An Investigation into the Use of Negation in Inductive Rule Learning for Text Classification

Thesis submitted in accordance with the requirements of
the University of Liverpool for the degree of Doctor in Philosophy
by
Stephanie H. L. Chua

June 2012

Dedication

To the love and joy of my life,
Justin, Ashlyn, Grace

Abstract

This thesis seeks to establish if the use of negation in Inductive Rule Learning (IRL) for text classification is effective. Text classification is a widely research topic in the domain of data mining. There have been many techniques directed at text classification; one of them is IRL, widely chosen because of its simplicity, comprehensibility and interpretability by humans. IRL is a process whereby rules in the form of *antecedent* $\Rightarrow$ *conclusion* are learnt to build a classifier. Thus, the learnt classifier comprises a set of rules, which are used to perform classification. To learn a rule, words from pre-labelled documents, known as features, are selected to be used as conjunctions in the rule antecedent. These rules typically do not include any negated features in their antecedent; although in some cases, as demonstrated in this thesis, the inclusion of negation is required and beneficial for the text classification task. With respect to the use of negation in IRL, two issues need to be addressed: (i) the identification of the features to be negated and (ii) the improvisation of rule refinement strategies to generate rules both with and without negation. To address the first issue, feature space division is proposed, whereby the feature space containing features to be used for rule refinement is divided into three sub-spaces to facilitate the identification of the features which can be advantageously negated. To address the second issue, eight rule refinement strategies are proposed, which are able to generate both rules with and without negation. Typically, single keywords which are deemed significant to differentiate between classes are selected to be used in the text representation in the text classification task. Phrases have also been proposed because they are considered to be semantically richer than single keywords. Therefore, with respect to the work conducted in this thesis, three different types of phrases (n-gram phrases, keyphrases and fuzzy phrases) are extracted to be used as the text representation in addition to the use of single keywords. To establish the effectiveness of the use of negation in IRL, the eight proposed rule refinement strategies are compared with one another, using keywords and the three different types of phrases as the text representation, to determine whether the best strategy is one which generates rules with negation or without negation. Two types of classification tasks are conducted; binary classification and multi-class classification. The best strategy in the proposed IRL mechanism is compared to five existing text classification techniques with respect to binary classification: (i) the Sequential Minimal

Optimization (SMO) algorithm, (ii) Naive Bayes (NB), (iii) JRip, (iv) OlexGreedy and (v) OlexGA from the Waikato Environment for Knowledge Analysis (WEKA) machine learning workbench. In the multi-class classification task, the proposed IRL mechanism is compared to the Total From Partial Classification (TFPC) algorithm. The datasets used in the experiments include three text datasets: 20 Newsgroups, Reuters-21578 and Small Animal Veterinary Surveillance Network (SAVSNET) datasets and five UCI Machine Learning Repository tabular datasets. The results obtained from the experiments showed that the strategies which generated rules with negation were more effective when the keyword representation was used and less prominent when the phrase representations were used. Strategies which generated rules with negation also performed better with respect to binary classification compared to multi-class classification. In comparison with the other machine learning techniques selected, the proposed IRL mechanism was shown to generally outperform all the compared techniques and was competitive with SMO.

Acknowledgements

Firstly, I would like to thank my financial sponsor, Universiti Malaysia Sarawak, for their financial assistance, without which I would not have been able to undertake my studies in Liverpool. I would also like to thank the Department of Computer Science, University of Liverpool for the funds provided to enable me to attend conferences so that I could present my work.

Secondly, I would like to thank my fellow PhD "lab mates", Santhana Chaimontree, Puteri Nor Ellyza Nohuddin, Mohd Hanafi Ahmad Hijazi, Ayesh Alshukri, Matías Fernando García-Constantino, Emmanuel Agiriga and Maya Wardeh. I appreciate the wonderful times we had, engaging in academic discussions about our PhD work and chatting about our life. I will cherish these memories with much affection. I do sincerely hope our friendship can last beyond the end of our PhD study. Not forgetting also, my thanks to fellow PhD colleagues who demonstrated on various courses with me, other PhD colleagues, lecturers and departmental staffs who in one way or another, made my time in the department enjoyable and memorable. To other friends and acquaintances whom I have met outside of the department throughout my time in Liverpool, thank you for your friendship, warmth and kindness. All of you have made my family and I feel at "home" here in Liverpool.

Thirdly, I am grateful for the support, encouragement, patience, love and care given to me by my husband Justin, who stood steadfastly by my side throughout the duration of my PhD studies. He has been wonderful and gracious in every way. Coming to Liverpool with me and leaving his job at the height of his career so that I can pursue my PhD is without a doubt the biggest sacrifice he has ever had to make. For that, I am forever indebted. Thank you also for braving the harsh weather and enduring the inconveniences so that I did not have to. I do hope that one day, I will be able to repay in kind, all that you have done for me and our family. To my darling daughter Ashlyn, I am thankful for the joy you bring when I see your face at the end of a long and tiring day. All your funny antics bring light to my darkest days. To my darling baby Grace, I am ever so grateful you came along and pushed me harder to finish my PhD. Writing my thesis with you at my side attempting to help me type was a challenging but joyful experience. I would not have wanted it any other way. I would also like to thank my grandma, parents, in-laws, other family members and relatives back home

for the support and encouragement they have given me while I have been away. Thank you for all the voice and video calls and not forgetting the wonderful packages of food sent from home. All these made us feel very much close to home indeed.

Finally, my most heartfelt appreciation and gratitude to Dr Frans Coenen, my main supervisor who constantly provides me with guidance and encouragement throughout the duration of my PhD. I could not have asked for a better supervisor who is kind and patient at all times. It really has been an honour to have been supervised by him. I have learnt a lot from his sharing of experience and knowledge. Through our regular meetings, he has always been there to advise, teach, talk and even joke. His feedbacks are constantly forthcoming and invaluable. Whenever I face a "crisis" in my research, he is always there to reassure me and to help me build up my confidence. He is certainly a "super" supervisor whom I have the utmost respect for. I could not have progressed the way I did if it were not for him. Frans, thank you from the bottom of my heart for all that you have done for me to ensure the success of attaining my PhD. I would also like to convey my special thanks to my second supervisor, Dr Grant Malcolm for his invaluable comments and feedbacks, especially for the in-depth discussions we had on my thesis chapters. Thank you also to Dr David Jackson, whom as my adviser, has provided me with invaluable feedbacks during my progress presentation every year.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The field of data mining has grown in significance over the last 20 years, in line with the increasing amount of data available for analysis. Data mining was originally concerned with tabular data, however its current application includes all kinds of data. Thus, data mining can be considered to comprise many sub-domains. These include: image mining, text mining, data stream mining and multimedia mining. The focus of this thesis is text mining. Text mining is different from the conventional tabular data mining in that it deals with unstructured text rather than structured tabular data.

Text mining is the process of extracting useful information from textual data sources such as documents, web pages, email, the free text element of questionnaires and so on. The challenges of text mining include mechanisms to effectively preprocess the text corpora to support the application of data mining techniques, and the derivation of algorithms that are capable of processing large document collections. Text mining encompasses a number of tasks including text classification, document summarization, named entity extraction and sentiment analysis among others. The task that is the focus of the research described in this thesis is text classification. Initially in the 1980s, text classifiers were built manually through a knowledge engineering process, where experts manually built classifiers by defining rules for classification [81]. One such system is the CONSTRUE system [41] built by the Carnegie Group for the Reuters news agency. However, with the steep increase in the quantity of digital documents available for analysis, manual classification by experts quickly became overwhelming, expensive and unfeasible. Hence, in the early 1990s, an automated approach to text classification began to take prominence [81]. The idea was to automatically build classifiers based on previous classification patterns. Thus, in this context, text classification is defined as a supervised learning task, whereby a classifier is built using a pre-labelled training set. The classifier can then be used to assign labels to new documents.

