VICTORIA UNIVERSITY OF WELLINGTON
Te Whare Wananga o te Upoko o te Ika a Maui

Computer Science

PO Box 600
Wellington
New Zealand

Tel: +64 4 463 5341
Fax: +64 4 463 5045
Internet: office@mcs.vuw.ac.nz

Classification of Scientific Papers Using Machine Learning

Minh Duc Cao

Supervisors: Dr. Xiaoying(Sharon) Gao
Dr. Mengjie Zhang

Submitted in partial fulfilment of the requirements for
Bachelor of Science with Honours in Computer Science.

Abstract

The project aims to develop a domain-independent and adaptive approach for scientific document classification using both information from document contents and citation links. We evaluate several content-based classification methods including K-nearest neighbours, nearest centroid, naive Bayes and decision trees and find that the naive Bayes outperform other when training set is sufficiently large. Using phrases in addition to words and a good feature selection strategy such as information gain is found to improve system accuracy in comparison with using words only. To combine citation links for classification, the project proposes two methods, linear labelling update and probabilistic labelling update. The two methods iteratively update the labellings of classified documents using categories information from neighbouring documents. Our experiments on the two methods show that, combining contents and citations significantly improves the system performance.

Acknowledgments

I would like to thank both of my supervisors Xiaoying(Sharon) Gao and Mengjie Zhang for their guidance and support throughout the course of the project. Without their help and advice, the work presented here would not have been possible.

Thank also goes to Vanie for her tolerance and constant encouragement during my busy year. This report is dedicated to her.

Minh Duc, Cao

Contents

Figures

List of Tables

Chapter 1

Introduction

Document classification (aka *document categorisation* or *topic identification*), the task of assigning text documents to predefined categories, attracts much of research interest in recent years. This is largely due to the increasing availability of tremendous amount of documents in electrical format. Classifying documents from a collection into categories is helpful for people to retrieve useful information more effectively and more efficiently

Research on document classification dated back to early 1960s, and became more popular in 1990s, when study in *machine learning* became active. Many machine learning methods are applied to document classification and are reported to gain an accuracy comparable to human experts while no human intervention is required [38]. These methods generally automatically train a classifier from a set of pre-classified documents, which are different from *knowledge engineering* methods used in 1980s, where expert knowledge is required to define a set of classification rules.

The set of pre-classified documents used to build the classifier is called training set. During training process, a set of discriminative characteristics, often from the document text, are generated from the training set and encoded into the classifier to produce classification rules. In research setting, another set of documents called test set, also pre-classified, is used to evaluate the performance of the classifier. Each test document is fed to the classifier, the class assigned by the classifier is compared with the true label of the test document. The rate of matches is the measurement of the accuracy of the classifier.

1.1 Issues & Motivations

Scientists spend much of their time reading papers related to their fields of research. With the increasing volume of scientific literature online nowadays, classification of scientific documents is becoming more and more important in research communities. A system of automatic paper classification and recommendation would be very helpful for scientists in organising and collecting their paper database. Conference organisers, librarians and journal publishers would find such a system considerably useful in dealing with an enormous number of papers submitted and published. It is obvious that, manual categorisation of those documents is very time consuming and requires much of human effort. Thus, it is essential to develop a scientific paper classification system that can work automatically and adaptively.

Several scientific paper search engines such as Google Scholar[2] and CiteSeer[1] provide very good tools for researchers for searching for scientific papers. However, searching of papers on those engines are primarily based on keyword matching and therefore, a search would often result in a large number of hits, majority of which may not be relevant to what users look for. A suggested solution to improve the search results is to categorise indexed documents into predefined topics. Users can select the topics they are interested in when performing a paper search. The search engines need to search for documents in selected topics only. That mechanism not only reduces searching time as only documents in those topics are searched but also produces more accurate hit list. Therefore, a scientific paper classification system would be a significant improvement for those scientific document search engines.

Literature has shown many machine learning approaches for general document classification. The most common methods are naive Bayes [21], k-nearest neighbours [42], nearest centroid [13], decision trees [22], support vector machines [16], maximum entropy [29] and neural networks [36]. However, their performances are reported differently, partly because each method performs differently in different application domains. To build an effective scientific document classification system, these methods need to be examined in order to find a suitable method for classifying scientific documents.

Generally, a classification method needs to extract discriminative features from documents to train a classifier. In document classification, features are normally single words, which can be extracted easily from documents. Nevertheless, in English many phrases represent meaning, especially many scientific terminologies are in compound words. It is suggested that a number of phrases could be good features to identify topic of papers. However, extracting phrases is hard as computers cannot differentiate a phrase and a number of consecutive words in documents. Furthermore, if any blocks of consecutive words are considered as phrases, then not only the number of features is overwhelming but also noise is introduced.

Scientific papers, different from general documents, do not exist in isolation but are linked together by a citation network. A paper normally cites other related published papers which are likely to have similar topics. In other words, the citations link similar papers together. Hence, besides information from document content, the citation structure provides another source of clue that could be exploited for a better classification. If we could combine the document contents information and the citation information, it is expected that a better classification accuracy can be obtained in comparison with using content information only.

1.2 Goals

This research project aims to develop a domain-independent and adaptive approach to scientific document classification, primarily along the way of using one of the best existing classification methods as well as investigating ways to improve the system performance. A number of commonly used machine learning methods for document classification such as k-nearest neighbours, nearest centroid, naive Bayes and decision trees will be implemented and embedded into the system. To examine the performance of this system, a real world scientific paper collection will be chosen as a test bed.

The specific goals of this research project are to:

- Evaluate a number of common document classification methods to find out which one is the most suitable for scientific documents.

- Examine the discriminative ability of phrases for document classification. Particularly, investigate an effective method to extract only useful phrases to avoid intractability and to gain good classification performance.

- Investigate the effectiveness of citation structure in scientific document classification and develop a system so that citation information can be combined with document contents. The combination system is expected to perform better than that of using only content information for classification.

1.3 Contributions

The project makes two major contributions in the field of information retrieval in general and document classification in specific:

1. The project shows that phrases from scientific document can be extracted and used as features in addition to single words to improve a classification system. It also presents a method to select good phrases using information gain.
2. The project shows that citation links can be combined with document contents for better classification of scientific documents. We develop two methods, LLU and PLU, to demonstrate the usefulness of citation links information.

Part of the project has resulted in a paper [5] to be presented in the Australian Joint Conference on Artificial Intelligence, Sydney 2005 and to be published in the Lecture notes in Artificial Intelligence by Springer.

1.4 Report Structure

The rest of the report is organised as followed.

- Chapter 2 introduces the background of current studies in machine learning and its application to classification problem. The chapter presents a review of document classification and current issues in the field.
- Chapter 3 describes the toolkit used for development and the notation used in this report. It also describes the dataset used in this project and some preliminary preprocessing of data.
- Chapter 4 shows our investigation of a number of content-based classification strategies on a real world scientific document corpus. The chapter also includes the investigation of using phrases as features.
- Chapter 5 presents the our work on citation link structure exploitation for classification. The two developed methods, *linear labellings update* and *probabilistic labellings update* are presented.
- Chapter 6 summarises our work and describes areas for future research.

Chapter 2

Background

This chapter overviews the current research in the field of document classification, particularly machine learning approaches. Section 2.1 briefly introduces the main concepts of machine learning and pattern classification. Section 2.2 surveys work on general document classification. Section 2.3 reviews works on document link mining for information retrieval. Finally, section 2.4 summaries the chapter and discusses several issues in the field of document classification.

2.1 Machine Learning and Classification

2.1.1 Machine Learning Overview

Machine learning is the study of designing of computer programs that can automatically improve themselves with experience. More specifically, it concerns about the ability of machines to adjust their structures and/or parameters according to inputs or perceptions from environment in order to achieve the best results for later computation. Machine learning has been successfully applied to many applications such as fraud detection, speech recognition, hand-written recognition, driving autonomous vehicles and games [28].

The necessity of machine learning arises from many real world problems that good solutions might not be directly designed. For instance, the underlying principle of many tasks are hidden or too complex and thus are not easy for human to encode into programs. However, if some examples of the tasks are available, a machine learning system would be able to discover the hidden underlying principle or build an approximate solution that can solve the problems at a satisfactory level.

Furthermore, as environment changes overtime, a static program may need to be redesigned in order to solve the tasks. An adaptive machine would be desirable to reduce human intervention. Machine learning is also often used in the area of data mining, where the hidden patterns need to be drawn from a large amount of data.

There are three major types of machine learning. In *supervised learning*, teachers are available to give feedback to the learners. For example, for each given input, the corresponding "real" output is given. The learner compares its output with the real output then adjusts itself accordingly. In *unsupervised learning* there is no explicit teacher, i.e. a set of input is available but no specific output values are available. The learner has to generate the natural patterns from the input. Such examples of unsupervised learning are automatic clustering or image compression. The third type of learning is *reinforcement learning*, in which the learner receives little or indirect feedback from the teacher. The learner has to learn from reinforcement signal instead of being told by the teacher the exact actions what to do [37].

2.1.2 Pattern classification

Pattern classification [35] is a subfield of machine learning. It concerns with the problem of mapping data into predefined groups or classes based on data attributes. Pattern classification generally is supervised machine learning. A set of example, called the *training set*, is manually labelled and used to train the classifier. The training process gleans discriminative characteristics of training examples in each category and encodes them into the classifier. The trained classifier is evaluated on another labelled set of data called the *test set*. The output of the classifier on each test sample is compared with its true label to determine the accuracy of the classifier.

Prior to learning and classifying, a set of features need to be selected. Features are attributes of data that could distinguish their classes. The feature values of each object are then extracted to form a *feature vector*, which represents the object. The learner takes feature vectors of training examples as input and the labelled output as feedback to train the classifier. The trained classifier then takes feature vectors of unknown objects to compute their classes.

2.2 Document Classification

Document classification is the process of analysing a corpus of documents and categorising them into predefined classes reflecting contents. The categorisation of documents would be helpful for human for better information retrieval, management and access.

The studies of document classification started in early 1960s when Huhn [23] and Borko and Bernick [3] proposed statistical approaches for document topic identification. Their methods extracted keywords from documents and classified them by analysing those keywords against a *controlled dictionary*. While these methods gained some level of automation, some human effort and domain knowledge were still required for the construction of the controlled dictionary and keywords specification.

During 1980s, most of work in the field concentrated on document categorisation based on manual crafted decision rules or expert systems [10, 14]. Again, in those methods, expert knowledge of each category is required and decision rules need to be manually designed. Furthermore, they are not adaptive as when categories catalogue changes, expert must intervene again.

From the early 1990s, most of research on document classification turned to machine learning which was reported to gain an accuracy comparable or some time better than manual classification while no human intervention was required [38]. A machine learning approach for document classification involved in features selection, document representation by *feature vectors* and classification algorithms. This section reviews the most important aspects of machine learning approaches. From here on, the term document classification refers to document classification using machine learning approaches.

2.2.1 Feature selection and document representation

Document classification, like pattern classification described in section 2.1, is supervised machine learning. A set of manually labelled training examples are used to train a classifier. The classifier reads a document in the feature vector forms.

To represent a document as a feature vector, feature selection and feature extraction need to be done. This is to select discriminative attributes and extract them to form a feature vector. The machine learning approaches for document classification generally use single words as features. An unique word used anywhere in the corpus is corresponding to a feature.

