been submitted by her earlier to any institution or organization for any degree or diploma to the best of my knowledge and belief.

Project Guide

Dr. Mridula Verma

Assistant Professor

Department of Computer Engineering

Delhi Technological University, Delhi-110042

ACKNOWLEDGEMENT

I take this opportunity to express my deepest gratitude and appreciation to all those who have helped me directly or indirectly towards the successful completion of this thesis.

Foremost, I would like to express my sincere gratitude to my guide **Mridula Verma**, **Assistant Professor, Department of Computer Engineering**, **Delhi Technological University, Delhi** whose benevolent guidance, constant support, encouragement and valuable suggestions throughout the course of my work helped me successfully complete this thesis. Without her continuous support and interest, this thesis would not have been the same as presented here.

Besides my guide, I would like to thank the entire teaching and non-teaching staff in the Department of Computer Science, DTU for all their help during my course of work.

RITU SRIVASTAVA

Table of Contents

List of Figures

List of Tables

ABSTRACT

An important prospect of machine learning for information extraction to deal with the problems of high cost of collecting labelled examples. Active Learning makes more efficient use of the learner’s time by asking them to label only instances that are most useful for the trainer. In random sampling approach, unlabeled data is selected for annotation at random and thus can’t yield desired result. In contrast, active learning selects the useful data from a huge pool of unlabeled data for the classifier. The strategies used often classify the corpus tokens (or, data points) under wrong classes. The classifier is confused between two categories if the token is located near the margin. We propose a novel method for solving this problem and show that it favourably results in the increased performance. Our proposed framework is based on an ensemble approach, where ID3 and C5 algorithms are used as the base classifiers. The proposed approach is applied for solving the problem of named entity recognition (NER) in the Bio-medical domain. Results show that the proposed technique indeed improves the performance of the system.

CHAPTER 1

INTRODUCTION

1.1 INTRODUCTION

There is a difficulty with machine learning for information extraction which is the high cost of collecting labelled examples. Active learning [1] is, these days, is a popular research area due to its many-fold potential benefits. It leads to drastic reductions in the amount of annotation that is necessary for training a highly accurate statistical classifier. It can make more efficient use of the learner's time by asking them to label only instances that are most useful for the trainer. Named Entity Recognition (NER) has important role in many Natural Language Processing (NLP) application areas such as information extraction, information retrieval, machine translation, question answering and automatic summarization etc.

In this thesis we propose a novel active learning technique based on the concept of classifier ensemble, where ID3 [3] and C5 [2] and are used as the underlying base classification methods. The outputs of ID3 and C5 are combined together using a weighted voting approach. The proposed approach is evaluated for the Bio-medical dataset.

We observe that our proposed ensemble based active learning method performs better compared to each of the baselines developed using either of the base classifiers, i.e. ID3 and C5.

Named entity recognition (NER) is a subtask of information extraction (I.E.) that seeks to locate and classify atomic element into text into predefined categories such as the names of persons, organizations, locations, expressions of time, quantities, monetary values, percentages, etc.

Due to the developments in investigation of molecular biology and publication of genes and protein sequence, the quantity of interrelated data is growing exponentially. Therefore how to get related biomedical information from a great deal of literature directly is becoming a matter of great urgency. To recognize named entity from biomedical literature such as protein, gene and DNA becomes a basic task of information extraction in bioinformatics.

Biomedical Named Entity Recognition (NER) can be defined as the task of recognizing and categorizing entity names in biomedical domains. Biomedical named entity is different from general NER. There are many reasons; here we have discussed a few. First one is the new named entity keeps emerging. As a result, it is hard to construct a complete dictionary contains various types of biomedical named entity. Secondly many biomedical named entities are multi-word phrases; some of the name is even being very long, which makes characteristics make it difficult to determine the boundaries of named entities. Third one is the same word or phrase can represent different types of biomedical named entities. As a result, context is getting more important or even critical in NER. Forth reason is that biomedical named entities have many different forms of writing and there are too many abbreviations which make it worse to identity items. As we can see here, biomedical named entity recognition is a challenging task.

1.2 MOTIVATION OF THE WORK

Automatic knowledge discovery and efficient information access is becoming a critical issue in the biomedical literature. Recently, a huge amount of biomedical information has become available in electronic form and the size of the knowledge repository is rapidly increasing. For instance, the MEDLINE database [12] contains over 12 million abstracts and many new abstracts are added monthly. Thus, it is difficult for biomedical researchers to find information of interest from such a vast database that is continuously updated. Therefore there is the necessity of information extraction based on computational text processing.

The biomedical domain, due to its complicated language, has consistently lagged behind. Here the goal is to extract named entities such as Gene, Protein or Disease names. Accurate extraction of these entities is central to the various text mining and knowledge discovery tasks that have now become essential due to the overwhelming amount of textual information being produced.

The Challenges

The biomedical domain contains its own specialized terminology and complex naming conventions, which make the task of identifying named entities a challenging one. Several attempts have been made to port existing open domain NER techniques to the domain However, few of them achieved satisfactory performance. A discussion of these domain specific challenges is provided below:

Open, growing vocabulary: Most new biomedical texts introduce new terms, such as specific notations, acronyms, and innovative names for new concepts. Thus, the rapid increase of biomedical terminology and its diverse usage are the most critical challenges for biomedical IE systems.

Inconsistent naming conventions: The naming conventions are not standardized; an entity can be represented by various spelling forms. For example, "IL-2" has many variants such as "IL2", "Interleukin 2", and "interleukin-2". In addition, some entities have very long names (e.g., *activated B cell lines, 47 kDa sterol regulatory element binding factor*), while some entities are represented by short abbreviations like *EGFR, or EGP receptor* (i.e., epidermal growth factor receptor).

Polysemy of a NE: Sometimes NEs with the same orthographical characteristics can be classified into different semantic classes depending on a given context. For example, "*IL-2*" is a protein in some contexts, but can be a DNA in other contexts.

Descriptive naming convention: Biomedical named entities are often descriptive, and contain premodifiers, which makes it difficult to identify left boundaries of the entity names. For example 'normal thymic epithelial cells' or 'activated B cell lines'. They can also be very long at times, e.g. '47 kDa sterol regulatory element binding factor'. Zhou *et al.* (2004) found that nearly 18.6% of biomedical entity names in the GENIA V3.0 corpus (Ohta *et al.*, 2002) contained at least four words, as shown in Figure 1.

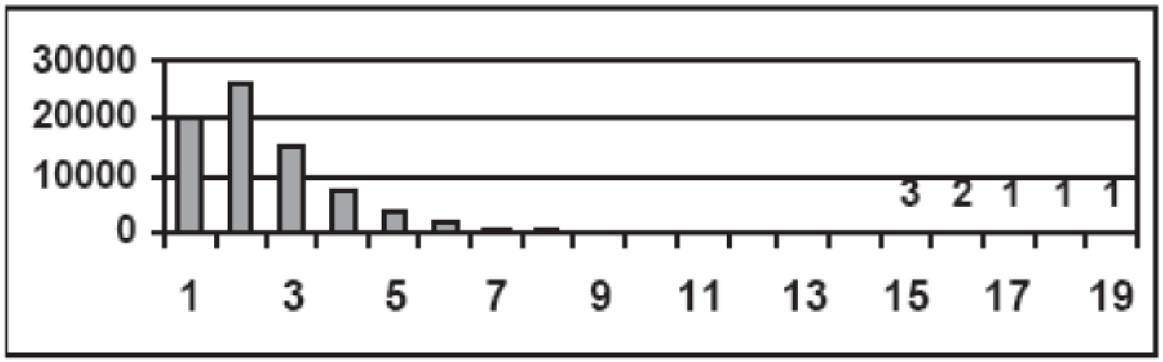


Figure 1 Distribution of the number of words in biomedical entity names(GENIA V3.0) ref. Zhou et. al (2004)

Conjunction and disjunction: Two or more biomedical entity names may share one head noun by using conjunction or disjunction. For example, '91 and 84 kDa proteins' consists of two entity names: '91 kDa proteins' and '84 kDa proteins'. In GENIA V3.0, there are about 2.06% of biomedical entity names in this form Zhou *et al.* (2004).