With the advent of the first text classifiers, research into text classification escalated. Parallel to this escalating growth, applications for text classification techniques also increased. Originally, text classification was directed at document collections (the

classification of newspaper articles was and remained a popular application area). Other application areas range from email spam filtering and fraud detection to specific applications such as medical journal abstract classification.

There are many machine learning techniques which have been applied to classification in general and text classification in particular. These include decision trees [71, 42, 47], nearest neighbour methods [94], Support Vector Machines (SVMs) [43], neural networks [94, 77], probabilistic Bayesian models [63, 94] and inductive rule learning (IRL) [4, 36, 22]. The latter is the focus of the research described in this thesis. IRL algorithms offer one distinct advantage over some other popular learning techniques, such as probabilistic models, neural networks and SVMs, in that they produce rules that are interpretable by humans. Thus, further verification and refinement (often needed with respect to many specific applications) can be done to improve the effectiveness of text classification.

In the context of IRL-based classifiers, a rule is represented in the following form:

$$antecedent \Rightarrow conclusion$$

where the *antecedent* consists of a conjunction of words from the documents in the dataset, called features, which occur together, and the *conclusion* is the resulting class label associated with the *antecedent*. For example, if a, b and c are features that appear in a document within a dataset, and x is a class label of that document, the rule $a \wedge b \wedge c \Rightarrow x$ is interpreted as "if a, b, and c occur together in a given document, then classify the document as belonging to the class x".

A text classification framework that uses IRL typically consists of three main processes, namely, document preprocessing, IRL and classification. Figure 1.1 shows a general IRL-based text classification framework.

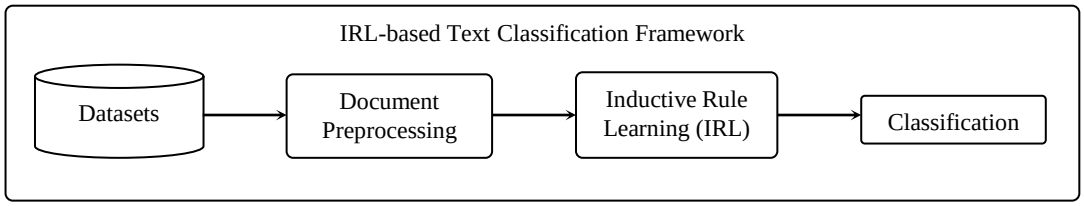

Figure 1.1: A general IRL-based text classification framework

Typically, a classifier built using IRL comprises rules that do not contain the negation of features in their antecedents. An example of a rule with negation is $a \wedge b \wedge \neg c \Rightarrow x$, which would be interpreted as, "if a and b occur together in a given document and c does not, then classify the document as belonging to class x". Negation can be included implicitly (with no explicit mechanism) by generating the negation of every feature available in the input feature space. However, this will substantially increase

the size of the input feature space. In the case of binary-valued feature sets, this will double the number of features. Increasing the size of the feature space makes very little difference given a small number of features. However, in the event of having a large number of features (as in the case of text mining), this is not a realistic proposition as it will be computationally intensive. A mechanism whereby negation can be included in IRL, without generating all the potential negated features, is therefore desirable. This is then the area of interest with respect to the work described in this thesis.

The rest of this chapter is organized as follows. Section 1.1 describes the motivation for the work conducted in this thesis. The research question and associated issues, with respect to the work described in this thesis, and the objectives to be achieved, are discussed in Section 1.2. The adopted research methodology is described in Section 1.3. Section 1.4 highlights the contributions of this thesis. The organization of the remainder of this thesis is described in Section 1.5. Lastly, Section 1.6 summarizes this chapter.

1.1 Motivation

The particular focus of this thesis is IRL for text classification that incorporates the ability to dynamically include negated features in the rule learning process. The motivation here is twofold. Firstly, it is conjectured that the inclusion of negated features in the IRL process can be used to generate rules with negation, which could be used for more effective classification, especially when the use of rules without negation is not sufficient to obtain an effective classification. Indeed, as will be illustrated later in this thesis, a text classification scenario which can only be resolved by including negated features in the rule generation process can be contrived. This is discussed in Chapter 3 of this thesis. Secondly, the dynamic identification of candidate features that can be negated is seen as desirable where the generation of the complete set of potential negations a priori is deemed to be unfeasible. In the case of datasets that have a small number of features, it is of course entirely feasible, and therefore justified, to include all potential feature negations as part of the input. However, this is not justified in the case of datasets with very large numbers of features. This type of dataset is exemplified by the document collections to which text classification is typically applied. Therefore, an approach to dynamically identify features to be negated in the IRL process is conjectured to be beneficial.

Document collections are typically represented, for text classification purposes, using the "bag-of-words" or "bag-of-phrases" representations. These collections generally feature large numbers of keywords and even larger numbers of phrases. The work described in this thesis is directed at both the bag-of-words and the bag-of-phrases representation. The latter was included because phrases contain semantic information that is not present in single keywords. It was also deemed desirable to investigate

whether the use of the bag-of-phrases representation can enhance IRL with negation (or otherwise) with respect to the text classification task.

1.2 Research Issues and Objectives

In the context of the classification rule generation process, when a rule is generated that covers both positive and negative documents, the rule has to be refined so that it is able to distinguish between the different document classes. In this context, positive documents are documents in the training set that are correctly classified by the current rule while negative documents are documents that are incorrectly classified. Generating rules without negation is straightforward: features that occur together in a positive document are used as conjunctions in a rule to separate the positive and negative documents. On the other hand, generating rules with negation requires the identification of the feature to be negated. This is one of the issues to be addressed in this thesis because negating all available features in the feature space is not a desirable option in the case of text classification, as text datasets tend to have an overwhelming number of features. Another issue concerns the strategies to refine a rule. When a rule can be refined with both positive and negated features, would it be better to refine the rule with a positive feature, thus generating a rule without negation, or to refine the rule with a negated feature, thus generating a rule with negation?

The desire to include negation in IRL requires the resolution of the above two issues. The first is the process for identifying appropriate features that can be advantageously negated. The second is the improvisation of rule refinement strategies which are capable of generating rules both with and without negation. In addition, the use of phrases is investigated to see whether it can enhance IRL with negation (or otherwise) with respect to the text classification task.

The aim of this thesis is therefore to investigate the use of negation in IRL, with the goal of establishing the effectiveness of IRL with negation using keywords and phrases as the text representation for the text classification task. The research question for this thesis can thus be formulated as follows: "*Can the use of negation in IRL, coupled with either a keyword or phrase representation, improve the effectiveness of the text classification task?*". To answer this research question, the following objectives need to be addressed:

- The derivation of an approach to dynamically identify features that can be advantageously negated in the IRL process, as opposed to including all possible negations as part of the input features.
- The investigation and derivation of different rule refinement strategies that can be used to generate rules both with and without negation in the IRL process.

- The identification and analysis of keywords, to be used in conjunction with IRL with negation, in a bag-of-words representation.
- The identification and analysis of a number of different types of phrases, to be used in conjunction with IRL with negation, in a bag-of-phrases representation.
- The evaluation of the use of negation in IRL by comparing the different proposed rule refinement strategies when using keywords in the text representation.
- The evaluation of the use of negation in IRL by comparing the different proposed rule refinement strategies when using phrases in the text representation.
- The overall evaluation of the proposed IRL mechanism by comparing it with existing machine learning techniques so as to determine its effectiveness with respect to the text classification task.

1.3 Research Methodology

To answer the research question laid out in Section 1.2, the following research methodology was adopted. The starting point for the work was an investigation into preprocessing strategies, an essential precursor to text classification. This initial investigation was founded on existing work on text classification, whereby several processes such as, data cleaning, keyword and phrase extraction, feature selection and text representation were identified as essential to prepare the input documents for the text classification task. Next, initial experiments were directed at establishing that there were classification scenarios that could only be resolved using IRL with negation. This resulted in a successful demonstration that this was indeed the case. The next stage was to consider an approach for identifying features appropriate for negation and strategies for generating rules which included both positive and negated features. Several strategies were derived and these will be described in Chapter 4 of this thesis. These strategies, which formed part of the proposed IRL mechanism, were then embedded into a text classification framework, initially designed to operate with a bag-of-words representation and then with a bag-of-phrases representation. Two types of classification settings were considered: (i) binary classification, where the classifier generated is used to classify a document as belonging to a class or not; and (ii) multi-class classification, where the classifier generated is used to classify a document as belonging to one of a collection of three or more classes.