A feature vector of a document is a vector of weights $d_j = \langle w_{j1}, w_{j2} ... w_{j|\tau|} \rangle$ where w_{ji} is the weight of feature i in document j and τ is the set of all features. There are several models to extract features and form feature vectors. The most commonly used are *bag of words* and *set of words* models. In bag of words model, the weight w_{ji} is the number of occurrences of word i in document j, while in set of words model, w_{ji} is the boolean value representing whether the the word i is in document j or not. The two representations, even being simple, have been reported as good as other more complex representations [9].

As any unique words in the corpus are features, the number of features in a classification system is large and thus the feature space dimensionality is high. To reduce the feature space dimensionality, several *feature selection* techniques are proposed to select only good features for classification. Generally, the "goodness" of each word for classification is examined and only "good" words are selected to be features. The measurements of "goodness" of words are based on some statistics drawn from the training set. The simplest measurement is *document frequency*, which considers the number of documents the word occurs in as the strength of the word. The method, as reported by Yang and Pedersen [45] could reduce the dimensionality by a factor of 10 without losing any accuracy.

The document frequency selection method, however, may consider commonly used words as features even though they do not help to distinguish the document topic. For example, words like "abstract" or "email" appear in many scientific papers, but they do not tell anything about the topic of the papers. Instead of counting only the presence of words in a document, the *information gain* feature selection method considers both the absence and presence of a word in a document. The method, as expected, is reported to perform better than document frequency. In fact, the performance of a classification system slightly improve when only 5% of words are selected by information gain [45]. Several other feature selection methods such as *chi-square* (*chi*^{2}) and *mutual information* are also shown to have a comparable performance with information gain.

2.2.2 Classification techniques

The most important component of a document classification system is the classification techniques. This section presents a number of the most common methods used in the field. These methods are naive Bayes, instance-based and decision trees classification.

Naive Bayes classification

The naive Bayes classification computes the probability that a document with feature vector d_j belongs to class c_i using Bayesian theorem as described by equation 2.1.

$$P(c_i|d_j) = \frac{P(d_j|c_i)P(c_i)}{P(d_j)} \tag{2.1}$$

To compute the probability $P(d_j|c_i)$, the naive Bayes assumes that all features in a document are statistically independent (that is why the method is called "naive" Bayes). With the independence assumption, the equation can be rewritten as

$$P(c_i|d_j) \propto P(c_i) \prod_{k}^{|\tau|} P(w_{jk}|c_i) \tag{2.2}$$

Section 4.2.3 describes the naive Bayes classification technique in a greater detail.

From the training set, the classifier only needs to learn the probability $P(w_k|c_i)$ for each feature k. There are several *event models* to learn these probabilities from the training set [21]. The *binary independence model* or *multivariate Bernoulli model* considers the existence of a feature w_k in a document as boolean value 1 and the absence as 0. In other words, this model is suitable for *set of words* representation of documents. With this presentation, the model loses the information about word frequency in documents.

On the other hand, the *multinomial* model captures the number of occurrences and thus obtains more information for classification. Experiments from McCallum and Nigam [25] confirmed that the multinomial model produces better classification than the binary independence model, especially when the number of features is large.

Instance-based classification

In instance-based classification, the classifier simply stores the presented training data instead of constructing a general, explicit target function. When a document needs to be classified, the classifier examines its relationship with the stored training data to determine its category. Therefore, instance-based classifications are normally called *lazy learners* as they postpone processing until the category of an unknown document is queried.

Generally, the instance-based classification defines a function to compute the similarity between any two documents. The function to decide the similarity of two feature vectors could be Euclidean distance [33], cosine distance [44] or Manhattan distance.

The most simplest example of instance-based is nearest neighbour where an unknown document is classified to the same class as the most similar document from the training set. When the number of training examples is large, it makes more sense to select more than one most similar documents from the training set for a better classification. That is the idea behind the k nearest neighbours method [41, 43].

Another method in the class of instance-based classification is nearest centroid developed by Han and Karypis [13]. Rather than remembering all training examples, the classifier represents each category by a vector, which normally is the mean of feature vectors of all documents in that category. An unknown document is classified to the category whose representing vector is the most similar to the input document.

2.2.3 Decision tree classication

Document classification using decision tree has been developed by Lewis and Ringuette [22]. Decision tree classifier uses a tree structure to decide the class of a document. Each node of the tree is associated with a feature and the leaf nodes are labelled with classes. A document is classified by recursively traverse the tree to a leaf node and the category is assigned based on the label of the leaf node reached. A more detailed description of decision trees is presented in section 4.2.4.

2.2.4 Other classification techniques

A number of other classification methods are also proposed in document categorisation. The support vector machines algorithm [16, 39] focuses on finding the hyperplane that maximises the margin between classes. The method has the advantage of being able to handle large number of dimensions, which is general the case for document classification.

Ruiz and Srinivasan applied neural networks for document classification [36]. They developed a back-propagation neural network and a counter propagation neural network [15] for classification. They reported that the back-propagation neural network generally performed better.

2.2.5 Comparison of classification techniques

There have been a number of studies on comparison of classification techniques. However, there are inconsistencies on the comparison of classification techniques reported from many studies. This is largely due to the fact that different studies use different dataset for the comparison.

Yang and Liu [44] performed experiments on Reuter dataset, one of the standard benchmarks often used, and reported that k nearest neighbours and support vector machines methods are among the best classification methods. They outperformed other methods such that naive Bayes and neural networks.

Han and Karypis [13] used 23 different datasets for comparison among naive Bayes, k nearest neighbours, decision trees and nearest centroid. Their research shows that, naive Bayesian outperformed the other schemes in five out of the 23 datasets, decision trees did better in one, the nearest centroid scheme did better in 17, whereas the k nearest neighbour algorithm never outperformed the other schemes. Another study by Lewis and Ringuette [22] on comparison between decision trees and naive Bayes reported that the two classification methods were comparable.

The disparity between findings could possibly due to the fact that a classification algorithm may work differently on different datasets. The performance of a classification technique could depend on the characteristics of documents in the collection such as document lengths, variety of words used etc and thus a method may work well on some dataset but not on the others. Therefore, given a document corpus to be classified, a classification algorithm must be carefully chosen to obtain the best results.

2.3 Document Links Mining

Scientific documents are linked together by a citation network. The connectivity nature of documents hints that, there is extra information rather than document contents that can be used for classification. The integration of linkage information and content information are expected to have a better classification than using contents only.

Link analysis has been researched intensively since the birth of the world wide web. Brin and Page exploited hypertext mining for PageRank, the technology behind the success of Google [4]. Their method iteratively compute the "importance" of web pages to rank the relevance of web pages which is used to prioritise search results. The anchor texts, i.e. texts of the links are taken into consideration as they well describe the pages the links lead to. Also, the rankings of pages are propagated through links in the PageRank system.

Chakrabarti et al. [6] explored the combining of words from connected documents into the feature vector of a document for classification. Their work showed that the naive use of text form neighbouring documents even degrades accuracy of the classification. Their explanation for the decrease is that link information is noisy and the distribution of terms from neighbouring document is not sufficiently similar to the distribution of the "true" class.

Instead of using contents from neighbouring documents, they incorporated the category information such as some initial guesses of categories. They then applied *Markov random field* model to iteratively update the category of documents. To make calculation manageable, they limited range of influence to 1 or 2 radius, which were equivalent to first and second order Markov random field. The approach is shown to improve the classification accuracy.

Another study by Oh et al. [30] reported similar finding: naive incorporating contents from neighbouring documents is generally not helpful while predicted class information is helpful. Their algorithm for hyperlink mining limits the influence of neighbouring documents to only *trustable* documents, which are documents similar enough to the target document.

Recently, work from Getoor et al. [11] applied the relational model [40] for link mining and Craven and Slattery [8] proposed combining statistical text analysis with a relational learner such as FOIL [34].

The connectivity nature of scientific papers forms a graph in which vertexes are papers and edges are citations. In recent years, many researchers focus on *graphical models*, a field combining graph theory and probability theory for a model representing the interaction between objects [18]. The model can represent the dependency among objects in the domain. The Bayesian framework is applied in the model for probabilistic inference of properties of objects. The important characteristic of the graphical model is that, a model can be learned from some available data. Either the graph structure (e.i. the patterns of edges) or graph parameters (e.i. quantitative dependency between objects) can be constructed by graphical learning.

2.4 Summary

This chapter presents an overview of machine learning and pattern classification and presents a survey of the field of document classification. It is noticed that,the performance of each classification method depends on the nature of the corpus. In order to have the best results for scientific document classification, a comparison of classification algorithms needs to be performed on a real world scientific paper collection.

Study on links mining in web pages recently reported promising results from combining document contents and hypertext for classification. It is expected that scientific documents, which share the connectivity property, could achieve a similar improvement for classification.

The study of links mining is relatively new and is an active area of research. Many techniques from machine learning such as graphical model are mature to be exploited for links mining, making the area an interesting and promising for research.

Chapter 3

Notations, Preliminary Processing and Dataset

This chapter presents an overview of some aspects of our development system and the dataset used in our experiments. Section 3.1 presents the notations used throughout this report and the toolkit used in our development. Section 3.2 shows some preliminary pre-processing of data prior to classification. Section 3.3 describes the dataset used in the experiments.

3.1 Notations and Toolkit

The problem of document classification can be stated formally as follows. Given a corpus of documents (representing in feature vectors) $\mathcal{D} = \{d_1, d_2, ..., d_n\}$ and a set of predefined categories or classes $C = \{c_1, c_2, ...c_m\}$, assign each of document d_j to one of the classes c_i.

More generally, a classifier is a function ϕ from document domain $\mathcal{D}$ to R^n where $\phi(d_j) = (s_{j1}, s_{j1}, .., s_{jn})$ and s_{ji} is the category score of document d_j assigning to class c_i. The document d_j will be assigned to the category which has the highest score:

$$\mathcal{C}(d_j) = arg\ max_i\{s_{ji}\} \tag{3.1}$$

The category score vector $V_j = (s_{j1}, s_{j1}, .., s_{jn})$ output from function ϕ, $V_j = \phi(d_j)$ is called the labelling of document j by function ϕ or by the corresponding classifier. If a document class is known, i.e. document from training set, we also define the labelling of the document as the vector $V_j = (s_{j1}, s_{j1}, .., s_{jn})$ where $s_{ji} = 1.0$ if the document is in class i, otherwise $s_{ji} = 0.0$.

The classifier is trained by a set of training examples S. A good training set should contain examples from all categories, so that the characteristics of all categories could be extracted by the classifier trainer. The training set S could be divided into disjoint sets S_i, which contains the set of training documents for class i.

$$\begin{aligned} S &= \bigcup_i S_i \\ \forall i \neq j, S_i \cap S_j &= \emptyset \end{aligned} \tag{3.2}$$

The project is carried out using Mallet toolkit [26] developed by McCallum from Carnergie Mellon University. The toolkit provides a framework for plug-and-play classification methods and some preprocessing of documents. Several classification techniques such as naive Bayes and decision trees are included in the toolkit. However, the toolkit is incomplete and under construction. Many functionalities are added into the toolkit during the development of the the project.

3.2 Preliminary Processing

Before documents are used they need to be pre-processed to convert into a format suitable for the trainer and the classifier. The pre-processing performed in a number of steps including *tokenisation*, *removal of stop words* and *stemming*. Tokenisation is to divide the document into word units. As the number of unique words taken from a corpus is large, removal of stop words and stemming are used to remove non-informative words.