Non-standardized naming convention: The biomedical language doesn't follow strict naming conventions. The use of capitalization or hyphen is more or less casual and the same entity may often be referred to in different spelling forms e.g. 'Nacetylcysteine', 'N-acetyl-cysteine', 'NAcetylCysteine', etc. Moreover as more and more new entity names are created by authors, the coverage of biomedical dictionaries is partial at best (Fukuda *et al*. 1998; Nobata *et al*. 1999), and performance of extraction techniques solely based on them is low. Zhou *et al*. (2004) showed that in GENIA V3.0, around 62.89% of words in biomedical entity names were lowercase.

Abbreviation: Biomedical domain texts frequently use abbreviations, many of which are non-standard. For example, ‘IL2’ for ‘Interleukin 2’ or ‘PAL’ for ‘palate’. These are often highly ambiguous and its not possible to classify them simply based on existing dictionaries. For example, ‘TCF’ may refer to ‘T cell Factor’ or ‘Tissue Culture Fluid’ in different articles. Chang *et al*. (2002) have shown that, in MEDLINE abstracts until the end of 2001, 42.8% of abstracts have at least one abbreviation and 23.7% of abstracts have two or more. They also showed that there is one new abbreviation in every 5–10 abstracts on average and the growth rate of new abbreviations is increasing. Liu *et al*. (2002) showed that 81.2% of abbreviations are ambiguous and have an average of 16.6 senses in MEDLINE abstracts.

Cascaded construction: One entity name may often be embedded in another entity name, e.g. ‘<PROTEIN><DNA>kappa 3</DNA>binding factor </PROTEIN>’. Around 16.57% of biomedical entity names have such cascaded construction in GENIA V3.0 (Zhou *et al.* 2004). It is therefore clear that the entity names in the biomedical domain are much more complex than those in the newswire domain, thus making it necessary to explore more evidential features to cope with the special phenomena of naming conventions in the biomedical domain.

1.3 GOALS OF THE THESIS

The goal of the work in this thesis is summarized below:

The proposed idea here is that instead of using a single algorithm for Named entity recognition we are trying to ensemble the two algorithms and combine its result. The two algorithms used are ID3 and C5. Unlike ID3, C5 is the latest algorithm in the decision tree learning field and it has many advantages. So here we are using ID3 and C5. First we find the F-measure of the original algorithms and then use the ensembled approach by combining the results of both the algorithms. Here we will see that the ensembled approach gives better result than each of the original classifiers.

1.4 ORGANIZATION OF THESIS

This thesis is organized as follows:

Chapter 2 discusses the previous work done in the field of Biomedical Named Entity Recognition. This includes the extensive study of various techniques used in Named Entity Recognition of the Biomedical domain. It also highlights some of the most relevant works in the direction of field of work presented in the thesis

Chapter 3 discusses in detail some of the basic terms and algorithms which are used in the thesis, e.g. C5 decision tree algorithm, memory based learner, class confidence etc. The reason for selecting a particular algorithm is also discussed in this section.

Chapter 4 focuses on the proposed algorithm for the Biomedical Named Entity Recognition. It presents the step by step detailed explanation of the algorithm.

Chapter 5 presents the dataset used and explanation of its various attributes.

Chapter 6 presents the results of the ensembled approach and the results of using this approach.

Chapter 7 presents the conclusions of the thesis and future work.

CHAPTER 2

LITERATURE SURVEY

The amount of biological literature published daily is growing exponentially. Medline alone contains 14 million abstracts and is a critical source of information for biologists and curators. As these scientists find it essential to search for information in an overabundance of documents, their need for text mining techniques tailored to the biological domain has become apparent.

The focus is on named entity recognition (NER) in biomedicine, a fundamental challenge for text mining due to the special problems caused by the complex nature of biological entity recognition, classification, and unique identification. This is a key factor for access to the information stored in literature, as it is the biological entities and their relationships that convey knowledge across scientific articles.

Textual terms (names of genes, proteins, gene products, organisms, drugs, chemical compounds, etc.) are the primary means of scientific communication because they are used in language to represent the concepts in the domain; it would be impossible to “understand” an article or to extract information from it without the precise identification and association of the terms. Biomedical terminology presents a special challenge. It is constantly changing; new terms are rapidly being introduced for each of the organisms being studied, while old ones are discarded (e.g., withdrawn or made obsolete). Biological names are very complex, as they are created and referenced by many different communities. They include an enormous amount of synonyms and variant forms, such as acronyms, and morphological, derivational, and orthographic variants, all of which are used interchangeably in the literature. In addition, many biological terms and their variants are ambiguous. They share their lexical representations with common English words (gene names/symbols, such as an, by, can, and for), or with other biomedical terms (gene names, such as demented, white eye, and hair loss). Existing text processing resources typically lack information that can support disambiguation of terms.

The collection of papers in this issue reports on diverse approaches that use a variety of natural language processing, corpus-based and machine learning techniques to recognize, classify, and/or identify biological entities. Recognition involves identifying the boundaries of the name in the text, whereas classification assigns a semantic class to the entity based on an appropriate ontology, and identification maps the term to a normalized form or to a unique identifier.

The paper by Morgan et al. [13] reports on a series of experiments related to the application of natural language processing as a tool to aid in curation of the FlyBase database. They used

Flybase resources, and a combination of techniques, such as pattern matching, tokenization, HMM-based tagging, disambiguation heuristics, etc., to automatically generate large quantities of high-quality training data to support the automatic learning of a gene name recognizer. The generation of normalized gene lists is also explored using simple pattern matching and an HMM gene name tagger.

Zhang and colleagues [14] adapt an HMM named entity recognizer to the biomedical domain via a rich feature set consisting of orthographic, morphological, part-of-speech, and semantic trigger features. These features are integrated via a HMM with back-off modeling. In addition, they propose methods for recognizing biomedical abbreviations and cascading (i.e., nested) named entities. Their treatment of the cascading phenomenon is novel as they recognize both the nested and the longest named entities. In their work, they propose two approaches for recognizing cascaded names: a post-processing rule-based approach and an HMM-based approach.

Variations of character-level orthographic features and part-of-speech (POS) features on the performance of NERs are examined by Collier and Takeuchi [15]. Their experiments, which are based on support vector machines (SVMs), revealed that orthographic features outperformed POS features. The reasons that POS features appear to be less useful than orthography are due to the complex relationship between name boundaries, local syntactic ambiguities, and class semantics. In addition, they demonstrate that the combination of orthographic features and POS degrades the overall performance of NERs slightly.

Lee et al. [16] present a two-phase named entity recognizer based on SVMs, which consists of two subtasks: a boundary identifier and a semantic classifier of named entities. This separation of the NER task allows the use of the appropriate SVM classifier and relevant features for each subtask, resulting in a reduction of computational complexity and improvement in performance. A hierarchical classification method is employed for semantic classification that utilizes 22 semantic classes that are based on the GENIA ontology [17].

An automatic method for mining collocates (i.e., two or more words that occur together much more frequently than by chance) in the literature is proposed by Hou and Chen [18]. They focus on collocations associated with gene and protein names, and use the extracted collocates to improve the precision rate of protein and gene name recognition. In addition, they integrate the results of multiple NERs, such as Yapex [19], KeX [20], ABGene [21], and Idgene [22], to improve the recall rates. The combination of filtering and integration strategies increased the performance of the NER.

Novel techniques for boosting the performance of dictionary-based protein name recognition are suggested by Tsuruoka and Tsujii [23]. They propose two alternative methods to tackle the problem of low recall due to spelling variations. One method uses approximate string matching, where similarity between two strings is computed based on an edit distance. What is interesting about their method is that the cost for individual operations varies depending on the letter being operated on (e.g., substitution of an alphabetic character costs more than substitution of a dash or a number). An alternate method, which is more efficient, involves expanding the dictionary in advance using a probabilistic variant generator. A method to filter out false positives is also presented, which is based on use of a naïve Bayes classifier.

The use of morphological analysis in protein name recognition to overcome problems such as boundary disagreement is proposed by Yamamoto and colleagues [24]. To overcome boundary disagreement that is caused by tokenization ambiguity, they apply techniques borrowed from Japanese (nonsegmented language) morphological analysis. The authors show that their augmented preprocessing improves the performance of protein name recognition over conventional preprocessing.