A number of existing machine learning techniques were used to compare and evaluate the performance of the proposed IRL mechanism in the context of the text classification task. The datasets used in the experiments conducted included the 20-Newsgroups and Reuters-21578 datasets. These are two well-established text datasets that have been used extensively in the reported literature on text classification. In addition, a

real-life text dataset provided by the Small Animal Veterinary Surveillance Network (SAVSNET) project[1] was also used. Five tabular datasets taken from the UCI Machine Learning Repository were also used to evaluate the proposed IRL mechanism in terms of general data mining.

1.4 Research Contributions

The research contributions of this thesis are summarized as follows:

1. An approach to dynamically identify features that can be advantageously negated in the IRL process.

2. Several rule refinement strategies to generate rules both with and without negation as part of the IRL process.

3. An IRL mechanism which comprises the dynamic identification of features to be negated and the rule refinement strategies, included as part of a text classification framework to conduct both binary and multi-class classification of text datasets.

4. Extraction of different types of phrases (in addition to keywords) to be used with the bag-of-phrases representation so as to support the investigation of using IRL with negation for text classification.

5. A comprehensive evaluation of the proposed IRL mechanism, in comparison with existing machine learning techniques, in the context of text classification using both binary and multi-class classification tasks.

1.5 Thesis Organization

The rest of this thesis is organized as follows. Chapter 2 presents the background knowledge required to support the work described and a literature review of previous related work in the field of text classification. The chapter first gives an overview of the data mining field and then the classification task in general and then focuses on the text classification task. The use of keyword and phrase as the text representation is also discussed. The chapter goes on to describe techniques for feature selection, followed by a discussion on techniques for text classification. The evaluation measures used in text classification are also described. Chapter 3 presents some possible classification scenarios and includes an example situation which can only be resolved by using negation in IRL. Chapter 4 presents a framework for the IRL-based text classification. The chapter includes discussions on the document preprocessing phase, the IRL phase and the classification phase. This includes the description of five sub-processes in the document

[1] http://www.liv.ac.uk/savsnet/

preprocessing phase, namely: (i) data cleaning, (ii) keyword extraction, (iii) phrase extraction, (iv) feature selection and (v) text representation. Importantly, the proposed IRL mechanism (forming the IRL phase) is introduced and described in detail. Binary and multi-class classification for text classification is then described in the classification phase. The datasets used in this thesis are summarized in Chapter 5. Chapters 6 and 7 describe the experiments used to evaluate the proposed IRL mechanism and the results obtained using the keyword and phrase representations respectively. Finally, Chapter 8 summarizes the thesis, presents the main findings and contributions, and gives some suggestions for future directions.

1.6 Summary

This chapter has introduced the research work conducted in this thesis by providing the necessary background. The motivation for the research, the research issues and objectives, as well as the methodology for addressing the issues and achieving the objectives have been described. The contributions that stem from the research work conducted are also listed. The next chapter will provide some background information and a literature review of previous relevant work.

Chapter 2

Background and Literature Review

This chapter provides the relevant background knowledge concerning the research work carried out in this thesis. A critical review of related literature is also presented. The organization of this chapter is as follows. First, Section 2.1 gives an introductory overview of data mining. This is then followed by consideration of the classification task in data mining in Section 2.2. Section 2.3 describes text classification, including the text classification process and applications of text classification. The use of keywords and phrases in text classification is next considered in Section 2.4. This is followed by the techniques used for feature selection, which is discussed in Section 2.5. The discussion includes the Chi-Square (χ^2) measure in Sub-section 2.5.1 and Information Gain (IG) in Sub-section 2.5.2. The different techniques used for text classification are discussed in Section 2.6, which includes Support Vector Machines (SVMs) in Sub-section 2.6.1, Naive Bayes (NB) in Sub-section 2.6.2, Classification Association Rule Mining (CARM) in Sub-section 2.6.3, Inductive Rule Learning (IRL) in Sub-section 2.6.4 and IRL with negation in Sub-section 2.6.5. Section 2.7 describes the evaluation measures used for text classification. Finally, Section 2.8 summarizes the chapter.

2.1 Data Mining

In the current technologically advanced era, the amount of data that is collected and stored has grown faster than ever, thus significantly increasing the amount of electronic data available for analysis. It is no longer feasible to rely solely on manual human effort to make sense of all this data. Experts have long turned towards the use of computational resources to process data for the purpose of knowledge discovery. Thus, the field of Knowledge Discovery in Databases (KDD), a term coined in 1989 [31], has grown rapidly. KDD is a field concerned with techniques to discover hidden knowledge in data [30]. It involves a number of processes and one of the core steps in KDD is data mining. Figure 2.1 shows an overview of the KDD process. The process encompasses:

1. **Selection** - Data selection and retrieval according to relevance to the analysis task.

2. **Preprocessing** - Data cleaning by removing noise and inconsistent data.

3. **Transformation** - Data transformation into formats suitable for mining.

4. **Data Mining** - Application of intelligent techniques to discover and extract useful patterns from the data.

5. **Interpretation/Evaluation** - Interpretation and evaluation of the patterns mined from the data.

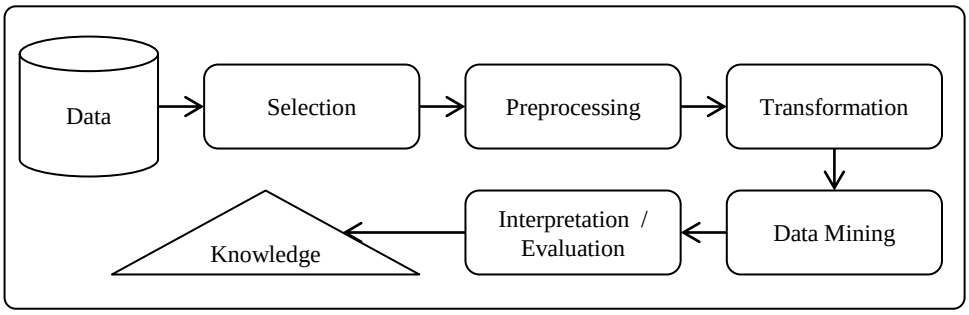

Figure 2.1: An overview of the KDD process (extracted from [30])

Data mining is the process concerned with the discovery and extraction of useful patterns from data. The term data mining encompasses a number of different tasks, including: classification, regression, clustering, summarization, dependency modeling and change and deviation detection [30]; each is briefly considered below.

Classification is defined as a process of learning patterns from pre-labelled examples and applying the learnt patterns to classify new examples whose labels are unknown. Classification can be applied to many domains. Examples include loan approval, patient diagnosis and the labelling of news stories.

Regression is a statistical method that learns a function from current data to predict future patterns. One of the most common applications is to predict consumer demand of a product based on some market variables. Other example applications include the prediction of the weight of individuals given their height and calorie intake and the prediction of the possibility of rain given some meteorological variables.

Clustering is defined as the task of grouping similar objects together in a group (cluster). Applications of clustering can be seen in business and marketing, where consumer populations are grouped into clusters for use in product development

or marketing strategies. Another common use is to group users of social networks to facilitate the deployment of advertisements to specific target groups.

Summarization is a process of compacting examples in a dataset to provide a condensed summary of the dataset. Applications of summarization are common in report generation and data visualization.

Dependency modeling is also known as association rule learning, where relationships between variables are identified. Applications of dependency modeling can be found in market basket analysis, where the objective is to discover products that are often bought together in supermarkets and to use this information for marketing and advertising purposes. Dependency modeling is also used in web usage mining to discover relationships between users and websites.

Change and deviation detection is the task of detecting changes in data in comparison with previously recorded data or standardize values. The task is typically applied with respect to network intrusion and credit card fraud detection.