3.2.1 Tokenisation

As the information from contents of documents for classification are taken from words, each document needs to be divided into word units. The process is called tokenisation. Essentially, documents are first read as a sequence of characters. Any non-alphabet characters (spaces, punctuation, numbers etc) are not part of the words and thus are considered as separators.

As most of scientific papers collected are parsed from PDF or PS format, there is noise resulting from parsing. For examples, parsing images or formulae produce meaningless text, most of them are one or two characters long. Most of English words are longer than two characters and words having length of two or less are normally either noise or topic-neutral and are not helpful for classification. Therefore, only words longer than two characters are retained.

3.2.2 Stop words removal

In documents, stop words are topic-neutral words such as *this*, *that*, *our*, *they* as they do not contain information for classification. They are contained in almost all documents and knowing their absence or presence in a document does not help to identify the topic of the document. Thus it is desirable not to include them in the feature list.

Apart from a standard stop word list included in the Mallet toolkit, we identify a number of stop words in scientific document domain. Such words like "email", "abstract" and "department" appear very frequently in scientific papers but do not help for topic identification. Any tokens of the document after tokenisation found in the stop word list are removed from the document presentation. The complete list of stop words used in this project is presented in Appendix A.

3.2.3 Stemming

Stemming refers to collapsing words having the same root, for example, words like *report*, *reporting* and *reported* are considered as one word. At this stage, terms remained after previous steps are passed on to a Porter stemming [32] which uses a suffix tripping technique to unify words with the same root.

3.3 Dataset

For experiments throughout this study, we use Cora, a real world scientific paper corpus collected by McCallum et al [27]. A subset of Cora collection is selected as a test bed for the project.

The test bed contains 4330 papers in seven subjects of machine learning: case based, probabilistic methods, learning theory, genetic algorithms, reinforcement learning, neural networks and rule learning. The proportion of papers in each topic ranges from 7% in rule learning topic to 32% in neural network topic. After removal of stop-words and stemming, the data set contain about 61000 words.

These papers are connected by a network of 12260 citation links. Examine the citation network, we find that 65% of citations are between by papers of same class, while 35% of citations are cross topic citation.

For each experiment, we randomly select an equal portion of the papers from each of the seven categories for training a classifier, and use the rest as the test set to evaluate the classifier. We carry out experiments on different training set sizes, which are 20%, 30%, 40%, 50% and 60% of the collection to examine the effect of each method. For the ease of notation, we name each configuration of data set as CORAXX where XX is the percentage of training set. For example, the set CORA40 refers to using 40% of the collection as training set and 60% as test set.

Each configuration of the system is tested 10 times on different set of training set and test set. As an example, when experiment with training set of size 20%, on each run, the system select randomly 20% of each class to make up the training set. The size of training set on 10 runs are the same but the documents selected are different. The mean accuracy of the 10 runs are recorded and reported.

Chapter 4

Content-Based Classification Methods

4.1 Introduction

This chapter aims to investigate the first two goals of the project, which are the evaluation of some common content-based classification methods for scientific documents, and exploration of the use of phrases as features for classification. The classification methods we investigated are k-nearest neighbours, nearest centroid, naive Bayes and decision trees.

For classification using information from document contents, each document j is represented by a feature vector $d_j = \langle w_{j1}, w_{j2} ... w_{j|\tau|} \rangle$ where τ is a set of features and w_{ji} is the weight of feature i in document d_j. Following the discussion of document representation models in section 2.2, we use *bag of words* model as it is expected to be better than *set of words* model, especially it allows us to use *multinomial* event model for naive Bayes classification method. In this model, each unique word in the dictionary corresponds to a feature and the weight w_{ji} reflects the number of occurrences of word i in document d_j.

The chapter is organised as follows. The four classification algorithms are described in Section 4.2. Section 4.3 shows our investigation of using phrases as features. Section 4.4 shows results from our work on content-based classification and Section 4.5 concludes this chapter.

4.2 Classification Methods

4.2.1 K Nearest Neighbours Method

The k nearest neighbours (kNN) method [41] is an example of instance-based classification strategy. The idea behind the kNN method is relatively simple. A similarity measurement between two documents is defined. The classifier simply "remembers" feature vectors of all documents in the training set. To rank categories of an unknown document d_j, the classifier selects *k* (a predefined number) most similar (or nearest) documents to the input document, and compute the weight of each category based on the labellings of selected documents.

The two important configurations for kNN are choices of the similarity measurement and ways to compute the labelling of the input document from labellings of selected documents. The similarity measurement used in this project is *cosine distance* where the similarity score between two documents d_x and d_y is computed as:

$$Sim(d_x, d_y) = \frac{\sum_i^{|\tau|} w_{xi} w_{yi}}{\sqrt{\sum_i^{|\tau|} w_{xi}^2} \sqrt{\sum_i^{|\tau|} w_{yi}^2}} \tag{4.1}$$

The higher the score, the more similar two documents are. Therefore, the k nearest neighbours of a document are those having k highest similarity scores.

The function to compute the score of an unknown document j assigning to class i is the sum of scores of all nearest neighbours that are from class i. Formally, the elements of the labelling of a document j is

$$s_{ji} = \sum_{d_t \in S_i} Sim(d_j, d_t) \tag{4.2}$$

where d_t is one of the nearest neighbours of d_j.

There are several problems of the kNN methods. The first problem is that, classification accuracy is badly affected by noise from the training set. The class of an unknown document is determined by the set of its nearest neighbours only without any global information for the true class. If some of its nearest neighbours are noisy, the document would has a high chance to be misclassified. It is desirable that noise is eliminated from the training set. The second problem of kNN lies in the complexity of classification. For each input document, the classifier has to compute its similarities to all of the training examples. A smaller training set would make the classification faster.

To improve kNN classification, this project applies *noisy pruning* [41], a process to remove noise as well as reduce the number of training examples. The documents in training set that tend to misclassify will be removed. The classifier is used to classify all documents in training set and the performance of each training example is recorded. This is done by keeping track of the number of correct and incorrect decisions each training example makes. If the ratio of incorrect decisions of a training example is over a threshold, the training example is considered as noise and is removed from training set.

4.2.2 Nearest Centroid Method

The nearest centroid (NC) classification method [13] is another instance-based strategy and is quite similar to the kNN method described above. The difference is that, for each category, the classifier only needs to "remember" a *centroid* vector representing that category. The centroid vector of a category is the summarisation of characteristics of that category drawn from training set. To compute how close a document is to a category, the classifier needs only to compute the similarity between the document and the centroid vector of that category. The centroid vector of each category is computed as the mean of feature vectors of all documents belonging to that category in the training set.

$$\overline{c}_i = \frac{1}{|S_i|} \sum_{d_k \in S_i} d_k \tag{4.3}$$

where S_i is the set of training documents in class c_i and $\overline{c}_i$ is the centroid of that class.

There will be one centroid vector for each document category. The NC also defines a similarity measurement of a document to a centroid, which is similar to that of kNN. The classifier ranks category scores of each unknown document by the similarity scores of that document to the centroids of the classes/categories.

$$s_{ji} = Sim(d_j, \overline{c}_i) \tag{4.4}$$

Comparing with kNN method, the computational complexity of NC is much better on classifying. The kNN classifier has to compute the similarities of each unknown document to all training documents while the NC method needs to compute the similarities to the centroids only.

4.2.3 Naive Bayes Method

The naive Bayes (NB) classification has gained popularity in document classification due to its computational efficiency and competitive classification accuracy. The framework of NB classification method bases on *Bayes theorem*, a well-known formula in probability theory. The score of document d_j in class c_i is defined as the probability of a document in class c_i given the document representation d_j, which is estimated as:

$$s_{ji} = P(c_i|d_j) = \frac{P(c_i)P(d_j|c_i)}{P(d_j)} \tag{4.5}$$

where the event $P(c_i)$ is the short notation for $P(\mathcal{C}(d_j) = c_i)$. $P(c_i)$ is the probability that a randomly picked document in class c_i and $P(d_j)$ is the probability of a document having representation as feature vector d_j. The probability $P(c_i)$ is called prior probability as it is the degree of belief about any document without any other information. $P(d_j|c_i)$ is the conditional probability representing the probability a document in class c_i represented by vector d_j.

As the classifier assigns the document d_j to the class i which has the highest $P(c_i|d_j)$ and all $P(c_i|d_j)$ with $i = 1..m$ share the common denominator $P(d_j)$, we can omit the term $P(d_j)$, which is generally referred to as the *normalisation factor*.

$$s_{ji} = P(c_i|d_j) = \frac{P(c_i)P(d_j|c_i)}{P(d_j)} \propto P(c_i)P(d_j|c_i) \tag{4.6}$$

One problem of the equation 4.6 is that, there are too many possible vectors d_j and thus the estimations of $P(d_j|c_i)$ is problematic. To alleviate the problem, it is normally assumed that in a document, each word is statistically independent to each other. The assumption is normally referred as the *independence assumption* or the *Naive Bayes assumption*. That is the reason why classification using this assumption is called naive Bayes classification.

To apply the multinomial model [25], we make two naive Bayes assumptions. These are that the lengths of documents are independent of class and each word event is independent of its context and position in the document. Define N_{jk} to be the number of occurrences of word w_k in document d_j and $|d_j|$ is the length of document d_j, the multinomial distribution of a document given its class is:

$$P(d_j|c_i) = P(|d_j|)|d_j|! \prod_{k=1}^{|\tau|} \frac{P(w_k|c_i)^{N_{jk}}}{N_{ik}!} \tag{4.7}$$

The independence assumptions allow us to compute $P(w_k|c_i)$ separately. During training phase, $P(w_k|c_i)$ are estimated as the counting of the occurrence of word k in class i. To prevent zero probability from happening, the *Laplace smoothing* is employed by adding one into the counting.

$$P(w_k|c_i) = \frac{1 + \sum_j^{|S_i|} N_{jk}}{|C| + \sum_l^{|C|} \sum_j^{|S_l|} N_{jk}} \tag{4.8}$$

The prior probabilities $P(c_i)$ are estimated using maximum likelihood estimation:

$$P(c_i) = \frac{|S_i|}{|S|} \tag{4.9}$$

The categories scores of each input document are computed by equation 4.6, substituting the relevant parameters by equations 4.9, 4.8 and 4.7.

4.2.4 Decision Trees Method

A text decision tree (DT) classifier [28] is a tree in which internal nodes are labelled by features and leaves are labelled by categories information. The decision tree is trained and constructed by selected features from the training set that can decide the category of a document. The common technique of measuring the influence of a feature on the classification of a document uses information gain, an entropy-based measurement [28]. The weight of those features in the feature vector of a document will recursively direct the document to traverse the tree until a leaf is reached. The labelling associated with that leaf will be assigned to the document.