Spasic and Ananiadou [25] examine term classification for the task of ontology management, where it is of interest to automatically augment an ontology with novel terms. A genetic algorithm is used in order to refine verb selectional preferences and to assign classes to domain-specific verbs. The class of a newly discovered term is suggested depending on its co-occurrence with a domain-specific verb, as well as a similarity measure to known terms with established term-class relationships.

The topic of term classification (independently from the task of term identification) is examined by Torii and colleagues [26]. They focus on different sources of information that can be used for classification and report on their effectiveness. They apply machine learning methods to build a classifier, and they use both name-internal features (e.g., suffixes) and name-external features (e.g., contextual information) for the classification task.

To conclude this section, state-of-the art approaches to term identification are reviewed by Krauthammer and Nenadic [27]. The paper features an extensive list of work published in the domain. It analyzes the process of identifying terms through three steps: term recognition, term classification, and term mapping, which in some cases can be merged. For each step, the main approaches and general trends, along with the major problems, are discussed. Also, by identifying various challenges that term identification is still faced with, the review tries to delineate directions for future work in the field.

CHAPTER 3

CONCEPTS AND BASICS

3.1 NAMED ENTITY RECOGNITION

Named-entity recognition (NER) (also known as **entity identification** and **entity extraction**) is a subtask of information extraction that seeks to locate and classify atomic elements in text into predefined categories such as the names of persons, organizations, locations, expressions of times, quantities, monetary values, percentages, etc. Named entities are often the pivotal as well as the most information bearing elements of a text, and NER systems find application in a number of tasks like information extraction, text mining and machine translation. Due to its immense importance, a substantial amount of work has been carried out for NER system development in various languages and domains.

Most research on NER systems has been structured as taking an unannotated block of text, such as this one:

Jim bought 300 shares of Acme Corp. in 2006.

And producing an annotated block of text, such as this one:

<ENAMEX TYPE = "PERSON">Jim</ENAMEX>bought<NUMEX TYPE = "QUANTITY">300</NUMEX>shares of<ENAMEX TYPE = "ORGANIZATION">Acme Corp.</ENAMEX> in <TIMEX TYPE = "DATE">2006</TIMEX>

In this example, the annotations have been done using so-called ENAMEX tags that were developed for the Message Understanding Conference in the 1990s.

State-of-the-art NER systems for English produce near-human performance. For example, the best system entering MUC-7 scored 93.39% of F-measure while human annotators scored 97.60% and 96.95%.

3.2 ACTIVE LEARNING

Active learning is a special case of semi-supervised machine learning in which a learning algorithm is able to interactively query the user (or some other information source) to obtain the desired outputs at new data points. In statistics literature it is sometimes also called optimal experimental design.

There are situations in which unlabeled data is abundant but manually labelling it is expensive. In such a scenario, learning algorithms can actively query the user/teacher for labels. This type of iterative supervised learning is called active learning. Since the learner chooses the examples, the number of examples to learn a concept can often be much lower than the number required in normal supervised learning. In this approach, there is a risk that the algorithm be overwhelmed by uninformative examples.

Active learning (sometimes called “query learning” or “optimal experimental design” in the statistics literature) is a subfield of machine learning and, more generally, artificial intelligence. The key hypothesis is that if the learning algorithm is allowed to choose the data from which it learns it will perform better with less training. This a desirable property for learning algorithms to have. Consider that, for any supervised learning system to perform well, it must often be trained on hundreds (even thousands) of labelled instances. Sometimes these labels come at little or no cost. Learning systems use these flags and ratings to better filter your junk email and suggest movies you might enjoy. In these cases you provide such labels for free, but for many other more sophisticated supervised learning tasks, labelled instances are very difficult, time-consuming, or expensive to obtain.

Here are a few examples:

- **Speech recognition** : Accurate labelling of speech utterances is extremely time consuming and requires trained linguists. Zhu (2005a) reports that annotation at the word level can take ten times longer than the actual audio (e.g., one minute of speech takes ten minutes to label), and annotating phonemes can take 400 times as long (e.g., nearly seven hours). The problem is compounded for rare languages or dialects.
- **Information extraction**: Good information extraction systems should be trained using labelled documents with detailed annotations. Users highlight entities or relations of interest in text, such as person and organization names, or whether a person works for a particular

organization. Locating entities and relations can take a half-hour or more for even simple newswire stories (Settles et al., 2008a). Annotations for other knowledge domains may require additional expertise, e.g., annotating gene and disease mentions for biomedical information extraction usually requires PhD-level biologists.

- **Classification and filtering** : Learning to classify documents (e.g., articles or web pages) or any other kind of media (e.g., image, audio, and video files) requires that users label each document or media file with particular labels, like "relevant" or "not relevant." Having to annotate thousands of these instances can be tedious and even redundant.

Active learning systems make an attempt to overcome the labelling bottleneck by asking queries in the form of unlabeled instances to be labelled by an oracle (e.g., a human annotator). In this way, the active learner aims to achieve high accuracy using as few labelled instances as possible, thereby minimizing the cost of obtaining labelled data. Active learning is well-motivated in many modern machine learning problems where data may be abundant but labels are scarce or expensive to obtain. Note that this kind of active learning is related in spirit, though not to be confused, with the family of instructional techniques by the same name in the education literature (Bonwell and Eison, 1991).

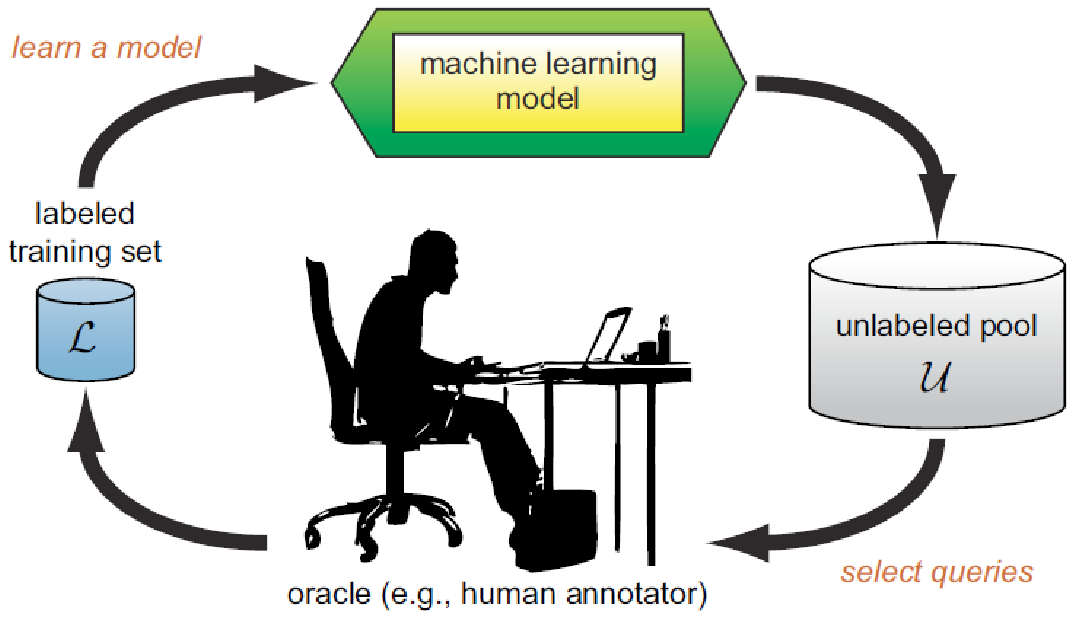


Figure 2 The pool-based active learning cycle ref. [1]

There are several scenarios in which active learners may pose queries, and there are also several different query strategies that have been used to decide which instances are most informative

Figure 2 illustrates the pool-based active learning cycle. A learner may begin with a small number of instances in the labelled training set L, request labels for one or more carefully selected instances, learn from the query results, and then leverage its new knowledge to choose which instances to query next. Once a query has been made, there are usually no additional assumptions on the part of the learning algorithm. The new labelled instance is simply added to the labelled set L, and the learner proceeds from there in a standard supervised way. There are a few exceptions to this, such as when the learner is allowed to make alternative types of queries, or when active learning is combined with semi-supervised learning.