The data mining task of interest in this thesis is that of classification; specifically, text classification. Section 2.2 therefore discusses classification in more detail, while Section 2.3 discusses text classification in particular.

2.2 Classification

As noted above, classification is the process of generalizing patterns from labelled examples and applying them to new examples. Essentially, it is the task of identifying a model with respect to a set of pre-defined class labels according to a set of labelled examples. As opposed to unsupervised tasks (for example, clustering), classification is a supervised task. In this context, supervised means that the class labels for the examples are pre-determined and known. There are two phases in the classification process; the learning phase and the classification phase. For the learning phase, the examples are divided into two parts that are mutually exclusive, called the training set and the test set. There are different ways in which the input dataset can be divided into training and test sets, for example, a 70:30% or a 50:50% split. Alternatively, Ten-fold Cross Validation (TCV) may be used. This latter approach splits the dataset into ten equal or almost equal sets and uses each set in turn as the test set while the other nine sets are combined and used as the training set. An average of the results is then taken. TCV is used in the evaluation of the classifier generated with respect to the work conducted in this thesis. This approach is adopted because it covers all parts of a dataset for the training and testing processes, thus providing an evaluation that is fairer. The training set is used to learn a classification model based on the patterns identified in the training set. The test set is then used to evaluate the resulting classifier

built from the training set. During the classification phase, the classifier generated in the learning phase is put "into service" to classify new and unseen data.

One of the issues in classification is the phenomenon of overfitting. Overfitting is defined as the learnt classifier being over-tailored to the training set such that it cannot performed well when classifying new and unseen datasets from the same domain. One of the ways to prevent the learnt classifier from overfitting to the training set is to include feature selection in the text classification process to select only a subset of features to be used for representation of the documents. In addition, the techniques for classification usually include mechanisms to further deter this phenomenon.

It is generally acknowledged that there are two types of classification formulation: (i) single-label classification and (ii) multi-label classification. Single-label classification refers to any classification task which assigns a single class label to examples, while in multi-label classification, each example can be labelled with more than one class label. Multi-label classification can be handled as multiple single-label classifications by dealing with each label individually. Therefore, solving the problem of the single-label classification also solves the problem of the multi-label classification [81]. There are a number of classification tasks for single-label classification. They include (i) binary classification, (ii) multi-class classification and (iii) hierarchical classification. In binary classification, the aim is to classify documents as either belonging to a class or not belonging to a class, in other words, a simple "yes-no" classification. In multi-class classification, the classifier is designed to assign one out of three or more class labels to each document. Hierarchical classification comprises the assignment of one or more class labels from a hierarchical class structure to each example. For example, a binary classifier for news articles might decide if an example belongs to class "Sports" or not, while a multi-class classifier might decide if an example belongs to the class "Finance", "Sports" or "Weather". A hierarchical classifier might decide if an example belongs to the class "Tennis", which is under the superclass "Sports". Multi-class classification can be handled as multiple cases of binary classification, by treating each class label individually using a binary classifier. With respect to the work conducted in this thesis, binary and multi-class classification will be considered. These will be further described in Section 4.4 in Chapter 4.

Classification has been applied to many forms of data. Classification algorithms were originally applied to tabular data but have since been adapted to classify examples using other data formats, including text, images and multimedia. The format of interest in this thesis is that of unstructured text and Section 2.3 will specifically discuss text classification, providing some background knowledge and information concerning previous research work done in the field.

2.3 Text Classification

Text classification is defined as the labelling of new and unseen text documents with a class label from a set of pre-defined labels based on the patterns learnt from a training set of labelled text documents. Being a mature field of research, there is much reported research in the application of text classification to different domains. Applications of text classification include, among others: web documents classification [68, 12], sentiment analysis [38, 25, 7], genre classification [32, 52, 29, 16] and email classification [23, 83, 14].

An important precursor to text classification is the preprocessing of the dataset used. The aim is to recast the dataset into a format that retains the salient features required for classification, while at the same time ensuring computational efficiency. There are many sub-processes involved in the preprocessing of the dataset used. These include data cleaning, keywords and/or phrase extraction, feature selection and text representation. The data cleaning sub-process includes removal of insignificant features such as stop words, symbols and numbers. Stop words are non-informative words such as articles, prepositions and conjunctions; whereas symbols and numbers are deemed insignificant in discriminating between classes in general text classification. Keyword and/or phrase extraction involves the extraction of significant words and/or phrases to represent the documents in a dataset. There are many techniques for extracting keywords and/or phrases and most are either statistical-based or natural language-based. Section 2.4 will discuss the use of keywords and phrases in text classification. Feature selection is a process of selecting a subset of words and/or phrases from a feature space, which normally consists of thousands of words and/or phrases extracted from the input dataset. Popular feature selection techniques include: Chi-Square (χ^2), Information Gain (IG), Document Frequency (DF), Odds Ratio (OR) and Mutual Information (MI). Section 2.5 will further consider the feature selection process.

The next step is to convert the input data into a format which is suitable for the adopted learning algorithm. The most popular format to represent text for classification is the bag-of-words (bag-of-phrases, if phrases are used) representation [26, 99, 75]. The bag-of-words representation is a simple but effective and widely used representation in text classification. In this representation, a text document is represented by a set of features in an n-dimensional feature vector space, where n is the number of features. The value of each feature in the feature vector space can take either a Boolean value to indicate the presence or absence of a feature in a document or a numeric value to indicate its frequency in a document. The order and position of the features in the documents are disregarded in this representation. The text representation used in this thesis is further described in Sub-section 4.2.5 in Chapter 4.

The learning of a classification model based on the patterns suggested in the training set will then be done using the desired learning algorithm. This will result in a text

classification model or simply, a text classifier. There are many learning algorithms that can be used for text classification. The most popular approaches are described in Section 2.6. The learnt text classifier will then be ready for use in the classification phase.

2.4 The Use of Keyword and Phrase in Text Classification

As already noted, the most common representation for text classification is the bag-of-words representation, which has been used widely in previous text classification research [57, 43]. In this representation, single keywords are selected from the dataset and used as the representation for the documents in the dataset. A keyword is defined as a word that is highly discriminative, i.e. can be used to distinguish between classes and selected from the collection of words from the documents in a dataset. The use of keywords as features for the text representation is fairly straightforward. The norm is to apply a feature selection technique to select a subset of words from the word collection to be used as keywords. These selected keywords are then used to represent the documents. Although the use of keywords is fairly effective in text classification, a lot of research has been directed at the development of richer representations than the bag-of-words. This has resulted in the the bag-of-phrases representation. The use of phrases for the text representation is motivated by the potential benefit of preserving semantic information in phrases that is not present in single keywords. There are various methods that may be adopted to identify and extract phrases for the bag-of-phrases representation. These methods tend to fall into two categories: linguistic phrase extraction [53, 54, 28] and statistical phrase extraction [34, 67, 11, 20]. The former is based on syntactic patterns while the latter is based on statistical patterns.

Previous work has reported on the use of phrases in text classification, albeit with mixed results. While some researchers reported better results with phrases, others claim that the use of phrases produced only marginal or zero improvement over the use of single keywords.

One of the earliest reports on research using phrases for the text representation in text classification is that of Lewis [54]. He studied the effects of the use of syntactic phrases in text classification and found that the use of noun phrases (in a Naive Bayes classifier) was less effective than individual words. The reason given for this was that not all phrases were good content indicators and that this affected the results when those phrases were used with better content indicators. Dumais et al. [28] extracted syntactic phrases in the form of factoids (for example, "Salomon_Brothers_International"), multi-word dictionary entries (for example, "New_York") and noun phrases (for example, "first_quarter") and reported no improvements on classification when using Naive Bayes and SVM classifiers. Based on the examples of phrases given, one could argue that factoids could be too "unique" (possibly infrequent) and could overfit the training data

and thus are not beneficial in this context. Multi-word dictionary entries and noun phrases could be similar in that they are proper two-word phrases which can both be common or unique. While common words could result in a classifier being over-generalized, unique words could cause the classifier to overfit the training data. In other words, the effectiveness of classification could be affected.