The information gain of a feature measures the amount of information obtained for classification by knowing the presence or absence of that feature in a document [45]. The information gain of a word w_i from a document set S is defined as:

$$\begin{aligned} IG(w) = -\sum_{i=1}^{|m|} P(c_i) log P(c_i) + \\ P(w) \sum_{i=1}^{m} P(c_i|w) log P(c_i|w) + \\ P(\neg w) \sum_{i=1}^{m} P(c_i|\neg w) log P(c_i|\neg w) \end{aligned} \tag{4.10}$$

where:

- $P(c_i), i = 1..m$ is the probability that a document is in category c_i. $P(c_i)$ is defined as the proportion of set S in category c_i.
- $P(w)$ is the probability of a document containing feature w. Likewise, $P(\neg w)$ is the probability that a document does not contain feature w.
- $P(c_i|w)$ is the conditional probability that a document is in category c_i given that it contains contain feature w. Similarly $P(c_i|\neg w)$ is the conditional probability that a document is in category c_i given that it does not contains contain feature w.

The decision tree classifier is trained by a divide-and-conquer technique. At the root of the tree, the feature w_1 of highest information gain on training set is selected. The training set is partitioned into two sets such that in each set, all documents either contain or do not contain feature w_1. The two sets are associated with two sub-nodes of the root node. The process is recursively performed in each sub-node until all documents in a node are the same category or the tree reaches a maximum depth. The labelling attached to each node reflects the distributions of categories in that node.

4.3 Using Phrases as Features

The use of phrases as features in text documents has been described as "not uniformly encouraging" by Sebastiani[38]. Lewis [20] explained that, although phrases have superior semantic quality, they have inferior statistical quality.

Nevertheless, it is observable that, many scientific terminologies are in compound words, and they are sources of information to identify topics of papers. It is expected that using phrases in addition to words could result in a better classification system. The second goal of this project is to explore the effectiveness of using phrases features for classification.

One of the difficulties of the construction of phrases is that, computers do not understand the semantic of words and phrases unless a dictionary is built in. Embedding a dictionary of phrases into the system would require expert efforts as phrases for different fields are different.

One possible solution of the construction is that, phrases are constructed by any two consecutive words (bigram) and added into the term set (term here refers to both word and phrase). The resulting set of terms, undoubtedly, contains very large number of features, but majority of them would be noise as the construction does not consider "real" compound words. Using all of them as features does not only requires too much computation power but also results in weak performance [38].

However, instead of selecting meaningful phrases to use as features, the classification system can select phrases that are discriminative. That is, the system does not need to know whether a term (word or phrase) is meaningful or not. As long as a term can help to identify the topic of documents, it is selected. It is desirable that the list of useful phrases can be obtained by examining training set.

Fortunately, in the field of document classification, there are techniques for *feature selection* or *dimensionality reduction* for selecting useful terms. The general idea of feature selection is to remove non-informative terms and retain discriminative terms. Removal of stop words is a trivial example of feature selection where the list of non-informative words are static and built based on some heuristic prior to training of classifier.

An adaptive approach to feature selection generally applies some statistical measurement to determine the "goodness" of terms. Examples of those statistical measurements are frequency thresholding, information gain, mutual information, χ^2 statistic and term strength. Work from Yang and Pedersen [45] showed that, several feature selection methods can dramatically reduce feature space dimensionality and thus improve training and classifying time, but do not hurt the performance.

Examining some previous works on comparison of feature selection methods, this project chooses information gain, which is reported as one of the best method for dimensionality reduction [45]. The method measures the "goodness" of term based on information gain, which is the same as one used in decision tree classification described above (equation 4.10). Terms with information gain less than a threshold are to be removed.

In summary, the approach used in this project for selecting phrases is that phrases are first constructed by any two consecutive words and added to the term list. The information gain of a term, either word or phrase, from the training set is computed as in equation 4.10. Only terms with high information gain are retained to convert to the feature vector.

4.4 Results and Analysis

4.4.1 Classification Methods

On the first experiment, we test the four classification methods using all words in the corpus as features. We perform each test on different configurations of datasets as described in section 3.3. Those dataset configurations are CORA20, CORA30, CORA40, CORA50 and CORA60 which respectively have 20%, 30%, 40%, 50% and 60% of the collection as training set.

The performances of those classification methods on five dataset configurations are shown on Table 4.1. Figure 4.1 shows the results in bar charts for comparison among those classification methods.

An interesting observation from the results shown is that, the performances of those methods depend on the amount of documents in the training set. When a small training set is used, the instance-based classification methods (kNN and NC) are superior to the others. On the other hand, if the training set is sufficiently large, the NB produces the best accuracy.

Classifiers	kNN	NC	NB	DT
CORA20	73.63%	71.26%	58.01%	65.40%
CORA30	75.77%	72.45%	69.86%	67.66%
CORA40	76.91%	72.63%	73.45%	71.08%
CORA50	77.60%	73.04%	79.45%	72.07%
CORA60	78.46%	73.84%	81.97%	72.56%

Table 4.1: Performance of classification methods.

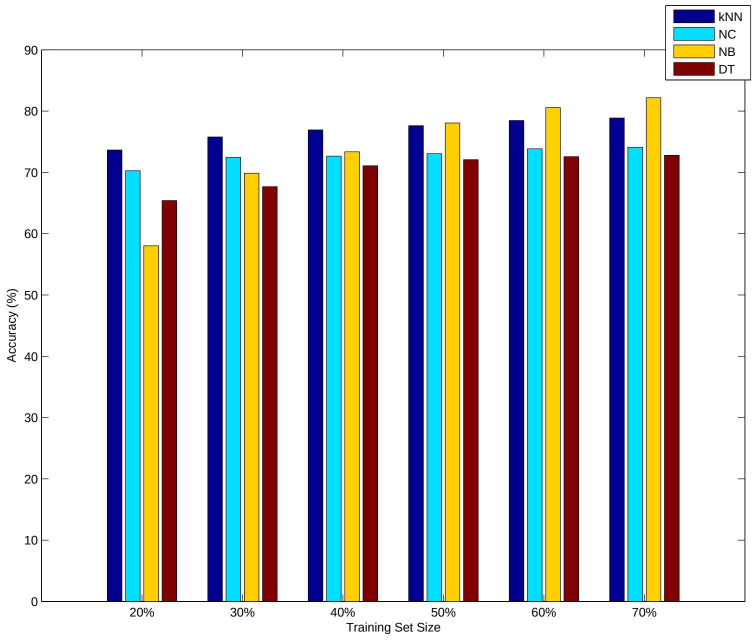

Figure 4.1: Comparison of classification methods

Specifically, when a subset of only 20% documents is chosen as the training set(CORA20), the kNN and NC can obtain accuracies of 73.63% and 71.26% respectively while NB and DT classifiers can only correctly classify 58.01% and 65.40% of test set. However, if about 2000 documents (in CORA50 configuration) are chosen for training, the NB classifier performs slightly better than kNN. It outperforms all other classification methods when a larger training set is used.

In general, the kNN is among the best classification methods. It can work well in cases of few training examples as well as many training examples. That is because in classification phase, the classifier needs a few known documents similar to the input document, and determines the category of the input document based on them. If training examples are uniformly distributed across the document space, a good accuracy is obtained.

The NC summarises the characteristics of each category in a centroid vector, which is the mean vector of feature vectors of all training documents in that category. Again, if a small but good training set is used, the centroids could be in good positions. However, if the training set is enlarged, especially by documents from the same distribution, the centroids do not change positions much. Therefore, not much classification improvement could be obtained by a larger training set. The results from the experiment are an illustration of the observations. The accuracy of a NC classifier trained by 60% documents in the corpus (about 2600 documents) is not much better than that trained by 20% of corpus.

The NB classification method is the best when trained by 60% of documents, but is the worst when only 20% of documents are used as training set. Apparently, in NB approach for document classification using bag of words model, if a feature does not appear in any of training documents, it is not considered for classification, i.e. $P(w_k|c_i)$ is not recorded by the classifier. Therefore, a large training set for NB is needed. The two instance-based classification methods discussed above, however, consider a feature not appeared in training set as having value 0, and thus can work well with small training set.

The last classification method, DT, does not seem to perform well in classification of scientific documents, despite being successfully used in many other classification applications. This could possibly because of the fact that, in document classification, the number of features is large. A decision tree encodes only a small number of features in decision nodes, and thus much of information from other features, often there are many of them, is ignored. That prevents DT from obtaining a good performance like the others.

4.4.2 Effectiveness of Using Phrases

This sub section presents the effectiveness of using phrases in addition to words as features for classification. As described previously, the project uses a bigram model to construct phrases from the words sequence. The "naive" construction of phrases from the Cora dataset results in a term set of 120000 terms, which is about 20 times bigger than the size of using only words.

In the first experiment on using phrases, we do not include the dimensionality reduction method. That mean all words and phrases are used as features. The results of the experiment are shown in Table 4.2. Comparing the results with that of Table 4.1, an interesting observation is drawn. The performance of all classification methods except DT degrade. The reason for the degradation of classification accuracy when more features used is that, the newly introduced features are noisy. Obviously any two words often cannot make up a valid phrase. The DT method, however, slightly improves. As described above, the DT does not use all features to construct the decision tree. Instead, it selects a number of high information gain features only. Obviously, the more features to select the better performance DT achieves.

Classifiers	kNN	NC	NB	DT
CORA20	71.82%	70.60%	56.21%	69.02
CORA30	73.03%	72.38%	67.70%	70.73
CORA40	75.13%	72.84%	71.03%	72.27
CORA50	76.17%	72.89%	76.24%	73.89
CORA60	77.13%	73.36%	79.85%	75.30

Table 4.2: Performance of using all phrases

In the second experiment on using phrases, we reduce the term set to the size of the original term set (words only). Only 5% the best information gain features are selected and used for classification. Obviously, terms removed might be invalid phrases or non-informative words. The performances of those classification methods with the selected features are presented in table 4.3. With this set of features, the accuracies of all classifiers are better than using words only. The DT however, performs similarly to using all phrases, due to the reason discussed above.

Classifiers	kNN	NC	NB	DT
CORA20	78.18%	76.93%	62.21%	69.32
CORA30	79.43%	77.65%	71.70%	70.70
CORA40	80.42%	78.02%	76.48%	72.50
CORA50	81.29%	78.32%	81.84%	73.34
CORA60	81.82%	78.36%	84.85%	75.37

Table 4.3: Performance of using 5% of terms (words + phrases)

The results from the experiment clearly indicate that phrases are helpful for document classification. Using a term set of phrases and words produces better classification than using a words only set of the same size. That proves the discriminative ability of selected compound phrases.

As using 5% of terms produces better performance for most classification methods, we further examine if a smaller set of only good features can further improve classification accuracy. A number of experiments on different ratios of features selected are carried out. We found that, in general those classification methods (not including DT) perform the best when only 0.20% - 0.30% of all terms are used. Table 4.4 presents their accuracies with 0.25% of terms selected. The results is interesting because the number of features is now 20 times smaller than the number of words, but the performance is much better.

Classifiers	kNN	NC	NB	DT
CORA20	79.38%	77.27%	62.31%	69.49
CORA30	80.00%	77.40%	71.90%	71.10
CORA40	81.44%	78.20%	76.48%	72.80
CORA50	81.95%	78.82%	82.04%	73.64
CORA60	82.02%	78.96%	84.65%	74.86

Table 4.4: Performance of using 0.25% of terms (words + phrases)

With adding of phrases to term set and using feature selection, the relative performance of those classification methods remains the same as using words only. In the dataset configuration of small training set, the kNN is still the best while NB is superior to others with large training set.

4.5 Summary

In this chapter, we describe our evaluation of several techniques for scientific document classification using information from document contents. We perform a comparison of a number of common classification methods including kNN, NC, NB and DT. The use of phrases as features is also investigated.