3.3 ACTIVE ANNOTATION

Active annotation–the term introduced by [10] to refer to the application of active learning [11] to corpus creation–is becoming a popular annotation technique because it can lead to drastic reductions in the amount of annotation that is necessary for training a highly accurate statistical classifier. In the traditional, random sampling approach, unlabeled data is selected for annotation at random. In contrast, in active learning, the most useful data for the classifier are carefully selected. In a typical active learning setup, a classifier is trained on a small sample of the data (usually selected randomly), known as the seed examples. The classifier is subsequently applied to a pool of unlabeled data with the purpose of selecting additional examples that the classifier views as informative. The selected data is annotated and the cycle is repeated, allowing the learner to quickly refine the decision boundary between the classes. The key question in this approach is how to determine the samples that will be most useful to the classifier.

3.4 BASICS OF DECISION TREE

A **decision tree** is a decision support tool that uses a tree-like graph or model of decisions and their possible consequences, including chance event outcomes, resource costs, and utility. It is one way to display an algorithm. Decision trees are commonly used in operations research, specifically in decision analysis, to help identify a strategy most likely to reach a goal.

Decision tree learning, used in statistics, data mining and machine learning, uses a decision tree as a predictive model which maps observations about an item to conclusions about the item's target value. More descriptive names for such tree models are **classification trees** or **regression trees**. In these tree structures, leaves represent class labels and branches represent conjunctions of features that lead to those class labels.

In decision analysis, a decision tree can be used to visually and explicitly represent decisions and decision making. In data mining, a decision tree describes data but not decisions; rather the resulting classification tree can be an input for decision making. This section deals with decision trees in data mining.

Decision tree learning is the construction of a decision tree from class-labelled training tuples. A decision tree is a flow-chart-like structure, where each internal (non-leaf) node denotes a test on an attribute, each branch represents the outcome of a test, and each leaf (or terminal) node holds a class label. The topmost node in a tree is the root node.

There are many specific decision-tree algorithms. Notable ones include:

1. ID3 (Iterative Dichotomiser 3)
2. C4.5 (successor of ID3)
3. CART (Classification And Regression Tree)
4. CHAID (CHi-squared Automatic Interaction Detector). Performs multi-level splits when computing classification trees.[9]
5. MARS: extends decision trees to better handle numerical data.
6. C5 : This is the latest algorithm in the decision tree algorithm.

The reasons for decision learning tree algorithms to be attractive are: -

1. They generalize in a better way for unobserved instances, once examined the attribute value pair in the training data.
2. They are efficient in computation as it is proportional to the number of training instances observed.
3. The tree interpretation gives a good understanding of how to classify instances based on attributes arranged on the basis of information they provide and makes the classification process self-evident.

The two algorithms used in this thesis work are –

1. Id3 Algorithm
2. C5 Algorithm

A brief introduction of the two algorithms is discussed below and how this algorithm is used in our project is discussed in Section -

3.4.1 Id3 Algorithm

In decision tree learning, **ID3** (**Iterative Dichotomiser 3**) is an algorithm invented by Ross Quinlan used to generate a decision tree from a dataset. ID3 is the precursor to the C4.5 algorithm, and is typically used in the machine learning and natural language processing domains.

ID3 (Iterative Dichotomized) algorithm is based on the Concept Learning System (CLS) algorithm. CLS algorithm is the basic algorithm for decision tree learning. The tree growth phase of CLS is the matter of choosing attribute to test at each node is by the trainer. ID3 improves CLS by adding a heuristic for attribute selection. ID3 is based on Hunt's algorithm and is implemented in serially. This algorithm recursively partitions the training dataset till the record sets belong to the class label using depth first greedy technique. In growth phase of the tree construction, this algorithm uses information gain, an entropy based measure, to select the best splitting attribute, and the attribute with the highest information gain is selected as the splitting attribute. ID3 doesn't give accurate result when there is too much noise or details in the training data set, thus an intensive pre-processing of data is carried out before building a decision tree model with ID3. One of the main drawbacks of ID3 is that the measure Gain used tends to favour attributes with a large number of distinct values. It only accepts categorical attributes in building a tree model. This decision tree algorithm generates variable branches per node.

The ID3 algorithm begins with the original set Sas the root node. On each iteration of the algorithm, it iterates through every unused attribute of the set $S$and calculates the entropy $H(S)$(or information gain $IG(A)$) of that attribute. Then selects the attribute which has the smallest entropy (or largest information gain) value. The set $S$is then split by the selected attribute (e.g. age < 30 and age >= 30) to produce subsets of the data. The algorithm continues to recurse on each subset. When every element in a subset belongs to the same class, this subset will no longer be recursed on, and this node in the decision tree becomes a terminal node with a class label same as the class all its elements belong to. The ID3 algorithm terminates when every subset is classified. Throughout the algorithm, the decision tree is constructed with each non-terminal node representing the selected attribute on which the data was split, and terminal nodes representing the class label of the final subset of this branch.

ID3 does not guarantee an optimal solution; it can get stuck in local optimums. It uses a greedy approach by selecting the best attribute to split the dataset with each iteration. One improvement that can be made on the algorithm can be to use backtracking during the search for the optimal decision tree.

ID3 can overfit to the training data, to avoid overfitting, one should prefer smaller decision trees over larger ones. This algorithm usually produces small trees, but it does not always produce the smallest possible tree.

ID3 is also harder to use on continuous data. If the values of any given attribute is continuous, then there are many more places to split the data on this attribute, and searching for the best value to split by can be time consuming.

Reasons for using ID3 algorithm

- Understandable prediction rules are created from the training data.
- Builds the fastest tree.
- Builds a short tree.
- Only need to test enough attributes until all data is classified.
- Finding leaf nodes enables test data to be pruned, reducing number of tests.
- Whole dataset is searched to create tree
- Builds decision tree in minimum steps

o The most important point while tree induction is collecting enough reliable associated data over specific properties.

- o Asking right questions determines tree induction.
- Each level benefits from previous level choices

3.4.2 C5 Algorithm

C5.0 algorithm is an extension of C4.5 algorithm. C5.0 is the classification algorithm which applies in big data set. C5.0 is better than C4.5 on the efficiency and the memory. C5.0 model works by splitting the sample based on the field that provides the maximum information gain. The C5.0 model can split samples on basis of the biggest information gain field..The sample subset that is get from the former split will be split afterward. The process will continue until the sample subset cannot be split and is usually according to another field. Finally, examine the lowest level split, those sample subsets that don't have remarkable contribution to the model will be rejected.

In our dataset C5.0 algorithm is used to split up data set & find out the result in the form of decision tree or rule set.

1) Splitting criteria used in C5.0 algorithm is information gain. The C5.0 model can split samples on basis of the biggest information gain.

2) Test criteria is decision tree have any number of branches available not fixed branches like CART.

3) Pruning method performed after creating decision tree i.e. post pruning single pass based on binomial confidence limits.

4) Speed of C5.0 algorithm is significantly faster than C4.5 & more accurate.

Reasons for using C5 algorithm (its advantages over C4.5)

C4.5 is a widely-used free data mining tool that is descended from an earlier system called ID3 and is followed in turn by See5/C5.0.

- C5.0 incorporates several new facilities such as variable misclassification costs. In C4.5, all errors are treated as equal, but in practical applications some classification errors are more serious than others. C5.0 allows a separate cost to be defined for each predicted/actual class pair; if this option is used, C5.0 then constructs classifiers to minimize expected misclassification costs rather than error rates.
- The cases themselves may also be of unequal importance. In an application that classifies individuals as likely or not likely to "churn," for example, the importance of each case may vary with the size of the account. C5.0 has provision for a case weight attribute that quantifies the importance of each case; if this appears, C5.0 attempts to minimize the weighted predictive error rate.
- C5.0 has several new data types in addition to those available in C4.5, including dates, times, timestamps, ordered discrete attributes, and case labels. In addition to missing values, C5.0 allows values to be noted as not applicable. Further, C5.0 provides facilities for defining new attributes as functions of other attributes.
- Some recent data mining applications are characterized by very high dimensionality, with hundreds or even thousands of attributes. C5.0 can automatically winnow the attributes before a classifier is constructed, discarding those that appear to be only marginally relevant. For high-dimensional applications, winnowing can lead to smaller classifiers and higher predictive accuracy, and can often reduce the time required to generate rulesets.