Fürnkranz [34] used the Apriori algorithm to generate n-gram features based on word sequences of length n and used RIPPER as its learning algorithm. He concluded that although there was slight improvement in including n-grams (up to 2-grams) for the text representation, word sequences of $n > 3$ were not useful and may decrease classification effectiveness. In addition, Fürnkranz et al. [35] investigated the use of linguistic phrases with both a Naive Bayes classifier (RAINBOW) and a rule-based classifier (RIPPER). Phrases were extracted using AUTOSLOG-TS, which used syntactic heuristics to create linguistic patterns. RAINBOW showed better performance when using phrases instead of words, while RIPPER showed worse performance when using phrases instead of words. Experimental results showed that the use of linguistic features could improve the precision of text classification, but at the expense of coverage. Although direct comparisons could not be made between these two pieces of research due to different experimental setups and the use of different datasets, it can be concluded that in both cases when RIPPER was used, the statistical phrases extracted in [34] could bring about a slight improvement over the case when linguistic features were used in [35].

Mladenić and Grobelnik [67] enriched their document representation by including n-grams of length up to five (5-grams) and used a Naive Bayes classifier for learning to classify the Yahoo text hierarchy. Their experiments showed that using word sequences of length up to three (3-grams), instead of using only single words, improved the classifier performance while longer sequences did not offer any benefits. This demonstrated that using statistical phrases could benefit text classification.

Scott and Matwin [80] extracted noun phrases and keyphrases and these were used with RIPPER for text classification. Noun phrases were extracted using the Noun Phrase Extractor (NoPE), which comprised a part-of-speech tagger and a regular expression algorithm to group tagged words into noun phrases. Keyphrases were extracted using a separate algorithm called the Extractor [86], which operated on an algorithm that mimicked the choice a human would make when selecting keyphrases. The use of noun phrases was found to be only slightly better than the use of keywords, while the use of keyphrases was found to be slightly worse. In general, the authors reported no significant benefit from using phrases and concluded that more complex natural language processing methods were needed to identify them. One could argue that in this case, the authors attempted to extract very "high level" phrases, in that the methods that they used extracted phrases that a human would choose to represent a class, i.e.

phrases that a human thought were semantically related to a particular class. While those phrases could be very good for text classification when a human performed the classification, they could be very statistically insignificant when a machine performed the classification.

Bakus and Kamel [5] extracted phrases using a statistical word association based grammar and a slight improvement over the use of the bag-of-words representation was reported using a Naive Bayes classifier. Although the extracted phrases were found to be good classification discriminators, the performance of classification was not significantly better than when using keyword feature. The authors pinned this down to two factors: (i) the number of extracted phrases was significantly less than the total number of extracted words and (ii) many phrases corresponded to the same keyword feature. It was suggested by the authors that these two reasons lessened the impact of phrases on the effectiveness of the classification.

In the work conducted by Kongovi et al. [49], they found instances where using phrases was more effective than when using single words. They reasoned that this was due to the fact that word pairs (two adjoining words) could provide some semantic value, as well as filter out words occurring frequently in isolation that are not discriminative. They defined a phrase as "two adjoining words in the text with zero word distance, eliminating all the stop words in between". Extracting phrases in this manner allowed patterns of co-occuring words to be extracted and statistical information concerning these words helped identify phrases that were good discriminator. Again, statistical phrases here were found to be beneficial as the text representation for text classification.

Tan et al. [84] extracted bigrams from the Reuters and Yahoo! Science datasets. Bigrams were extracted such that they contain at least one keyword; the keywords were selected based on a document frequency ranking and only highly ranked words that were deemed more significant than lower ranking ones were considered. The bigrams that were extracted were then further filtered by using TF-IDF and Information Gain ranking. Tan et al. used two classifiers, Naive Bayes and maximum entropy, and reported better classification results when bigrams were included in the representation. It was suggested by the authors that the improvement in classification results was due to a number of factors: (i) bigrams were used in addition to single words and not in place of; (ii) the number of bigrams selected was equivalent to 2% of the number of keywords and (iii) information gain was used in addition to document frequency and term frequency to choose bigrams, resulting in the bigrams being good discriminators.

Li et al. [59] reported that the use of phrases benefited classification when classifying texts about closely related topics in the same domain. They used an n-gram word extractor to extract frequent phrases and used them for classifying research paper abstracts using various classifiers. Experiments showed that the use of phrases was better than the use of keywords as the text representation for the classification of their

dataset. This was understandable because topics in the same domain may share a lot of technical terms. Therefore, phrases could serve as good discriminators to differentiate between classes. As given in the example by the authors: "text mining" and "data mining" were good phrases to differentiate between the classes "text mining" and "data mining" while "text", "data" and "mining" alone were common words shared by the two classes and thus less discriminating when used by themselves.

Chang and Poon [14] investigated the use of phrases for email classification using a Naive Bayes classifier and two k-nearest neighbour classifiers and found that using phrases of size two for the text representation gave the best classification results. Phrases were extracted using the Shingling algorithm [8] where contiguous sequence of words were extracted in an overlapping manner. The authors named the phrases w-shingles, where w was the length of the phrase. An example given by the author was "a rose is a rose is a rose" and the 4-shingles extracted from the example comprised {(a rose is a), (rose is a rose), (is a rose is), (a rose is a), (rose is a rose)}. They also experimented with removing "stop-shingles", which was a shingle containing only stop words. The authors commented that the removal of stop-shingles only had a very marginal effect on the classification of their dataset.

In general, the literature has reported various outcomes from the use of phrases in text classification. While some results are promising, others reported very small or no improvements in classification effectiveness. It is clear that direct comparison cannot be made between the different bodies of research work because of the different experimental setups that were used. Both linguistic and statistical phrase extraction had been experimented with and previous works has reported more favourable results for statistical phrase extraction. With respect to the work conducted in this thesis, three different kinds of statistical phrases are extracted as reported in Sub-section 4.2.3 in Chapter 4.

2.5 Techniques for Feature Selection

One of the important process in data preprocessing is the feature selection process. The vast amount of documents in a text dataset means that the amount of features that may be identified can be overwhelming. Feature selection therefore plays the role of reducing the dimensionality of the feature space by selecting a subset of features instead of using the entire feature space. One advantage of feature selection in text classification is that it can reduce the risk of overfitting as already noted, an occurrence where the induced classifier is tuned to the training data and will perform badly at classifying new and unseen data.

There are two approaches to feature selection: *local* feature selection and *global* feature selection. Features are usually ordered according to their discriminative values, usually calculated based on the mathematical formula associated with the feature selec-

tion technique used. In local feature selection, features that are local to a specific class are selected for learning. This means that a different subset of features are selected for each different class in a dataset. Global feature selection involves the selection of features from across all classes in a dataset. The maximum or weighted-average value of each feature's class-specific value is used to order the features for selection. One notable advantage of local feature selection is that the number of features to be dealt with at a time is confined to features that occur locally in a class. Each class is dealt with one at a time. Therefore, the number of features is comparably much less than in global feature selection, where features from all classes are pooled together. Using the local feature selection approach also ensures that each class is represented by a subset of features in a fixed proportion. In global feature selection, a subset of features is selected from the feature space which comprises collective features from all the classes in a dataset. The selected subset of features could under-represent some classes, particularly smaller classes, as features from those classes may not be selected. As with most previous researches [54, 100, 64], local feature selection is used in the experiments described in this thesis.

Many feature selection techniques have been implemented and used in text classification. Among them are χ^2 [37, 76, 64, 65], IG [11, 68], MI [68, 65], OR [11, 68, 65], DF [95, 11, 68] and many others. Previous work [95, 76] found that that χ^2 and IG were more efficient. Therefore, these two techniques have been adopted with respect to the work described in this thesis and are further described in Sub-sections 2.5.1 and 2.5.2 below.