Our comparison of classification methods shows that, the NB method outperforms others when the training set is sufficiently large. However, it performs badly on small training set.

The kNN method is also among the best methods for scientific document classification. In the dataset configurations with large training set, it is inferior to only NB but outperforms the other two. It can produce the best classifier from a small training set.

From the comparison, we find that, it is practical to use kNN when the number of labelled documents is small. If the number of available training examples is large, kNN also produces good classification accuracy but the classification time would be very long as it has to compare each unknown document to every training example. In this case, NB is a good choice because not only it produces better classification accuracy, but also its running time is much shorter.

We designed a method to extract phrases from document contents. Phrases are constructed by bigram model that is, any two consecutive words form a phrase. We then apply information gain feature selection method to select the best discriminative terms. Although adding all phrases degrades classification performance, selecting only "good" terms helps all classification method to achieve better performance. We also find that selecting only 0.20%-0.30% of terms from phrases and words set generally gives the best results.

There is a interesting question arise from this work. Our work and other work on comparison of classification methods only compare them based on empirical experiments and find that a method may work well on a dataset but not on others. Given a document corpus, is that possible that we can analytically choose the best classification method without empirical experiment? In other words, whether or not we can identify the strengths and weaknesses of a classification method without experiments? The question would be interesting for future research.

Chapter 5

Improve Classification Accuracy using Citation Links

5.1 Introduction

In scientific papers domain, citations are clearly a rich source of information to identify topic of documents. There is a correlation of topics between two citing documents. It is observable that the topic of a paper is related to that of papers it links to. This feature suggests that, information in the citation structure could help the classification of scientific papers. This chapter describes our investigation of combination of citation links and document contents for scientific document classification.

As reported by [6, 30], integration of contents of connected document is not helpful. Instead of using text from neighbouring documents (i.e. documents having links to or from), this project explores the update of labelling of a document from the labellings of its neighbouring documents. If the surrounding documents' categories of a document are known, the class of that document can be derived from them. However, that is not always the case. The set of labelled documents in the corpus is limited as labelling of document is expensive. Instead of using "true" labelling, this project first applies one of the content-based classification method to classify all documents in the corpus as a starting point. The labellings are then updated to give a better classification result.

As described in section 3.1, the labelling of each document d_j by a classifier is a vector $V_j = (s_{j1}, s_{j2}, ..., s_{jm})$ in which s_{jk} is the likelihood of that paper in class c_k. We need to design an updating function $\mathcal{F}$ to update the labelling of a document from its neighbouring labellings. Define N_j be the set of documents that connect with document d_j, the general form of the updating function $\mathcal{F}$ is:

$$V_j = \mathcal{F}(V_j, V_{j1}, V_{j2}, ..., V_{j|N|}) \tag{5.1}$$

or

$$V_j = \mathcal{F}(V_j, V_k) \tag{5.2}$$

for updating from an individual neighbour document d_k.

As we want to combine the information from document contents and information from neighbouring documents, the feature vector of a document now is $d_j = (T_j, N_i)$ where T_j is the text information and N_j is the neighbouring information.

We developed two methods for updating labellings in this project. The first one *linear labelling update* described in Section 5.2, defines a linear function to combine labellings of neighbouring documents. Sections 5.3 presents the second method, *probabilistic labelling update* which is based on Bayesian networks to define the updating function. Section 5.6

shows results from those methods and presents our discussions. Section 5.8 summaries this chapter.

5.2 Linear Labelling Update

The rationale behind the linear labelling update (LLU) is that, the information to identify topic of paper comes from the document itself and its neighbouring documents. Therefore, the labelling of a document is the combination of its own labelling (output from a content-based classifier) and labellings of connected documents. The method assumes that, each neighbouring document has an equal influence on the document. The updating function therefore adds a fraction of those labellings into the document's labelling.

$$V_j = (V_j + \eta \sum_{k:d_k \rightarrow d_j} V_{jk}) \tag{5.3}$$

where the updating rate η is a parameter that reflects the influence of linked papers. The higher η the more influence of the categories of neighbouring documents on the target document.

The updating procedure is performed iteratively until the labellings of all documents in the corpus become stable, i.e. the change of labellings is small enough.

5.3 Probabilistic Labelling Update

There are several limitations of the LLU approach above. Firstly, the model is static; the inference is dependent on the parameter η. The system may work well in some parameter values but may fail in other. Deriving a good parameter value requires an empirical search and the parameter value found is unlikely to be optimal. It would be desirable if parameters can be learnt from the training set. Secondly, the model is rather "naive" and may not capture the dependency well.

The dependence among documents in the corpus is presented by citation links. The citation structure can be modelled as a directed graph in which vertexes are documents and edges are citation links. As a paper can only cite other papers already published, there should not be a circle in the graph. The graph is similar to Bayesian network or belief network [31]. Therefore, it is suggested that Bayesian network can be used to model citation link structure. We call the method derived the Baysian network probabilistic labelling update (PLU).

5.3.1 Modelling Citation Links by Bayesian Network

Bayesian network is a directed graph representing dependencies among random variables [12]. Each node in the graph represents a random variable. An edge from node X to node Y represents dependency in the form of conditional probability $P(X|Y)$. In this case, variable X is said to be a parent of variable Y. Each node is associated with a conditional probability distribution $P(X|parents(X)$ that represents the effects of the parents on the node.

A Bayesian network allows calculation of the probability of some node given that some others nodes are observed. The well-known algorithm for exact inference in Bayesian network is *probability propagation* or sometime called *sum-product algorithm* or *message passing* [31, 18, 46, 7]. The idea behind the algorithm is that, belief is passed across the network to update probabilities of unobserved nodes.

If the graph is a poly-tree i.e. between any two nodes, there is at most one undirected path, the complexity of exact inference in Bayesian network is linear to the size of the network. However, in most cases, the network is multiply-connected, inference is NP-hard [37]. If that is the case, several tractable approximate inference methods could be used. Messages are still "sent anyway" between any two nodes which introduces some imprecision or Monte Carlo method [24] is used to generate samples from the distributions in the network.

From the above observation, this project models the link citation by a Bayesian network. An example of the network is shown in Figure 5.1. There are two types of nodes in the network. Each round node represents a document, whose category is directly influenced by its text and other neighbouring documents. A square node represents the text of a document. Parents node of a document are its text node and other documents that is cited by the document.

For the example, in Figure 5.1, document d_3 cites documents d_1 and d_2. Therefore, its parents are its text node T_3 and two document nodes d_1 and d_2. Document d_3 in turn, is the parent of documents d_4 and d_5 as it is cited by those two.

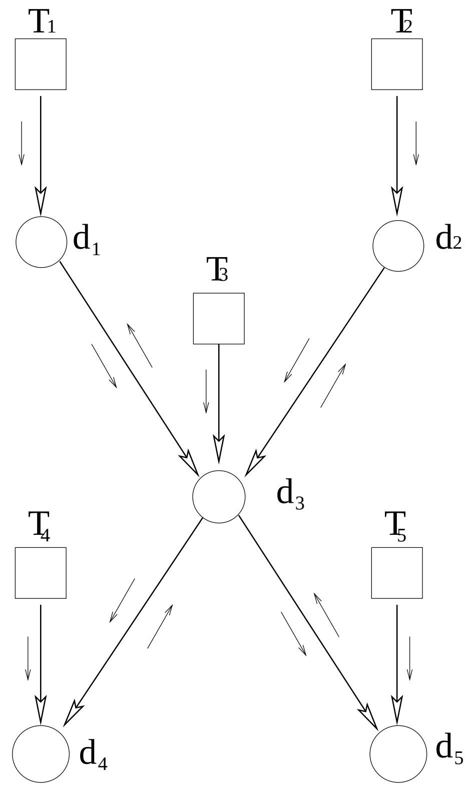

Figure 5.1: A sample citation Bayesian network.

In the citation Bayesian network described above, the text nodes are observed. As a text node has no parent node and only a child node, it sends probability message to is child node, which is the document having it as content. Each document, upon receiving belief probability from its text node, sends messages to its parents and its children document nodes. A detail of inference in Bayesian citation network is described in next sub section.

5.3.2 Inference in Citation Network

Assume we need to find the likelihood of a document d_j belonging to class c_i, $P(d_j = c_i)$ (a short notation for $P(\mathcal{C}(d_j) = c_i)$) based on available information from its text T_j and citation information N_j. We rewrite Bayesian rule from equation 4.5, but now d_j includes text information T_j and neighbouring information N_j:

$$P(d_j = c_i|T_j, N_j) = \frac{P(T_j, N_j|d_j = c_i)P(d_j = c_i)}{P(T_j, N_j)} \tag{5.4}$$

Again we make another independence assumption, which states that, in a document, the content T and the neighbouring N are statistical independent. The independence assumption is interpreted by the equation:

$$P(T_j, N_j|d_j = c_i) = P(T_j|d_j = c_i)P(N_j|d_j = c_i) \tag{5.5}$$

We also can cancel out the normalisation factor $\frac{1}{P(T_j, N_j)}$ (as in equation 4.6), equation 5.4 can be written as:

$$\begin{aligned} P(d_j = c_i|T_j, N_j) &\propto P(T_j|d_j = c_i)P(N_j|d_j = c_i)P(d_j = c_i) \\ &= P(T_j|d_j = c_i)P(d_j = c_i)P(N_j|d_j = c_i) \\ &= P(d_j = c_i|T_j)P(N_j|d_j = c_i) \end{aligned} \tag{5.6}$$

$P(d_j = c_i|T_j)$ is the probability of document d_j in class c_i given its content. The probability is exactly the category score computed by a content-based classifier. Therefore the improvement would be obtained by multiplying with $P(N_j|d_j = c_i)$.

If we further assume the independence among information from each neighbouring documents n_k given d_j, we can write $P(N_j|d_j = c_i)$ as

$$\begin{aligned} P(N_j|d_j = c_i) &= \prod_{k:d_k \to d_j} P(n_k|d_j = c_i) \\ &= \prod_{k:d_k \to d_j} \frac{P(d_j = c_i|n_k)P(n_k)}{P(d_j = c_i)} \\ &= \prod_{k:d_k \to d_j} \frac{P(d_j = c_i, n_k)}{P(d_j = c_i)} \end{aligned} \tag{5.7}$$

where d_k is in the set of neighbouring documents of d_j and n_k is the information d_j obtains from d_k.

The node d_k is actually not observed. In fact, it can be computed by information from its text node, which is equivalent to $P(d_k = c_i|T_k)$, and its neighbouring $P(d_k = c_i|N_k)$. In other words, the categories information of d_k is obtained from the content-based classification and labelling updating from its neighbour. d_k itself does not generate information but encapsulate all messages it receives and passes on to d_j. Suppose messages received by d_k are from some sources of evident e_k (which are T_k and N_k as shown on Figure 5.2), message n_k from d_k to d_j is actually is e_k. Therefore we have

$$P(d_k|e_k) = P(d_k|T_k, N_k) \tag{5.8}$$

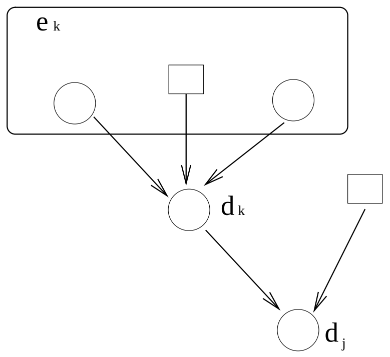

Figure 5.2: Information from d_k to d_j

and

$$P(d_j = c_i, n_k) = P(d_j = c_i, e_k) \tag{5.9}$$

As can be seen from Figure 5.2, information from e_k is sent to d_j via d_k. Therefore, the joint probability $P(d_j = c_i, e_k)$ can be factored as

$$\begin{aligned} P(d_j = c_i, e_k) &= \sum_l P(d_j = c_i, d_k = c_l, e_k) \\ &= \sum_l P(d_j = c_i | d_k = c_l) P(d_k = c_l | e_k) P(e_k) \\ &= \sum_l P(d_j = c_i | d_k = c_l) P(d_k = c_l | T_k, N_k) P(e_k) \end{aligned} \tag{5.10}$$

where l are all possible values of class c_l.