- C5.0 is also easier to use. Options have been simplified and extended -- to support sampling and cross-validation, for instance -- and C4.5's programs for generating decision trees and rulesets have been merged into a single program.

3.5 ENSEMBLING OF CLASSIFIERS

The main aim for classifiers ensemble in our study is to improve recognition accuracy. Classifier ensembles are effective methods for machine learning and can improve the classification performance of individual classifiers. There are several approaches for ensemble of classifiers. Here, three distinct strategies are discussed to combine multiple predictions from separate base classifiers.

The strategies are: -

1. Arbitration rules

2. Stacked generalization including class-stacking and class-attribute-stacking

3. Cascade generalization

1. The approach of arbitration rules uses heuristic rules to judge which prediction to be selected if the participating base learners cannot reach a consensus decision. Figure 3 depicts how the final prediction is made with input predictions of base learners using arbitration rule. Let x be an instance whose classification to be confirmed, and *Ck(x)*, k = 1, 2, ..., K be the predicted classes of x from the Kth classifier models *Mk*, k = 1, 2, ..., K. We first compute the classification predicted by each of the single classifiers. The arbitration rule is then applied to judge which prediction is selected if the participating base classifiers cannot reach a consensus decision.

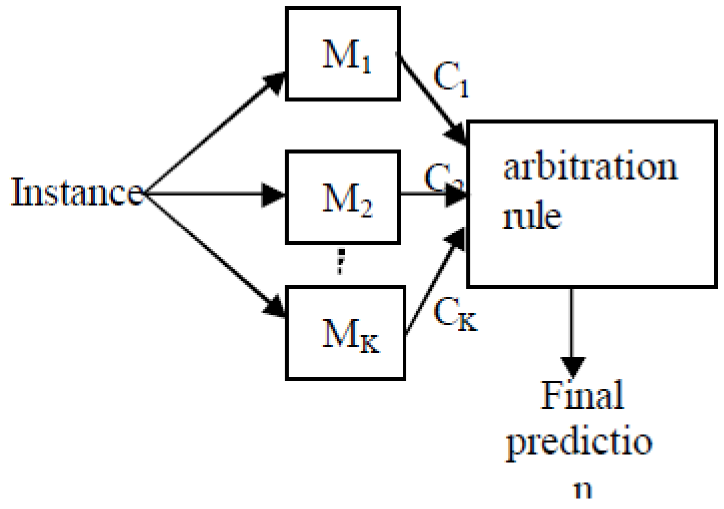


Figure 3 Classifiers Ensemble using Arbitration rule

2. Meta-learning is loosely defined as learning from the output of concept learning systems [5]. Meta-learning studies how a learning system can increase the efficiency through experiences, one common approach to meta-learning is known as stacked generalization also called stacking [6]. In stacking method the transformation of the training set conveys information about the predictions of the base-classifiers. After transforming the original training set, each example contains the original predictions of the base-classifiers, and may also contain the feature vectors.
3. The following notations are defined before introducing the details of the stacking strategy. Let x be an instance whose classification to be confirmed, and *Ck(x), k* = 1, 2, ..., *K* be the predicted classes of x from the *K*th classifier models *Mk, k* = 1, 2, ..., *K*. *class(x)* and *attrvec(x)* denote the correct classification and attributes vector of example x, respectively. Given a data set *D* = *{(class(xi), attrvec(xi)), i* = 1,…,*I*}, we randomly split the data into *J* almost equal parts *D*1,…,*Dj*, and define *Dj* and *D(-j)=D- Dj* to be the test and training sets for the *j*th fold of a J-fold cross-validation..

Given *K* learning algorithms, which we call level-0 generalizers, the *k*th algorithm will be invoked on the data in the training set *D(-j)* to induce a classifier model *Mk (-j)* , for *k* = 1,…,*K*. These classifier models are called level-0 models. For each instance xi in *Dj*, the test set for the *j*th cross validation fold, let *Cki(xi)*denote the prediction of the classifier model *Mk (-j)* on *xi*.

At the end of the entire cross-validation process, the data set assembled from the outputs of the K classifier models is *DCV* = {(*class (xi)*, *C1i(xi)*,…, *CKi(xi)*), *i* = 1,…,*I*}. *DCV* is the level-1 data. In level-1, a learning algorithm, i.e. the generalizer is used to derive a model *MStack* from *DCV*. Figure 4 illustrate the cross-validation process. To complete the training process, the final level-0 models *Mk*, *k* = 1,…,*K*, are derived using all the data in D.

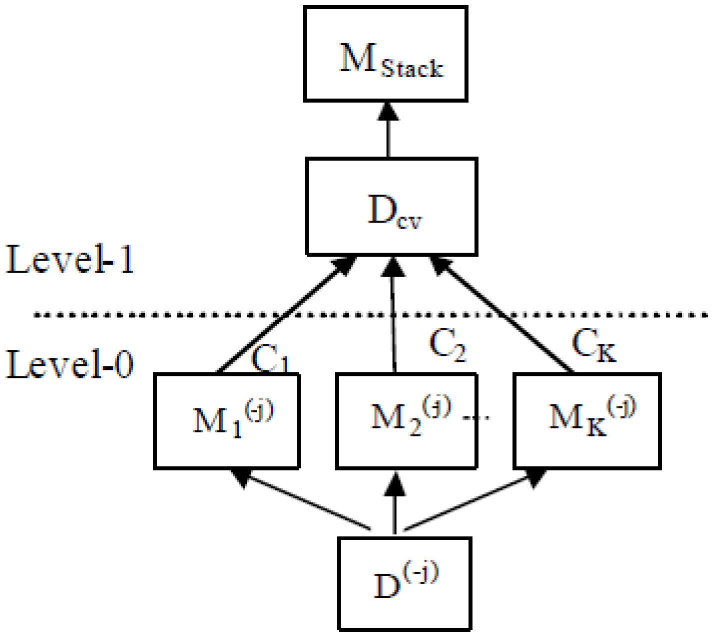


Figure 4 J-Fold cross-validation process in level – 0 and the level – 1 model produced process

Now let us consider the classification process, in which the models *Mk*, *k* = 1,…,*K*, are used in conjunction with *MStack*. Given a new instance, models *Mk* , *k* = 1,…,*K*, produce a vector (*C1(x)*,…, *CK(x)*). This vector is the input to the level-1 model *MStack*, whose output is the final classification result for that instance[7]. This classification process is shown in figure 5.

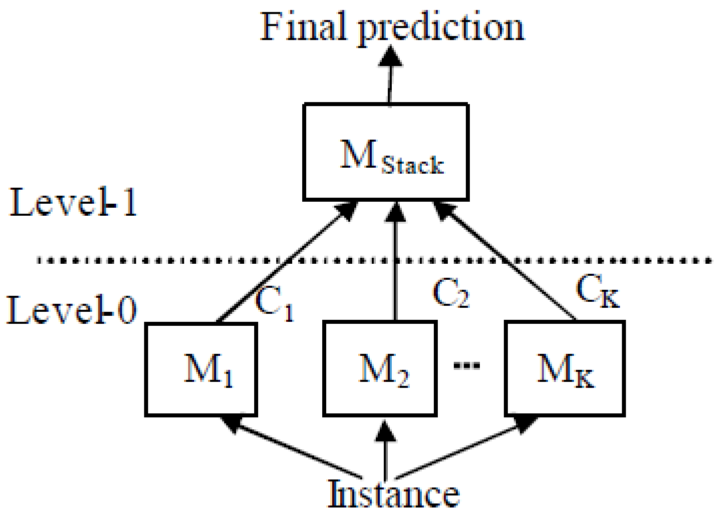


Figure 5 Classification process of stacking

For the training set, we have both the predictions of the base classifiers and the true class. The matrix containing the predictions of the base classifiers as predictors and corresponding true classes for training cases will be called the meta-data set. The classifier trained on this matrix will be called the meta-classifier. We experiment with two stacking schemes which determined by the content of training examples for the meta-classifier.