2.5.1 Chi Square

Chi Square (χ^2) is a feature selection technique used widely in text classification. The formula for χ^2, extracted from [81], is given as follows:

$$\chi^2(t_k, c_i) = \frac{\mid Tr \mid \cdot [P(t_k, c_i) \cdot P(\bar{t_k}, \bar{c_i}) - P(t_k, \bar{c_i}) \cdot P(\bar{t_k}, c_i)]^2}{P(t_k) \cdot P(\bar{t_k}) \cdot P(c_i) \cdot P(\bar{c_i})} \tag{2.1}$$

where:

- $\mid Tr \mid$ is the number of documents in the training set;
- $P(t_k, c_i)$ represents the probability that term t_k occurs in document d and that d belongs to class c_i;
- $P(\bar{t_k}, c_i)$ represents the probability that term t_k does not occur in document d and that d belongs to class c_i;
- $P(t_k, \bar{c_i})$ represents the probability that term t_k occurs in document d and that d does not belong to class c_i;

- $P(\bar{t_k}, \bar{c_i})$ represents the probability that term t_k does not occur in document d and that d does not belong to class c_i;
- $P(t_k)$ represents the probability that term t_k in document d occurs in the training set.
- $P(\bar{t_k})$ = 1 - $P(t_k)$;
- $P(c_i)$ represents the probability that document d belongs to class c_i in the training set;
- $P(\bar{c_i})$ = 1 - $P(c_i)$.

In the context of text classification, χ^2 is a measure of the degree of dependence between a term and a class. A term with a smaller χ^2 value is more independent of a class, while a bigger value shows otherwise. Therefore, a term that has a bigger χ^2 value is more discriminative. Discriminative terms, identified by sorting all the terms in a class in descending order by their χ^2 values are selected based on a pre-determined percentage or a threshold of top ranking terms and these terms become features that represent documents in a class.

2.5.2 Information Gain

Information Gain (IG) is a feature selection technique that takes into consideration the presence and absence of a term in a class. In the field of machine learning, IG is usually used as a "term-goodness" criterion. The number of bits of information is used as a measure for the prediction of a class by using knowledge of the presence and absence of a term. The amount of information that a term contains about a class is calculated based on the following formula extracted from [81]:

$$IG(t_k, c_i) = P(t_k, c_i) \cdot \log \frac{P(t_k, c_i)}{P(c_i) \cdot P(t_k)} + P(t_k, c_i) \cdot \log \frac{P(\bar{t_k}, c_i)}{P(c_i) \cdot P(\bar{t_k})} \tag{2.2}$$

where,

- $P(t_k, c_i)$ represents the probability that term t_k occurs in document d and that d belongs to class c_i;
- $P(\bar{t_k}, c_i)$ represents the probability that term t_k does not occur in document d and that d belongs to class c_i;
- $P(c_i)$ represents the probability that document d belongs to class c_i in the training set;
- $P(t_k)$ represents the probability that term t_k in document d occurs in the training set;

- $P(\bar{t_k}) = 1 - P(t_k)$.

In text classification, a term with a bigger value of IG is deemed to be more informative. All the terms in a class are sorted in descending order by their IG values and a pre-determined percentage of the terms or a threshold of the top ranking terms are selected as features to represent documents in a class.

2.6 Techniques for Text Classification

Many techniques have been developed over the years to handle text classification. One of the earliest approaches to text classification is the knowledge engineering approach. In this approach, human experts manually build a knowledge-based system. The CONSTRUE system [41], an automated news story classification system developed by the Carnegie Group for the Reuters dataset, was built using a rule-based knowledge engineering approach. The rules generated, in the form of "if <DNF Boolean formula> then <class>", placed a document in a class if the document satisfied the Disjunctive Normal Form (DNF) Boolean formula. The noted disadvantage of this approach to text classification is the infeasibility of upgrading the system, which will require a substantial human expert resource. Perhaps the most popular approach to text classification is the supervised machine learning approach, whereby a learner automatically constructs a classifier by learning from a set of pre-labelled documents. The advantage of using the machine learning approach for text classification is that it is faster and more feasible compared to the manual labelling of documents by human experts.

Many machine learning techniques have been applied to text classification since the 1990s [81]. These include: support vector machines (SVMs) [28, 43, 94, 62, 99, 29, 98]; probabilistic Naive Bayes (NB) [57, 28, 63, 94, 27, 14]; classification association rule mining (CARM) [2, 21, 90]; inductive rule learning (IRL) [91, 4, 22]; decision trees (DT) [71, 72, 57, 28, 42]; neural networks (NN) [24, 82, 29, 98]; k-nearest neighbour (k-NN) methods [94, 75, 14] and many more. The techniques used in the experiments described in this thesis for comparison with the proposed IRL mechanism are SVM, NB, CARM, IRL and IRL with negation. Therefore, these techniques are further described in Sub-sections 2.6.1, 2.6.2, 2.6.3, 2.6.4 and 2.6.5 respectively.

2.6.1 Support Vector Machine

A Support Vector Machine (SVM) is a supervised learning technique that was first introduced by Vapnik [87] for solving two-class pattern recognition problems. It is based on the Structural Risk Minimization principle from computational learning theory. It operates on the notion of a hyperplane that separates two classes. The SVM algorithm constructs a model that assigns a new document to either one of the two classes. The aim is to find the optimal separating hyperplane such that the widest possible margin

between the two classes is obtained. Figure 2.2 shows an ideal case of two classes, one class with instances represented by circles and another represented by triangles. In this ideal example, the classes are linearly separable and the SVM algorithm will find a one-dimensional hyperplane to separate the two classes, as shown by the solid line in the figure. This solid line is just one of the infinite number of possible lines that may be used to separate the two classes. The dashed lines parallel on either side of the hyperplane, represent the distance between the closest vectors and the hyperplane. These vectors are the "support vectors" and the distance between the dashed lines is the margin. The aim is to identify the support vectors that produce the widest possible margin.

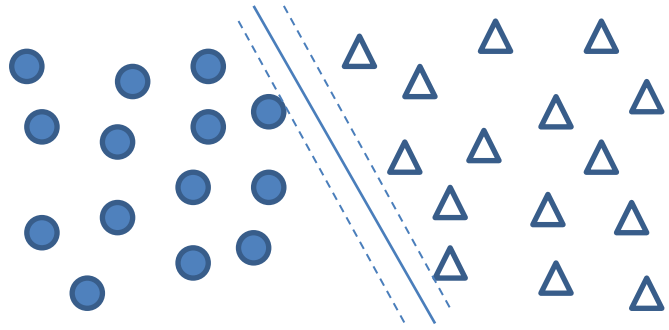

Figure 2.2: An example of how SVM works

SVMs are widely used as seen in the text classification literature. Radovanović and Ivanović [75] made use of a SVM and four other learning techniques to do an extensive study of the bag-of-words representation in short web-page descriptions. Kolcz and Chowdhury [48] proposed a technique to avoid model re-induction in SVM-based feature selection for text classification. Joachims, in particular, did much work on SVMs for text classification. Among others, his work included: standard SVMs for text classification [43]; transductive SVMs for text classification [44]; an approach for estimating the generalization performance of a SVM for text classification [45] and the training of linear SVMs in linear time [46].

SVM is one of the most successful techniques ever applied to text classification. The experiments conducted for this thesis therefore include comparison with the Sequential Minimal Optimization (SMO) algorithm, a SVM algorithm provided in the Waikato Environment for Knowledge Analysis (WEKA) machine learning workbench [92].

2.6.2 Naive Bayes

The Naive Bayes (NB) classifier is based on the Bayes Theorem. The Bayes Theorem is shown in Equation 2.3, where H is defined as the hypothesis and D refers to the input document that bears on H.

$$P(H \mid D) = \frac{P(D \mid H)P(H)}{P(D)} \tag{2.3}$$

where:

- P(H) is the probability of hypothesis H being correct;
- P(D) is the probability of document D being observed;
- P(D|H) is the probability of observing document D under the assumption of H being correct;
- P(H|D) is the probability that hypothesis H is correct, where document D is observed.