$P(e_k)$ is again the common factor for all categories c_i and thus can be normalised. Plug equation 5.10 into equation 5.7 and equation 5.6, we obtain:

$$\begin{aligned} &P(d_j = c_i | T_j, N_j) \propto \\ &\quad P(d_j = c_i | T_j) \prod_{k:d_k \to d_j} \frac{\sum_l P(d_j = c_i | d_k = c_l) P(d_k = c_l | e_k) P(e_k)}{P(d_j = c_i)} \\ &= P(d_j = c_i | T_j) \prod_{k:d_k \to d_j} \frac{\sum_l P(d_j = c_i | d_k = c_l) P(d_k = c_l | T_k, N_k) P(e_k)}{P(d_j = c_i)} \\ &\propto P(d_j = c_i | T_j) \prod_{k:d_k \to d_j} \frac{\sum_l P(d_j = c_i | d_k = c_l) P(d_k = c_l | T_k, N_k)}{P(d_j = c_i)} \end{aligned} \tag{5.11}$$

In the above formula, $P(d_j = c_i | T_j)$ is calculated from the embedded content-based classification. $P(d_k = c_l | T_k, N_k)$ is the previous calculation on d_k. The term $P(d_j = c_i | d_k = c_l)$ and $P(d_j = c_i)$ can be learned from training set.

5.3.3 Learning the Model

The independence assumption we made above makes learning the network easier. Instead of learning the full conditional probability distribution $P(d_j|\text{parent}(d_j))$, the model needs only to learn the distribution $P(d_j = c_i | d_k = c_l)$ that can applied to all citations dependences.

As a citation link can be either in-link (cited) or out-link (citing), we need to estimate two conditional probability distributions, one for the case d_j cites $d_k)$ and one d_j is cited by $d_k)$. Each distribution is a matrix $|C| \times |C|$. The entry (i,l) of the in-link matrix is estimated from training set by the maximum likelihood estimation:

$$\begin{aligned} \text{in-link}_{il} &= P(d_j = c_i | d_k = c_l) \\ &= \frac{L_{li}}{\sum_k^{|C|} L_{lk}} \end{aligned} \tag{5.12}$$

where L_{lk} is the number of citation links from a document in class c_l to c_k.

Similarly, the matrix out-link is also estimated as:

$$\begin{aligned} \text{out-link}_{il} &= P(d_j = c_i | d_k = c_l) \\ &= \frac{L_{il}}{\sum_k^{|C|} L_{kl}} \end{aligned} \tag{5.13}$$

Finally, the prior probability is estimated as similar to equation 4.9:

$$P(d_j = c_i) = \frac{|S_i|}{|S|} \tag{5.14}$$

5.3.4 Probabilistic Labelling Update Summary

The classification system using PLU is summarised as follows:

- Train the classifier using content-based classification method as described in Chapter 4. Following the discussing of PLU, the reasonable content-based classification method should be NB as the PLU assumes the category scores from content-based classification is $P(d_j = c_i|T_j)$, which is exactly the output of the NB classifier.

- Use the training set to learn the parameters of the Bayesian citation network as described by equations 5.12, 5.13 and 5.14.

- Apply the content-based classification to classify unknowns documents. More specifically, calculate category scores $s_{ji} = P(d_j = c_i)|T_j)$ for each document d_j.

- Iteratively update the labellings of all unknowns documents using equation 5.11. The update process terminates when the system converge i.e. the changes of labelling is small enough.

5.4 Update Modes

The two discussed updating functions LLU and PLU show how information is obtained from neighbouring documents to integrate with existing labelling of a document. There are two variations of update mode which specify at a time, information from which neighbour document(s) to be incorporated. They are *batch update* and *information exchange*, which are described in the next two sub sections. The two modes seem similar, but they turn out to have different effect on the classification systems.

5.4.1 Batch Update mode

In batch update mode, the labellings of ALL neighbouring documents are incorporated at once. In other words, the updating function computes the labelling of one document before doing so of of another. The pseudocode for the batch update is:

```
foreach document $d_j$
  foreach document $d_k$ neighbouring with $d_j$
      $V_j \;=\; F(V_j, V_k)$
```

5.4.2 Information Exchange

In contrast to batch update mode, in information exchange mode, the updating function incorporates labelling from one neighbour document only. When document d_j is updated by labelling from document d_k, labelling of d_k is also updated by labelling of d_j. The updating is similar to exchanging information between the two neighbouring documents.

```
foreach citation $d_i \;\rightarrow\; d_k$
  $V_j \;=\; F(V_j, V_k)$
  $V_k \;=\; F(V_k, V_j)$
```

5.5 Experiment Setup

From the content-based classification methods described in chapter 4, we choose naive Bayes classification method as the baseline due to the following reasons:

- NB is among the best content-based classification methods, especially with sufficiently large training set, as shown in Chapter 4.
- NB is computationally cheap, in terms of both training and classifying. It would be a practical choice for classification of large document collections.
- The underlying foundation of NB bases on Bayes theorem, which is also the foundation of the PLU method.

Similar to experiments in the previous chapter, each experiment is performed in different dataset configurations CORA20, CORA30, CORA40, CORA50 and CORA60.

As this experiment involves in training classifier using citation link, and to make the training set realistic, only citations between documents in training set are used. If a document in training set has a citation link with some other document not in the training set, the link is not used for training. However, the link may later be used to propagate information to the unknown document.

The training set is used to train a content-based NB classifier, as described in Chapter 4 and train the Bayesian citation network. The content-based NB classifier first classify the test set so that every test document has an initial labelling. One of the labelling update methods is then applied to update labellings iteratively . Each updating method is tested in both batch update and information exchange modes.

In order to apply LLU for labelling update, a value for the updating rate parameter η need to be specified. Getting an optimal value for η requires some empirical experiment. We run LLU using various values of η and record the performance of each of them. The values used in our experiments are η = .05, .10, .15, .20, .25, .30, .35 and .40.

5.6 Results and Discussion

This section presents the results of combination of text information and citation link for scientific document classification. We present the results of LLU and PLU in the next two sub sections, follow by a comparison of the two methods.

5.6.1 Results of LLU

We experiment LLU in different updating rates. Their results are shown in two tables 5.1 and 5.2. Tables 5.1 show the performance of LLU using information exchange mode and tables 5.2 presents that of using batch update mode. On each table, the left most column shows the configuration of dataset, which specifies the size of the training set. The results of each setting are shown in three columns. The first column shows the mean accuracy of the classification system. The second column shows the improvement in accuracy in compared with using only NB for content-based classification and the last column shows the number of average iterations for the labelling updating process to converge.

Dataset	$\eta = 0.05$			$\eta = 0.10$			$\eta = 0.15$			$\eta = 0.20$		
	Acc.	Impr.	Iter.	Acc.	Impr.	Iter.	Acc.	Impr.	Iter.	Acc.	Impr.	Iter.
CORA20	78.42%	20.41%	55	78.29%	20.28%	52	78.22%	20.21%	35	78.04%	20.03%	27
CORA30	80.28%	10.42%	36	80.13%	10.27%	32	79.90%	10.04%	25	79.80%	9.94%	17
CORA40	81.76%	8.31%	25	81.55%	8.10%	22	81.30%	7.85%	18	81.15%	7.70%	14
CORA50	82.77%	3.32%	20	82.59%	3.14%	18	82.42%	2.97%	13	82.23%	2.78%	11
CORA60	84.14%	2.17%	15	84.08%	2.11%	13	84.08%	2.11%	9	83.87%	1.90%	8
Dataset	$\eta = 0.25$			$\eta = 0.30$			$\eta = 0.35$			$\mu = 0.40$		
	Acc.	Impr.	Iter.	Acc.	Impr.	Iter.	Acc.	Impr.	Iter.	Acc.	Impr.	Iter.
CORA20	77.87%	19.86%	20	77.57%	19.56%	18	77.43%	19.42%	16	77.23%	19.22%	15
CORA30	79.75%	9.89%	16	79.66%	9.80%	14	79.62%	9.76%	11	79.51%	9.65%	10
CORA40	81.04%	7.59%	12	80.89%	7.44%	12	80.71%	7.26%	8	80.53%	7.08%	7
CORA50	82.04%	2.59%	8	81.91%	2.46%	9	81.76%	2.31%	6	81.70%	2.15%	5
CORA60	83.74%	1.77%	5	83.61%	1.64%	4	83.49%	1.52%	4	83.42%	1.45%	3

Table 5.1: Improvement by LLU using information exchange mode.

Dataset	$\mu = 0.05$			$\mu = 0.10$			$\mu = 0.15$			$\eta = 0.20$		
	Acc.	Impr.	Iter.	Acc.	Impr.	Iter.	Acc.	Impr.	Iter.	Acc.	Impr.	Iter.
CORA20	62.01%	4.00 %	16	62.02%	4.01 %	14	62.02%	4.01 %	12	61.99%	3.98 %	10
CORA30	73.97%	4.11 %	13	73.96%	4.10 %	11	73.95%	4.09 %	10	73.96%	4.10 %	7
CORA40	77.25%	3.80 %	10	77.04%	3.59 %	9	77.02%	3.57 %	8	77.02%	2.57 %	6
CORA50	81.83%	2.38 %	9	81.62%	2.17 %	8	81.59%	2.14 %	6	81.60%	2.15 %	5
CORA60	83.67%	1.70 %	6	83.68%	1.71 %	6	83.65%	1.68 %	5	83.66%	1.69 %	4
Dataset	$\eta = 0.25$			$\eta = 0.30$			$\eta = 0.35$			$\eta = 0.40$		
	Acc.	Impr.	Iter.	Acc.	Impr.	Iter.	Acc.	Impr.	Iter.	Acc.	Impr.	Iter.
CORA20	62.05%	4.04 %	10	62.04%	4.03 %	9	62.04%	4.03 %	8	62.01%	4.00 %	7
CORA30	73.96%	4.10 %	6	73.96%	4.10 %	7	73.94%	4.08 %	6	73.93%	4.07 %	5
CORA40	76.90%	3.45 %	5	77.03%	3.58 %	5	77.01%	3.56 %	5	76.98%	3.52 %	4
CORA50	81.41%	1.96 %	4	81.57%	2.12 %	5	81.57%	2.12 %	5	81.21%	2.06 %	4
CORA60	83.61%	1.64 %	4	83.65%	1.68 %	4	83.62%	1.65 %	3	83.51%	1.54 %	3

Table 5.2: Improvement by LLU using batch update.