Class-stacking The meta-level train ing instances consist of the correct classification and the predictions from base classifiers, i.e., *T* = {(*class(x)*,*C1(x)*,*C 2(x)*, ...,*C K(x)*) | *x* ? *D*}.

Class attribute stacking The meta-level training instances consist of the correct classification and the predictions from base classifiers with the addition of the attribute vectors, i.e., *T*= {(*class (x),C1(x),C2(x),* .... *,CK(x), attrvec(x)*) | *x* ? *D*}. Cascade generalization [8] is defined as combining the learning algorithms in sequence essentially. The meta-level training instances are obtained by adding the predicted classifications of classifiers to *attrvec(x)* in sequence, i.e., *Tk* = {(*class(x), Ck-1(x), attrvec(x)*) | *x* ? *D*}, *Ck-1(x)* is obtained by level-(*K*-2) classifiers which is shown in figure 6.

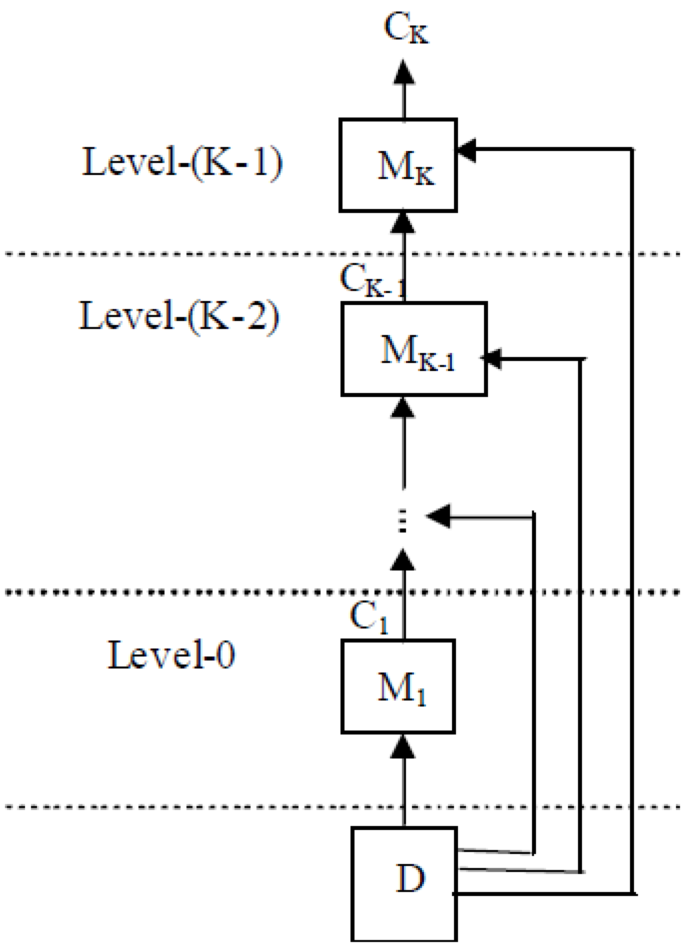


Figure 6 Classifiers Ensemble using cascade generalization

CHAPTER 4

PROPOSED WORK

4.1 INTRODUCTION

For each of the base classifiers, ID3 and C5, a feature vector consisting of the features described in the section 5.2 is extracted for each word in the annotated data. Now, we have a training data in the form (Wi, Ti), where, Wi is the ith word and its feature vector and Ti is its output class. We consider the feature vector consisting of all the features of the current token and varied the contexts within wi+2 i−2 = wi−2 . . .wi+2. Our technique is based on the combined decisions of both ID3 and C5. As both algorithms produce two different kinds of probabilistic scores, we first normalize all the confidence values within the range [0,1]. We consider these as the actual confidence scores of outputs. Our proposed selection criterion is based on the differences between the confidence values of the most probable two classes for a token, the hypothesis being that items for which this difference is smaller are those of which the classifier is less certain. A threshold on the confidence interval is defined, and for each base classifier we generate a set of uncertain samples. These sets contain the selected sentence identifiers along with the confidence intervals for which they are included into the respective sets. Thereafter we combine the decisions of ID3 and C5, and generate a new set of uncertain samples by taking the unions of these two sets. The union is taken in such a way that the common sentence is assumed to have the confidence interval, equal to the minimum of two values assigned to that particular sentence in two sets. Finally, we select 10 most uncertain sentences from the development data. Thus, in each iteration of the algorithm, we actually add 10 most informative sentences to the training set. We run the algorithm for the maximum 10 iterations. In some cases, the performance starts to decrease even at the earlier step of the algorithm. In order to account this fact we stop the algorithm's iteration, and retrieve last iteration's training data as the final one.

The main steps of the proposed active annotation are shown in Table 1.

4.2 PROPOSED IDEA

The proposed idea here is that instead of using a single algorithm for Named entity recognition we are trying to ensemble the two algorithms and combine its result. The two algorithms used are ID3 and C5. Unlike ID3, C5 is the latest algorithm in the decision tree learning field and it has many advantages. So here we are using ID3 and C5. First we find the F-measure of the original algorithms and then use the ensembled approach by combining the results of both the algorithms. Here we will see that the ensembled approach gives better result than each of the original classifiers.

The proposed algorithm is shown in the table on the next page. It shows the step by step execution of the ensemble algorithm.

Step 1	Train the base classifiers with the initial training data and evaluate with the gold standard test data.
Step 2	Train the base classifiers with the initial training data and evaluate with the development data.
Step 3	Calculate the confidence value of each token of each output class.
Step 4	Normalize the confidence scores within the range of [0,1]
Step 5	Compute the confidence interval (C.I.) between the two most probable classes for each token of the development data computed on outputs of both ID3 and C5.
Step 6	From each of the output perform the following operations:
Step 6.1	If C.I. is below the threshold value (set to 0.2) then add the NE token along with its sentence identifier and CI in a set of effective sentences, selected for active annotation
Step 6.2	Create two different sets (Set ID3 and Set C5) for two classifiers.
Step 7	Combine two sets into one, named as EA in such a way that if the sentence identifiers are same, then for that sentence $CI_{New} = \min(CI_{C5}CI_{ID3})$. All the dissimilar sentences are added as they are.
Step 8	Sort EA in ascending order of CI_{new}.
Step 9	Select the top most 10 sentences, and remove these from the development data.
Step 10	Add the sentences to the training set. This generates new training set. Retrain the ID3 and C5 classifiers and evaluate with the test set.
Step 11	Repeat steps 3-10 for some iteration(10 in our case)

Table 1 Main Steps of the proposed ensemble technique for active annotation

Detailed Explanation of the proposed Algorithm

- Training the base classifiers

  Here in our project we have used two classifiers as our base classifier, so we will be training both our classifiers one by one. They are ID3 and C5 algorithm; both are a type of decision tree algorithm.

  ID3 :- We train our genia corpus biomedical data on the ID3 algorithm. The ID3 algorithm is implemented in java on windows platform.

  C5 :- We again train our genia corpus biomedical data on the C5 algorithm. The C5 algorithm here is implemented in C on the Ubuntu platform.

- Calculate the confidence value

  In this step we calculate the confidence value with both the algorithm i.e. ID3 and C5. Here the confidence value is calculated for each token of each output class. The output of both the algorithm is the output class along with the confidence value with which each class is selected.

- Normalize the confidence scores

  The confidence values are normalized in the range of [0,1].

- Compute the Confidence Interval

  Here the confidence Interval is calculated. Confidence Interval is the difference between the confidence values of the two most probable classes i.e. the best class confidence value and the second best class confidence value. This confidence interval is calculated on output of both the algorithm

- Selecting the Confidence Interval below a threshold

Here the threshold set is 0.2; if the confidence interval happens to be below 0.2 then the Named entity token along with the confidence interval is added to a set of effective sentences. This set of effective sentences is further used for active annotation.

The concept used here is that higher the Confidence Interval between the two most probable classes the better the result.

Suppose the confidence interval between the best class selected and the second best class selected is smaller, that means there is lesser clarity between which class should be selected to annotate the sentence. The probability of the second best class being the correct one is higher.