In the context of text classification, given the hypothesis H that "the input document D belong to class C", P(H|D) then refers to the probability that the input document D belongs to class C. The NB classifier works based on the "Naive" assumption that, given a particular class C, the presence or absence of a feature is independent from the presence or absence of all other features.

The NB classifier has been widely used in text classification. Langley et al. [51] implemented an NB classifier and compared it with a decision tree learner, C4.5, and found that the NB classifier performed as well or better than C4.5 for four out of five UCI Machine Learning Repository datasets [69]. Chai et al. [13] implemented Bayesian online classifiers and found that they performed comparably to SVM, noted in Subsection 2.6.1 as being one of the best learning algorithms for text classification. In personal email filtering, Diao et al. [27] implemented a NB classifier and a decision tree classifier and found that for optimal parameter settings, the decision tree classifier performed better than the NB classifier but that the NB classifier was more robust. Calado et al. [10] also used the NB classifier in their work to include link-based information for web document classification. Radovanović, and Ivanović [76] used both a NB classifier and a SVM classifier to study the interaction between document representation and feature selection.

Previous work has shown wide and successful usage of the NB classifier in text classification. Thus, the NB classifier implemented in the WEKA machine learning workbench was adopted for use with respect to the work described in this thesis for comparison with the proposed IRL mechanism.

2.6.3 Classification Association Rule Mining

Association Rule Mining (ARM) is a technique to extract association rules from a transactional database, first introduced in Agrawal et al. [1]. ARM is defined as follows:

- D_T is a transactional database;
- $I = \{a_1, a_2, ... , a_n\}$ is a set of binary-valued database attributes called items;
- $T = \{t_1, t_2, ... , t_m\}$ is a set of database records called transactions;
- D_T is described by T, where $t_i \in T$ comprises a set of items $I' \subseteq I$.

An association rule describes the co-occurrence relationship between two sets of items in D_T and is expressed as $X \Rightarrow Y$, where X, $Y \subseteq I$ and $X \cap Y = \emptyset$. The quality of an association rule is typically measured by using the support and confidence framework. The support and confidence measures are defined as follows:

1. **Support:** The support of an itemset is used to determine if an itemset is frequent. If the support value of an itemset is more than a pre-determined threshold σ, then the itemset is said to be frequent.
2. **Confidence:** The confidence value is used to determine how strongly X implies Y in an association rule of the form $X \Rightarrow Y$. A pre-determined threshold α is used to filter high confidence association rules from low confidence association rules.

Equation 2.4 and 2.5 are used to compute the support and confidence values respectively.

$$support(X \cup Y) = \frac{count(X \cup Y)}{\mid T \mid} \tag{2.4}$$

$$confidence(X \Rightarrow Y) = \frac{support(X \cup Y)}{support(X)} \tag{2.5}$$

The support and confidence framework with pre-defined thresholds (σ and α respectively) is used to identify frequent itemsets. These frequent itemsets are then used to generate association rules. The most frequently cited ARM algorithm is the *Apriori* algorithm, introduced by Agrawal and Srikant [2] and subsequently used to form the basis of other ARM algorithms. In the *Apriori* algorithm, frequent itemsets are iteratively identified by using the "downward closure property" of itemsets, where an itemset is considered frequent if and only if all its subsets are identified as frequent in the previous pass. Algorithm 1 shows the *Apriori* algorithm for identifying frequent itemsets.

Classification Association Rule Mining (CARM) is the use of ARM algorithms to induce rules for use in classification tasks. ARM algorithms are employed to extract classification association rules from transactional databases with binary features. A

Algorithm 1: Apriori algorithm for identifying frequent itemsets

input : $I_k \in D_T$ and minimum support threshold σ
output: S, a set of frequent itemsets

```
k ← 1;
S ← an empty set to hold frequent itemsets;
generate I_k ∈ D_T;
while I_k ≠ ∅ do
    for all I_k ∈ D_T do
        determine the support for I_k ∈ D_T;
        if support for I_k ≥ σ then
            Store I_k in S;
        end
        else
            remove I_k;
        end
    end
    generate I_{k+1} ∈ D_T;
    k = k + 1;
end
return S;
```

classification association rule describes the association between a set of binary feature-value pair and a class feature. Therefore, in terms of the association rule $X \Rightarrow Y$, X is some subset of binary feature-value pairs, while Y is the class feature.

The use of CARM techniques has been reported by a number of authors [58, 19, 21, 89, 97]. Popular techniques include: Classification Based on Associations (CBA) [60]; Classification based on Multiple Association Rules (CMAR) [58]; Classification based on Predictive Association Rules (CPAR) [96] and Total From Partial Classification (TFPC) [89].

The TFPC algorithm will be used as a technique for comparison with the proposed IRL mechanism in the multi-class classification task in this thesis as it is a multi-class text classification system. It has one keyword selection strategy and four phrase selection strategies built into its system. The keyword selection strategy is based on selecting words that exceed a minimum threshold for a user-defined contribution value, which is a measure of how discriminative a word is. The four phrase selection strategies, on the other hand, are based on the notion of noise words, significant words, ordinary words and stop marks. They are defined as follows:

- **Noise words** - Common and rare words which occur in the documents in the dataset.
- **Significant words** - Selected keywords which are used to differentiate between classes.

- **Ordinary words** - Other non-noise words which are not selected as significant words.
- **Stop marks** - Punctuation marks comprising {, . : ; ! and ?}

The four phrase selection strategies are then derived as follows:

1. DelSN_contGO - These phrases are delimited by stop marks and/or noise words and are made up of sequences of one or more significant words and ordinary words.
2. DelSN_contGW - These phrases are delimited by stop marks and/or noise words and are made up of sequences of one or more significant words and "wild card" words, which can be matched to any single word.
3. DelSO_contGN - These phrases are delimited by stop marks and/or ordinary words and are made up of sequences of one or more significant words and noise words.
4. DelSO_contGW - These phrases are delimited by stop marks and/or ordinary words and are made up of sequences of one or more significant words and "wild card" words, which can be matched to any single word.

2.6.4 Inductive Rule Learning

Inductive Rule Learning (IRL) is a widely used technique for text classification. Rule-based classifiers offer an advantage over "blackbox" style classifiers, such as SVMs and NB, in that they are easily interpretable and simple to apply (an advantage also shared by some other methods, for example, decision tree classifiers). This advantage is of importance, particularly in applications where manual human intervention is essential in analyzing the classifiers, for example, in the medical domain. The IRL technique (without featuring the use of negation) for text classification is a mature research field. Numerous algorithms, mostly based on the *covering* algorithm, have been implemented and used in text classification. The proposed IRL mechanism in this thesis will also adopt the *covering* algorithm. In the *covering* algorithm, rules are "learned" sequentially based on the documents in the training set. The "covered" documents are then removed and the process is repeated until all the documents are covered or no more rules can be generated. The rules that are generated are added into the ruleset. The ruleset is then the set of rules that is used as a classifier. The *covering* algorithm is shown in Algorithm 2.

The goodness of a rule learnt is usually measured by its accuracy. To measure how accurate a rule is, the number of documents correctly covered by the rule is taken into consideration. A rule may cover both documents from the class at which the rule is

Algorithm 2: *Covering* algorithm

```
input  : D, a dataset of class-labelled documents,
         Feature_set_c, the set of features for class c
output: Ruleset, a set of rules
Ruleset = { }; //initial set of rules learned is empty;
for each class c do
    while stopping condition is not met do
        Rule = LearnOneRule(Feature_set_c, D, c);
        Remove documents covered by Rule from D;
        Ruleset_c = Ruleset_c + Rule;
    end
    Ruleset = Ruleset + Ruleset_c;
end
return Ruleset;
```

directed (positive documents) and documents from other classes (negative documents). The formula for the accuracy of a rule is given in Equation 2.6:

$$Accuracy = \frac{P}{P+N} \tag{2.6}$$

where P is the number of positive documents covered and N is the number of negative documents covered.