In general, experiments show that LLU with any settings always improves the classification system. Particularly, on CORA20 dataset, LLU with information exchange update mode improves the classification by 20% (from 58.01% to 78.42%). However, the improvement is less in the dataset with larger training set. This is because the performance of the content-based NB classification performs well on large training set.

Examining LLU with various updating rates, we observe that smaller updating rates tend to produce better classification accuracy. However, the tradeoff is that they require more iterations to converge. As shown on the first row of table 5.1, the LLU using information exchange mode and $\eta = .05$ takes on average 55 iterations to gain 20.41% while that of $\eta = .40$ converges after average 15 iterations and gains 19.22%.

The information exchange update mode is shown to perform much better than batch update mode. On CORA20 dataset and $\eta = .05$, LLU with information exchange mode can increase the accuracy 20.41% while that with batch update can improve only 4.00%. However, the batch update results in a faster convergence. For example, the above setting for batch update converges after 16 iterations while that using information exchange mode takes 55 iterations.

5.6.2 Results of PLU

Table 5.3 shows results of PLU with both information exchange and batch update modes. The format of the table is similar to the two tables described in the previous sub section.

Dataset	Information exchange			Batch update		
	Acc.	Impr.	Iter.	Acc.	Impr.	Iter.
CORA20	69.06%	11.05 %	250	69.74 %	11.73 %	186
CORA30	80.06%	10.20 %	187	78.95 %	9.09 %	153
CORA40	81.69%	8.24 %	155	80.95 %	7.50 %	81
CORA50	82.70%	3.25 %	110	82.03 %	2.58 %	75
CORA60	84.15%	2.18 %	75	83.50 %	1.53 %	62

Table 5.3: Improvement by PLU

One observation of the PLU method is that, it updates in information exchange node always converges, while PLU using batch update results in several divergences, i.e. the updating method does not converge in a number of cases. The divergent cases make up 40% of all experiments. The results of PLU using batch update shown on table 5.3 are based on the convergent cases only.

The PLU method also produces significant improvement of classification accuracy whenever it converges. On CORA20 dataset, an improvement of more than 11% is gained from both update modes.

The goodness of information exchange update mode is again confirmed by the PLU updating method. In most cases, PLU with information exchange performs slightly better than that with batch update. It is also worth reminding that the PLU with batch update is not reliable as it fails to converge in a number of cases.

5.7 Comparison between LLU & PLU and further discussion

Table 5.4 shows the comparison of the best results in terms of classification accuracy of the two methods LLU and PLU. The best results of LLU are taken when updating rate $\eta = 0.05$

and update mode is information exchange mode. The best results of PLU are also from information exchange update mode.

Dataset	LLU			PLU		
	Acc.	Impr.	Iter.	Acc.	Impr.	Iter.
CORA20	78.42%	20.41%	55	69.06%	11.05 %	250
CORA30	80.28%	10.42%	36	80.06%	10.20 %	187
CORA40	81.76%	8.31%	25	81.69%	8.24 %	155
CORA50	82.77%	3.32%	20	82.70%	3.25 %	110
CORA60	84.14%	2.17%	15	84.15%	2.18 %	75

Table 5.4: Comparison of LLU and PLU.

Comparing the two methods, we see that they perform comparably in case the content-based classification performance is significantly good. On dataset CORA30, the accuracy of the naive Bayes content-based classifier is 79.86%, both methods improve the accuracy to more than 80%. A similar observation is for dataset CORA60, where both combination systems can reach accuracy of 84.14%.

However, when the content-based classification accuracy is low the LLU can produce much better improvement. Particularly, on CORA20 dataset where the content-based classifier can obtain an accuracy of only 58.1%, the improvement obtained by LLU is 20.41%, almost double the improvement by PLU, which is only 11.05%.

Not only having better performance on some datasets, the LLU converge much faster than PLU in all experiments carried out. On CORA20 dataset, the PLU takes an average 250 iterations to converge while the LLU iterates only 55 times. On another dataset, CORA60, the average number of LLU is 15, much smaller than 75 iterations of PLU

While PLU does make improvement, the improvement is not as big as expected. It does not perform better than LLU which is a very simple model. We are expecting that, the message passing "send message anyway" causes imprecision of information passing. It is expected that some exact inference algorithms such as junction tree [19], cutset conditioning [18] or clustering [37] would perform better in information propagation.

The improvement of performance by combination of document contents and citation links confirms that citation structure is helpful for scientific document classification. A good model to exploit the citation structure would significantly improve the classification system.

5.8 Summary

In this chapter, we show that citation link can be combined with document content to improve classification performance. We described two methods to integrate the citation link of scientific documents with a content-based classification method. The two developed methods, LLU and PLU are shown to significantly improve the scientific document classification.

Given an updating method, the update mode does matter in system performance. In general, the information exchange update mode performs better than batch update. Particularly in PLU method, batch update tends to be less reliable as it results in divergence in a number of cases.

For future work, we will investigate other models for citation link structure. Particularly, other machine learning approaches such as neural networks and genetic programming will be investigated. We will also consider a novel training architecture in which the link citation trainer used outputs of the content-based on training set as training data.

The probability propagation mechanism in PLU in this project is simplified to "send anyway" for tractability. That inevitably results in loss of precision of probability. This suggests that a more precise probability propagation could be applied to gain better performance Such inference algorithms are junction tree, cutset conditioning and clustering. Alternatively, some approximation inference such as Monte Carlo [24] and variational approximation [17] could also be applied.

The iteration on Bayesian network suggests the inference on Bayesian network over time. That leads to a possible direction for future works to apply a temporal model for labellings update. A Hidden Markov model, or more generally, a dynamic Bayesian network could be the candidate for citation modelling.

Chapter 6

Conclusions

This chapter presents the main conclusions of the project and describes several directions for future work.

6.1 Main Conclusions

The project's overall goal is to develop an adaptive approach for scientific document classification. Specifically, it aims to make the best use of available information for a high performance classification system.

The project investigates both document contents and citation links as sources of information for classification. For content-based classification, it evaluates a number of the most common classification methods to find the most suitable one. It also examines the discriminative ability of phrases for classification. For citation link exploitation, the project developed two methods to combine citation link information with content information. The following two subsections summarise the achievements of this research.

6.1.1 Content-based classification

In the investigation of using document contents for classification, the project evaluates a number of common document classification methods, namely K-nearest neighbours, nearest centroid, naive Bayes and decision trees. They are thoroughly tested on real world scientific document datasets using training sets of different sizes.

The evaluation shows that, among the tested methods, the naive Bayes and K-nearest neighbours methods are among the best. In order to use the naive Bayes method effectively, the training set need to be sufficiently large. The K-nearest neighbours performs well in any size of training set but would suffer in terms of speed if the training set is too large.

The project also finds that, phrases are useful for classification. A naive use of phrases degrades the classification system but selective phrases used together with words could improve classification accuracy. The project demonstrates the usefulness of phrases by showing that, a feature set of good phrases and words performs better than another feature set of words only with the same size.

Furthermore,the information gain feature selection is found to be able to get rid of the majority of terms while gaining some improvement in performance. In this project, the construction of phrases using bigram model increases the feature space dimensionality to 20 times higher, but selecting only 0.25% of the total features, which is equivalent to only 5% of the original number of features, produces a better classification performance.

6.1.2 Combining document contents and citation structure

Our investigation of citation links shows that citation structure can be combined with one of the content-based classification method. Our framework for combining document contents and citation links includes a content-based classifier and an updating function. The labellings of papers computed by the content-based classier are iteratively updated by the updating function using categories information from neighbouring documents.

We developed two labelling updating methods, linear labelling update(LLU) and probabilistic labelling update(PLU). The LLU method updates the labelling of a document by adding the labellings of its neighbours multiplied by a constant. The PLU applies the Bayesian network to model the citation structure and uses message passing algorithm for propagate category information around the citation network.

The two methods are found to significantly improve the classification system. Generally the two updating methods are comparable, but when the accuracy of the content-based classifier is low, the LLU performs better than PLU.

6.2 Future Work

While the project does achieve its goals, there is much room for further development. Some of possible directions to be considered are:

- The strengths and weaknesses of each content-based classification method is examined. The objective is, given a document corpus, a good classification method could be chosen analytically rather than empirically.
- The success of PLU suggests that the hidden characteristics of citations can be learned. Some other learning approaches such as neural network or genetic programming could be investigated to learn the citation structure.
- The inference approach currently applied by PLU, which is "send message anyway", is not generally considered good in Bayesian network inference as it introduces imprecision and divergences. Other more reliable inference algorithms such as junction tree, cutset conditioning and clustering could be investigated to design a better updating mechanism.
- Instead of using message passing algorithm, an approximate inference algorithm could also be used. Some examples of these are sampling, variational methods and loopy propagation.
- A number of temporal approaches such as hidden Markov model or dynamic Bayesian network are suggested to be used for modelling the citation structure.

Appendix A

List of stop words

able
about
above
abstract
according
accordingly
acknowldegment
acknowldegments
across
actually
after
afterwards
again
against
all
allow
allows
almost
alone
along
already
also
although
always
among
amongst
and
another
any
anybody
anyhow
anyone
anything
anyway
anyways
anywhere
apart
appear
appreciate
appropriate
are
around
aside
ask
asking
associated
author
available
away
awfully
became
because
become
becomes
becoming
been
before
beforehand

behind
being
believe
below
beside
besides
best
better
between
beyond
both
brief
but
came
can
cannot
cant
cause
causes
certain
certainly
changes
clearly
com
come
comes
concerning
conclusion
conclusions
consequently
consider
considering
contain
containing
contains
corresponding
could
course
currently
definitely
department
described
despite
did
different
does
doing
done
down
downwards
during
each
edu
eight
either
else
elsewhere
email
emails
enough
entirely
especially
etc
even
ever
every
everybody
everyone
everything
everywhere
exactly
example
except
experiment
experiments
far

few
fifth
first
five
followed
following
follows
for
former
formerly
forth
four
from
further
furthermore
get
gets
getting
given
gives
goes
going
gone
got
gotten
greetings
had
happens
hardly
has
have
having
hello
help
hence
her
here
hereafter
hereby
herein
hereupon
hers
herself
him
himself
his
hither
hopefully
how
howbeit
however
ignored
immediate
inasmuch
inc
indeed
indicate
indicated
indicates
inner
insofar
instead
into
introduction
inward
its
itself
just
keep
keeps
kept
know
known
knows
last
lately

later
latter
latterly
least
less
lest
let
like
liked
likely
little
look
looking
looks
ltd
mainly
many
may
maybe
mean
meanwhile
merely
might
more
moreover
most
mostly
much
must
myself
name
namely
near
nearly
necessary
need
needs
neither
never
nevertheless
new
next
nine
nobody
non
none
noone
nor
normally
not
nothing
novel
now
nowhere
obviously
off
often
okay
old
once
one
ones
only
onto
other
others
otherwise
ought
our
ours
ourselves
out
outside
over
overall
own

particular
particularly
per
perhaps
placed
please
plus
possible
presumably
probably
provides
que
quite
rather
really
reasonably
regarding
regardless
regards
relatively
research
researches
respectively
right
said
same
saw
say
saying
says
school
second
secondly
see
seeing
seem
seemed
seeming
seems
seen
self
selves
sensible
sent
serious
seriously
seven
several
shall
she
should
since
six
some
somebody
somehow
someone
something
sometime
sometimes
somewhat
somewhere
soon
sorry
specified
specify
specifying
still
study
sub
such
sup
sure
take
taken
tell

tends
than
thank
thanks
thanx
that
thats
the
their
theirs
them
themselves
then
thence
there
thereafter
thereby
therefore
therein
theres
thereupon
these
they
think
third
this
thorough
thoroughly
those
though
three
through
throughout
thru
thus
together
too
took
toward
towards
tried
tries
truly
try
trying
twice
two
under
unfortunately
university
unless
unlikely
until
unto
upon
use
used
useful
uses
using
usually
uucp
value
various
very
via
viz
want
wants
was
way
welcome
well
went
were
what

whatever
when
whence
whenever
where
whereafter
whereas
whereby
wherein
whereupon
wherever
whether
which
while
whither
who
whoever
whole
whom
whose
why
will
willing
wish
with
within
without
wonder
would
yes
yet
you
your
yours
yourself
yourselves
zero

Bibliography

[1] Citeseer, http://citeseer.ist.psu.edu/.