- The Above process is repeated for both the algorithms, so we get two different sets of effective sentences i.e. Set ID3 and Set C5. These two sets will be used for active annotation.
- Combine the two sets into one named as EA. We can say that take a union of the two sets such that all the dissimilar sentences are added in the set EA.
- Sort the set in the ascending order.
- Select the topmost 10 sentences and remove these sentences from the dataset.
- Train these top 10 sentences manually and assign a class to these sentences.
- Repeat the step 3 – 10 for 10 iterations.

4.3 Modules

Our project has 3 modules :

1. ID3 classification module
2. C5 classification module
3. Ensemble module

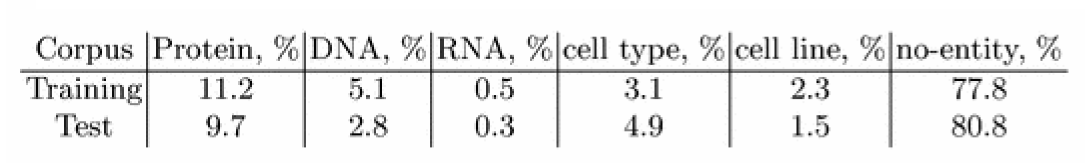
CHAPTER 5

DATASET USED

5.1 THE GENIA CORPUS

Any supervised machine-based model depends on a corpus that has been used to train it. At the moment the largest and, therefore, the most popular biomedical annotated corpus is Genia corpus v. 3.02 which contains 2,000 abstracts from the MEDLINE collection annotated with 36 biomedical entity classes. In our experiments, we have used its JNLPBA version.

The JNLPBA corpus is annotated with 5 classes of biomedical entities: protein, RNA, DNA, cell type and cell line. Biomedical entities are tagged using the IOB2 notation. In Table 1 a tag distribution within the training and test corpora is shown. It can be seen that the majority of words (about 80%) does not belong to any biomedical category. Furthermore, the biomedical entities themselves also have an irregular distribution: the most frequent class (protein) contains about 10% of words approximately, whereas the most rare one (RNA) – less than 0.5%. The tag irregularity may cause a confusion among different types of entities with a tendency for any word to be referred to the most numerous class.



Corpus	Protein, %	DNA, %	RNA, %	cell type, %	cell line, %	no-entity, %
Training	11.2	5.1	0.5	3.1	2.3	77.8
Test	9.7	2.8	0.3	4.9	1.5	80.8

Table 2 Entity tag distribution in the training and test corpora

5.2 NAMED ENTITY FEATURES

Feature selection plays an important role for the success of machine leaning techniques. We use a large number of following features for constructing our classifiers. These features are easy to derive and don't require deep domain knowledge and/or external resources for their generation. Thus, these features are general in nature and can be applied for other domains as well as languages. Due to the use of variety of features, the individual classifiers achieve very high accuracies.

1. **Context words:** These are the words occurring within the context window $w^{i+3}_{i-3} = w_{i-3} \ldots w_{i+3}$, $w^{i+2}_{i-2} = w_{i-2} \ldots w_{i+2}$ and $w^{i+1}_{i-1} = w_{i-1} \ldots w_{i+1}$ where w_i is the current word. This feature is considered with the observation that surrounding words carry effective information for identification of NEs.
2. **Word prefix and suffix:** These are the word prefix and suffix character sequences of length up to *n*. The sequences are stripped from the leftmost (prefix) and rightmost (suffix) positions of the words. We set the feature values to 'undefined' if either the length of w_i is less than or equal to n-1, w_i is a punctuation symbol or if it contains any special symbol or digit. We experiment with n=3(i.e., 6 features) and 4 (i.e., 8 features) both.

3. **Word length:** We define a binary valued feature that fires if the length of w_i is greater than a pre-defined threshold. Here, the threshold value is set to 5. This feature captures the fact that short words are likely not to be NEs.

4. **Infrequent word:** A list is compiled from the training data by considering the words that appear less frequently than a predetermined threshold. The threshold value depends on the size of the dataset. Here, we consider the words having less than 10 occurrences in the training data. Now, a feature is defined that fires if w_i occurs in the compiled list. This is based on the observation that more frequently occurring words are rarely the NEs.

5. **Part of speech (POS) information:** POS information is a critical feature for NERC. In this work, we use POS information of the current and/or the surrounding token(s) as the features. This information is obtained using GENIA tagger V2.0.2[3], which is used to extract POS information from the biomedical domain. The accuracy of the GENIA tagger is 98.26%.

6. **Chunk information:** We use GENIA tagger V2.0.2 to get the chunk information. Chunk information (or, shallow parsing features) provides useful evidences about the boundaries of biomedical NEs. In the current work, we use chunk information of the current and/or the surrounding token(s).

7. **Dynamic feature:** Dynamic feature denotes the output tags $t_{i-3}t_{i-2}t_{i-1}, t_{i-2}t_{i-1}, t_{i-1}$ of the word $w_{i-3}w_{i-2}w_{i-1},\ w_{i-2}w_{i-1},\ w_{i-1}$ preceding w_i in the sequence w_1^n. For CRF, we consider the bigram template that considers the combination of the current and previous output labels.

8. **Unknown token feature:** This is a binary valued feature that checks whether the current token was seen or not in the training corpus. In the training phase, this feature is set randomly.

9. **Word normalization:** We define two different types of features for word normalization. The first type of feature attempts to reduce a word to its stem or root form. This helps to handle the words containing plural forms, verb inflections, hyphen, and alphanumeric letters. The second type of feature indicates how a target word is orthographically constructed. Words shapes refer to the mapping of each word to their equivalence classes. Here each capitalized character of the word is replaced by ‘A’, small characters are replaced by ‘a’ and all consecutive digits are replaced by ‘0’. For example, ‘IL’ is normalized to ‘AA’, ‘IL-2’ is normalized to ‘AA-0’ and ‘IL-88’ is also normalized to ‘AA-0’.
10. **Head nouns:** Head noun is the major noun or noun phrase of a NE that describes its function or the property. For example, transcription factor is the head noun for the NE NF-kappa B transcription factor. In comparison to other words in NE, head nouns are more important as these play a key role for correct classification of the NE class. In this work, we use only the unigram and bigram head nouns like receptor, protein, binding protein etc. For domain independence, we extract these head nouns from the training data only. These are compiled to generate a list of 912 entries that contain only the most frequently occurring head nouns. Apart from these head nouns, we also consider the unigrams and bigrams extracted from the left ends of the NEs of the training data. A list of 578 entries is created by considering only the most frequent such n-grams. A feature is defined that fires if and only if the current word or the sequence of words appears in either of these lists.
11. **Verb trigger:** These are special type of verb (e.g. binds, participates etc.) that occur preceding to NEs and provide useful information about the NE class. To maintain the nature of domain independence, these trigger words are extracted automatically from the training

corpus based on their frequencies of occurrences. A feature is then defined that fires iff the current word appears in the list of trigger words.

12. **Word class feature:** Certain kind of NEs, which belong to the same class, are similar to each other. The word class feature is defined as follows: For a given token, capital letters, small letters, numbers and non-English characters are converted to “A”, ”a”, ”O” and ”-”, respectively. Thereafter, the consecutive same characters are squeezed into one character. This feature will group similar names into the same NE class.

13. **Informative words:** In general, biomedical NEs are too long and they contain many common words that are actually not NEs. For example, the function words such as of, and etc; nominals such as active, normal etc. appear in the training data often more frequently but these don’t help to recognize NEs. In order to select the most important effective words, we first list all the words which occur inside the multiword NEs. Thereafter digits, numbers and various symbols are removed from this list. For each word (w_i) of this list, a weight is assigned that measured how better the word is to identify and/or classify the NEs. This weight is denoted by NEweight (w_i), and calculated as follows:

    The effective words are finally selected based on the two parameters, namely NEweight and number of occurrences. The threshold values of these two parameters are selected based on some experiments. The words which have less than two occurrences inside the NEs are not considered as informative. The remaining words are divided into the following classes:

    Class1: This includes the words that occur more than 100 times. Here, we consider those words whose NEweights are greater than 0.4.

Class 2: this includes the words having occurrences >=20 and <100, Here, we set NEweight >= 0.6.