[2] Google Scholar, www.scholar.google.com.

[3] BORKO, H., AND BERNICK, M. Automatic Document Classification. *J. ACM 10*, 2 (1963), 151–162.

[4] BRIN, S., AND PAGE, L. The anatomy of a Large-scale Hypertextual Web search Engine. *Computer Networks and ISDN Systems 30*, 1–7 (1998), 107–117.

[5] CAO, M. D., AND GAO, X. Combining Contents and Citations for Scientific Document Classification. *Lecture Notes in Artificial Intelligence*, 3809 (2005), 143–152.

[6] CHAKRABARTI, S., DOM, B. E., AND INDYK, P. Enhanced hypertext categorization using hyperlinks. In *Proceedings of SIGMOD-98, ACM International Conference on Management of Data* (Seattle, US, 1998), L. M. Haas and A. Tiwary, Eds., ACM Press, New York, US, pp. 307–318.

[7] COWELL, R. Introduction in Inference in Bayesian Networks. MIT Press, Cambridge, MA, USA, 1999, pp. 9–26.

[8] CRAVEN, M., AND SLATTERY, S. Relational Learning with Statistical Predicate Invention: Better Models for Hypertext. *Mach. Learn. 43*, 1-2 (2001), 97–119.

[9] DUMAIS, S. T., PLATT, J., HECKERMAN, D., AND SAHAMI, M. Inductive learning algorithms and representations for text categorization. In *Proceedings of CIKM-98, 7th ACM International Conference on Information and Knowledge Management* (Bethesda, US, 1998), G. Gardarin, J. C. French, N. Pissinou, K. Makki, and L. Bouganim, Eds., ACM Press, New York, US, pp. 148–155.

[10] FUHR, N., AND KNORZ, G. Retrieval test evaluation of a rule-based automated indexing (AIR/PHYS). In *Proceedings of SIGIR-84, 7th ACM International Conference on Research and Development in Information Retrieval* (Cambridge, UK, 1984), C. J. Van Rijsbergen, Ed., Cambridge University Press, pp. 391–408.

[11] GETOOR, L., SEGAL, E., TASKAR, B., AND KOLLER, D. Probabilistic Models of Text and Link Structure for Hypertext Classification. In IJCAI Workshop on Text Learning: Beyond Supervision, 2001.

[12] GHAHRAMANI, Z. Graphical Models: Parameter Learning. In *Handbook of Brain Theory and Neural Networks*, M. A. Arbib, Ed., 2 ed. MIT Press, 2003, pp. 486–490.

[13] HAN, E.-H., AND KARYPIS, G. Centroid-Based Document Classification: Analysis and Experimental Results. In *PKDD '00: Proceedings of the 4th European Conference on*

Principles of Data Mining and Knowledge Discovery (London, UK, 2000), Springer-Verlag, pp. 424–431.

[14] HAYES, P. J., AND WEINSTEIN, S. P. CONSTRUE/TIS: a system for content-based indexing of a database of news stories. In *Proceedings of IAAI-90, 2nd Conference on Innovative Applications of Artificial Intelligence* (Boston, US, 1990), A. Rappaport and R. Smith, Eds., AAAI Press, Menlo Park, US, pp. 49–66.

[15] HECHT-NIELSEN, R. Counter propagation networks. In *Proceedings of the IEEE First International Conference on Neural Networks* (1987), vol. 2, pp. 19–32.

[16] JOACHIMS, T. Text categorization with support vector machines: learning with many relevant features. In *Proceedings of ECML-98, 10th European Conference on Machine Learning* (Chemnitz, DE, 1998), C. Nédellec and C. Rouveirol, Eds., Springer Verlag, Heidelberg, DE, pp. 137–142. Published in the "Lecture Notes in Computer Science" series, number 1398.

[17] JORDAN, M. I., GHAHRAMANI, Z., JAAKKOLA, T. S., AND SAUL, L. K. An introduction to variational methods for graphical models. *Mach. Learn. 37*, 2 (1999), 183–233.

[18] JORDAN, M. I., AND WEISS, Y. Graphical Models: Probabilistic Inference. In *Handbook of Brain Theory and Neural Networks*, M. A. Arbib, Ed., 2 ed. MIT Press, 2003, pp. 490–496.

[19] LAURITZEN, S. L., AND SPIEGELHALTER, D. J. Local computations with probabilities on graphical structures and their application to expert systems. 415–448.

[20] LEWIS, D. An Evaluation of Prasal and Clustered Representation of Text Categorisation Tasks. In *Proceedings of SIGIR-92, 15th ACM International Conference on Reseach and Deveplopment in Information Retrieval* (1992), pp. 289–297.

[21] LEWIS, D. D. Naive (bayes) at forty: The independence assumption in information retrieval. In *Proceedings of ECML-98, 10th European Conference on Machine Learning* (Chemnitz, DE, 1998), C. Nédellec and C. Rouveirol, Eds., Springer Verlag, Heidelberg, DE, pp. 4–15. Published in the "Lecture Notes in Computer Science" series, number 1398.

[22] LEWIS, D. D., AND RINGUETTE, M. A comparison of two learning algorithms for text categorization. In *Proceedings of SDAIR-94, 3rd Annual Symposium on Document Analysis and Information Retrieval* (Las Vegas, US, 1994), pp. 81–93.

[23] LUHN, H. P. The automatic creation of literature abstracts. *IBM Journal of Research and Development*, 2 (1958).

[24] MACKAY, D. J. C. Introduction to Monte Carlo Methods. In *Learning in graphical models*. MIT Press, Cambridge, MA, USA, 1999, pp. 175–204.

[25] MCCALLUM, A., AND NIGAM, K. A comparison of event models for naive bayes text classification. In *Proceedings of AAAI-98, Workshop on Learning for Text Categorization* (1998).

[26] MCCALLUM, A. K. Mallet: A machine learning for language toolkit. http://mallet.cs.umass.edu, 2002.

[27] MCCALLUM, A. K., NIGAM, K., RENNIE, J., AND SEYMORE, K. Automating the Construction of Internet Portals with Machine Learning. *Information Retrieval 3*, 2 (2000), 127–163.

[28] MITCHELL, T. *Machine Learning*. McGraw-Hill, 1997.

[29] NIGAM, K., LAFFERTY, J., AND MCCALLUM, A. Using Maximum Entropy for Text Classification. In *IJCAI-99 Workshop on Machine Learning for Information Filtering* (1999), pp. 61–67.

[30] OH, H.-J., MYAENG, S. H., AND LEE, M.-H. A practical hypertext categorization method using links and incrementally available class information. In *Proceedings of SIGIR-00, 23rd ACM International Conference on Research and Development in Information Retrieval* (Athens, GR, 2000), N. J. Belkin, P. Ingwersen, and M.-K. Leong, Eds., ACM Press, New York, US, pp. 264–271.

[31] PEARL, J. *Probabilistic Reasoning in Intelligent Systems: Network of Plausible Inference*. Morgan Kaufmann publishers, 1988.

[32] PORTER, M. F. An Algorithm for Suffix Stripping. *Readings in Information Retrieval* (1997), 313–316.

[33] QIAN, G., SURAL, S., GU, Y., AND PRAMANIK, S. Similarity between euclidean and cosine angle distance for nearest neighbor queries. In *SAC '04: Proceedings of the 2004 ACM symposium on Applied computing* (New York, NY, USA, 2004), ACM Press, pp. 1232–1237.

[34] QUINLAN, J. R. Learning logical definitions from relations. *Mach. Learn. 5*, 3 (1990), 239–266.

[35] R. O. DUDA, P. E. H., AND STORK, D. G. *Pattern Classification*. Wiley-Interscience, 2001.

[36] RUIZ, M. E., AND SRINIVASAN, P. Hierarchical neural networks for text categorization. In *Proceedings of SIGIR-99, 22nd ACM International Conference on Research and Development in Information Retrieval* (Berkeley, US, 1999), M. A. Hearst, F. Gey, and R. Tong, Eds., ACM Press, New York, US, pp. 281–282.

[37] RUSSELL, S., AND NOVIG, P. *Artificial Intelligence: A Modern Approach*, 2 ed. Prentice Hall, 2005.

[38] SEBASTIANI, F. Machine learning in automated text categorization. *ACM Computing Surveys 34*, 1 (2002), 1–47.

[39] SHANAHAN, J. G., AND ROMA, N. Boosting support vector machines for text classification through parameter-free threshold relaxation. In *Proceedings of CIKM-03, 12th ACM International Conference on Information and Knowledge Management* (New Orleans, US, 2003), ACM Press, New York, US, pp. 247–254.

[40] TASKAR, B., SEGAL, E., AND KOLLER, D. Probabilistic Classification and Clustering in Relational Data. In *Proceeding of IJCAI-01, 17th International Joint Conference on Artificial Intelligence* (Seattle, US, 2001), B. Nebel, Ed., pp. 870–878.

[41] WITTEN, I. H., AND FRANK, E. *Data Mining, Practical Machine Learning Tools and Techniques with Java Implementations*. Morgan Kaufmann Publishers, 2000.

[42] YANG, Y. Expert network: effective and efficient learning from human decisions in text categorisation and retrieval. In *Proceedings of SIGIR-94, 17th ACM International Conference on Research and Development in Information Retrieval* (Dublin, IE, 1994), W. B. Croft and C. J. Van Rijsbergen, Eds., Springer Verlag, Heidelberg, DE, pp. 13–22.

[43] YANG, Y., AND CHUTE, C. G. An example-based mapping method for text categorization and retrieval. *ACM Transactions on Information Systems 12*, 3 (1994), 252–277.

[44] YANG, Y., AND LIU, X. A re-examination of text categorization methods. In *Proceedings of SIGIR-99, 22nd ACM International Conference on Research and Development in Information Retrieval* (Berkeley, US, 1999), M. A. Hearst, F. Gey, and R. Tong, Eds., ACM Press, New York, US, pp. 42–49.

[45] YANG, Y., AND PEDERSEN, J. O. A comparative study on feature selection in text categorization. In *Proceedings of ICML-97, 14th International Conference on Machine Learning* (Nashville, US, 1997), D. H. Fisher, Ed., Morgan Kaufmann Publishers, San Francisco, US, pp. 412–420.

[46] YEDIDIA, J. S., FREEMAN, W. T., AND WEISS, Y. Understanding Belief Propagation and its Generalizations. Tech. Rep. TR-2001-22, Mitsubishi Electric Research Laboratories, Inc., jan 2002.