Class 3: This class includes the words having occurrences >=10 and <20. Here, we chose NEweight >= 0.6.

Class 4: This class includes the words having occurrences >= 5 < 10. Here, we chose NEweight >= 0.85.

Class 5: This includes the words having occurrences < 5. Here, we chose NEweight > 1.0.

We compile five different lists for the above five classes of informative words. A binary feature vector of length five is defined for each word. If the current word in training (or, test) is found in any particular list then the value of the corresponding feature is set to 1.

14. **Orthographic features:** We define a number of orthographic features depending upon the contents of the word forms. Several binary features are defined which use capitalization and digit information. These features are: initial capital, all capital, capital in inner, initial capital then mix, only digit, digit with special character, initial digit then alphabetic, digit in inner. The presence of some special characters like (‘,’, ‘-’, ‘.’, ‘)’, ‘(’, etc.) is very much helpful to detect NEs especially in biomedical domain. For example, many biomedical NEs have ‘-’ (hyphen) in their construction. Some of these special characters are also important to detect boundaries of NEs. We also use the features that check the presence of ATGC sequence and stop words.

CHAPTER 6

RESULTS

Classifiers are evaluated in terms of the standard recall, precision and F-measure.

Precision is the ratio of number of correctly found Named entity chunks (i.e. more than one token) to the number of found Named entity chunks, and recall is the ratio of the number of correctly found Named entity chunks to the number of true Named entity chunks. From the definition we can see that while recall tries to increase the number of correctly tagged entries from the entire data set as much as possible, precision tries to increase the number of correctly tagged entries from the total number of identified entries. These two capture two different classification qualities.

The value of the metric F - measure, which is the weighted harmonic mean of recall and precision, is calculated as below

$$F_\beta = \frac{(1+\beta^2)(recall \times precision)}{\beta^2 \times precision + recall}, \quad \beta = 1$$

Iteration Number	ID3			C5			Ensembled		
	Recall	**Precision**	**F-measure**	**Recall**	**Precision**	**F-measure**	**Recall**	**Precision**	**F-measure**
1	74.4	72.0	73.1	75.2	73.1	74.13	76.3	74.4	75.33
2	74.4	72.2	73.28	75.3	73.2	74.23	76.3	74.5	75.38
3	74.6	72.2	73.3	75.3	73.6	74.44	76.5	74.7	75.58
4	75.2	72.2	73.96	75.6	73.8	74.68	77.2	74.1	75.61
5	75.3	73.2	74.23	75.8	74.0	74.88	77.6	74.4	75.96
6	75.4	73.5	74.43	76.0	74.0	74.98	77.7	75.0	76.32
7	75.4	73.6	74.48	76.3	74.3	75.28	77.9	75.2	76.52
8	76.3	73.9	75.08	76.5	74.7	75.58	78.2	75.8	76.98
9	76.5	74.2	75.33	76.8	74.7	75.73	78.5	76.4	77.43
10	76.5	74.2	75.33	76.8	74.8	75.78	78.6	76.8	77.68

Table 3 Evaluation result of Active Learning with ID3, C5 and Ensembled Algorithms

CHAPTER 7

CONCLUSIONS AND FUTURE WORK

In this paper we have proposed an ensemble based active annotation technique that could be helpful for many applications where there is a scarcity in the amount of available labeled data, and preparation of these data is both time consuming and cost-intensive. The proposed approach is evaluated for solving the problem of NER, an important pipelined module in many NLP application areas. We develop a ensemble based model by combining two supervised classifiers, namely ID3 and C5. The proposed system is evaluated for the biomedical dataset. Evaluation shows satisfactory performance for the available dataset. Evaluation also suggest that the proposed approach i.e ensembling the classifiers can effectively improve the performance over the individual model (i.e. ID3 or C5).

In the future we can employ three algorithms instead of two to see if there is a performance improvement. Systematic feature selection is also essential to study the effectiveness of the chosen features.

REFERENCES

[1] B. Settles. Active learning literature survey. In In Computer Sciences Technical Report 1648 University of Wisconsin-Madison, 2009.

[2] Information on See5/C5.0 - RuleQuest Research Data Mining Tools, 2011. [Online] Available: http://www.rulequest.com/see5-info.html

[3] V. N. Vapnik. The nature of statistical learning theory. Springer-Verlag New York, Inc., New York, NY, USA, 1995.

[4] GENIA Project : http://www.geniaproject.org

[5] Zhou G, Zhang J, Su J, Shen D, Tan C. (2004). Recognizing names in biomedical texts: a machine learning approach. Bioinformatics, Vol. 20, No. 7. pp. 1178-1190.

[6] Zhou G. D. and Su J. (2002) Named entity recognition using an HMM-based chunk tagger. Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL'2002). Philadelphia, July, pp. 473–480.

[7] Zhou G. (2004) Recognizing Names in Biomedical Texts using Hidden Markov Model and SVM plus Sigmoid. International Joint Workshop on Natural Language Processing in Biomedicine and its Applications (JNLPBA): 2004.

[8] Zhou G., Su J. (2004) Exploring deep knowledge resources in biomedical name recognition. In : Proceedings of the joint workshop on natural language processing in biomedicine and its applications, Geneva, Switzerland; pp. 96-9.

[9] Ohta T, Tateisi Y, Kim J, Mima H, Tsujii J. (2002) The GENIA corpus: an annotated research abstract corpus in molecular biology domain. In: Proceedings of HLT 2002;

[10] A. Vlachos. Active annotation. In Proc. EACL 2006 Workshop on Adaptive Text Extraction and Mining, Trento, 2006.

[11] B. Settles. Active learning literature survey. In In Computer Sciences Technical Report 1648 University of Wisconsin-Madison, 2009.

[12] Available at: <http://www.ncbi.nlm.nih.gov/PubMed/>. Accessed July 22, 2004

[13] A.A. Morgan, L. Hirschman, M. Colosimo, A. Yeh, J. Colombe Gene name identification and normalization using a model organism database J. Biomed. Inform., 37 (6) (2004)

[14] J. Zhang, D. Shen, G. Zhou, J. Su, C.L. Tan Enhancing HMM-based biomedical named entity recognition by studying special phenomenaJ. Biomed. Inform., 37 (6) (2004)

[15] N. Collier, K. Takeuchi Comparison of character-level and part of speech features for name recognition in bio-medical texts J. Biomed. Inform., 37 (6) (2004)

[16] K.J. Lee, Y.S. Hwang, S. Kim, H.C. Rim Biomedical named entity recognition using two phase model based on SVMs J. Biomed. Inform., 37 (6) (2004)

[17] Ohta, T, Tateisi Y, Kim J, Mima H, Tsujii J-I. The GENIA Corpus: an annotated research abstract corpus in molecular biology domain. In: The 2nd International Conference on Human Language Technology

[18] W.J. Hou, H.H. Chen Enhancing performance of protein and gene name recognizers with filtering and integration strategies J. Biomed. Inform., 37 (6) (2004)

[19] Olsson F, Eriksson G, Franzen K, Asker L, Liden P. Notions of correctness when evaluating protein name taggers. In: Proceeedings of the 19th International Conference on Computational Linguistics 2002

[20] K. Fukuda, T. Tsunoda, A. Tamura, T. Takagi Toward information extraction identifying protein names from biological papers Pac. Symp. Biocomput. (2003)

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

[22] Fan JW. Information retrieval and extraction for the Chinese Gene Variation Database (CGVdb). Unpublished Master's Thesis

[23] Y. Tsuruoka, J. Tsujii Improving the performance of dictionary-based approaches in protein name recognition J. Biomed. Inform., 37 (6) (2004)

[24] K. Yamamoto, T. Kudo, A. Konagaya, Y. Matsumoto Use of morphological analysis in protein name recognition J. Biomed. Inform

[25] I. Spasic, S. Ananiadou Using automatically learnt verb selectional preferences for classification of biomedical terms J. Biomed. Inform., 37 (6) (2004)

[26] M. Torii, S. Kamboj, K. Vijay-Shanker Using name-internal and contextual features to classify biological terms J. Biomed. Inform., 37 (6) (2004)

[27] M. Krauthammer, G. Nenadic Term identification in the biomedical literature J. Biomed. Inform., 37 (6) (2004)