The use of accuracy as a measure for rule quality however, can be a little misleading. Say, for example, referring to Table 2.1, there are two rules: Rule 1 covers only one positive document and has a higher accuracy because it does not cover any negative documents, in other words, 100% accuracy. Rule 2 however, covers 100 positive documents and two negative documents. Although one can argue that Rule 2 is a much better rule as it covers a lot more positive documents, the rule's accuracy is dragged down by the coverage of two negative documents. The comparison of these two rules using the accuracy measure is therefore misleading. A better measure that can give a fairer evaluation of the rule quality is the accuracy with Laplace estimation. The formula for calculating rule accuracy with Laplace estimation is given in Equation 2.7.

Rule	P	N	Accuracy
1	1	0	100%
2	100	2	98%

Table 2.1: Examples of rule accuracy

$$AccuracywithLaplaceestimation = \frac{P+1}{P+N+numberOfClasses} \tag{2.7}$$

The accuracy of the rules in Table 2.1, using Laplace estimation and assuming that there are 10 classes in the dataset, is shown in Table 2.2. With the inclusion of Laplace estimation in the accuracy measure, the quality of a rule can now be more fairly reflected. Rule 2 now has a higher accuracy than Rule 1.

Rule	P	N	Accuracy with Laplace estimation
1	1	0	18.2%
2	100	2	90.2%

Table 2.2: Examples of rule accuracy with Laplace estimation

The rules in an IRL ruleset are usually ordered so that they can be fired according to their priority. Two rule ordering strategies are described by Han and Kamber [40]; namely, class-based ordering and rule-based ordering. For class-based ordering, the classes are sorted by their prevalence. This means that the rules for the most frequent class will rank at the top, followed by the next frequent class and so on. On the other hand, rule-based ordering sorts the rules according to the rule quality, taking measures like rule accuracy, coverage or length as the ordering priority. Ordering by accuracy usually ranks the higher accuracy rules first. This means that more accurate rules will be fired first. Ordering by coverage will cause rules with larger coverage (covering more documents) being ranked higher. Length ordering orders the rules according to the antecedent length. The more features in the antecedent, the longer the rule and the higher the ranking. This means that more specific rules are ordered first.

Much previous research into IRL has been applied to the text classification task. Perhaps the most popular of the IRL algorithms is the Repeated Incremental Pruning to Produce Error Reduction (RIPPER) [22]. This algorithm evolved from the Reduced Error Pruning (REP) algorithm [9]. An extension to RIPPER was proposed by Vasile et al. [88]. The evolution of REP to RIPPER is described below.

REP is an algorithm for decision tree pruning, which can be adapted to rule learning systems [9]. The training data used for REP for rule learning is split into a growing set and a pruning set. An initial ruleset that overfits the growing set is generated and then pruned based on the pruning set using pruning operators that yields the greatest reduction of error. Pruning will stop when the application of any further pruning would increase the error with respect to the pruning set. Though REP can work rather well for noisy data, Fürnkranz and Widmer [36] outline some problems they found in REP; the main problem being its efficiency on large datasets. They proposed a rule-learning algorithm called Incremental Reduced Error Pruning (IREP) to address the problems in REP.

IREP integrates pre-pruning and post-pruning into the learning process. After a rule is learned from the growing set, pruning is immediately done in a greedy fashion

until the accuracy of the rule no longer improves on the pruning set. The rule is then added to the ruleset and all the covered examples in both the growing and pruning set are removed. The remaining examples will then be split into the growing and pruning set and another new rule is learned in the same manner. The process is repeated until the predictive accuracy of the pruned rule is worse than the empty rule. IREP was shown to be more efficient than REP with a slight gain in accuracy. However, it was found that IREP did not perform well for domains with a very specific concept description [36].

Cohen made some improvements to IREP and came up with an algorithm called the Repeated Incremental Pruning to Produce Error Reduction (RIPPER) [22] . RIPPER was the result of three modifications to IREP. The first modification was an alternative rule value metric for pruning. Next, a new MDL-based heuristic to decide when to stop adding rules to the ruleset was devised. These two modifications improved IREP's generalization performance and was referred to as the modified IREP (IREP*). The third modification was a postpass to optimize the ruleset, which was produced by IREP*. These three modifications resulted in the formulation of RIPPER. The author reported that RIPPER was comparable to C4.5Rules [73] in terms of error rates, but was more efficient in dealing with large datasets. RIPPER is thus an IRL system which uses the *covering* algorithm to learn rules. RIPPER generates rules by greedily adding features to a rule until the rule achieves a 100% accuracy. This process tries every possible value of each feature and chooses the one with the highest information gain. Following this rule building phase, a rule pruning phase is applied, whereby the generated rule is pruned using a pruning metric.

Vasile et al. [88] subsequently extended RIPPER to include external knowledge in the form of taxonomies for IRL. Their work made used of WordNet [66], an online lexical reference system built by the Cognitive Science Laboratory at Princeton University. This extended rule induction algorithm is called Taxonomical RIPPER (TRIPPER). External knowledge was used both in the rule generation and rule pruning stages. In the rule generation stage, a process called feature space augmentation was introduced. The augmented set of features was obtained based on taxonomies defined over the values of the original features. In the rule pruning stage, pruning was replaced with taxonomy-guided abstraction, where different levels of specificity can be chosen for a feature under consideration. The results obtained from experiments on the ten biggest classes in the Reuters 21578 dataset showed that TRIPPER outperformed RIPPER in eight out of ten classes in terms of the precision and recall break-even point. TRIPPER was also able to generate rules which were generally more comprehensible than RIPPER.

Another IRL algorithm is called Swap-1 [91]. Apté et al. [4] made used of Swap-1 to induce rules for text classification. In Swap-1, a covering set of rules is obtained through a heuristic search to find a single best rule that covers only one class. The

rule is then added into the ruleset and the examples covered are removed. This process is repeated until there are no more examples to be covered. Swap-1 uses local optimization techniques to dynamically improve the ruleset. Pruning is done progressively to decrease the complexity of the ruleset. A method of pruning, called weakest-link pruning, is done to obtain a series of rulesets in decreasing order of complexity. The best ruleset is the one that results in the lowest observed true error rate with respect to a test dataset.

RIPPER is regarded as one of the most successful IRL algorithms. Therefore, it was used by many other researchers as a benchmark for comparison or for testing other elements of text classification [35, 80, 85, 79]. JRip, which is the WEKA's implementation of RIPPER, will be used in this thesis as a technique for comparison with the proposed IRL mechanism.

2.6.5 Inductive Rule Learning with Negation

The IRL methods described in Sub-section 2.6.4 do not explicitly advocate any usage of negated features in the learning of classification rules and typically employed a two-stage process whereby rules were first learnt and then pruned to improve the effectiveness of the ruleset. On the surface, it does not look like RIPPER includes any mechanism for explicitly generating rules with negation. However, in the case of binary-valued features, a feature value of zero (0) can be interpreted as a negated feature (the absence of a feature). For example, a rule $a = 1 \wedge b = 1 \wedge c = 0 \Rightarrow x$ could be interpreted in a similar manner to $a \wedge b \wedge \neg c \Rightarrow x$. Both rules are interpreted as, "If a and b occur in a document and c does not occur, then classify the document as class x". However, the disadvantage of RIPPER in implicitly including negation is that every possible value of each feature is tried until the highest information gain is obtained. This doubles the number of features in the feature space in the case of binary-valued features. In the context of text classification, the number of features is an issue; doubling the number of features will therefore be undesirable.

In IRL, apart from RIPPER which implicitly includes negation when using the binary representation, another system that includes negation in IRL is the Olex system. The Olex system was developed by Rullo et al. [78] and is founded on the idea of using a fixed template that allows only one positive feature and zero or more negative features to generate rules. There are two versions of the Olex system. They include OlexGreedy [78] and OlexGA [70]. OlexGreedy, as the name suggests, uses a "greedy", single stage, rule learning process. A disadvantage of OlexGreedy, highlighted by the authors, is that the template approach is not able to express co-occurrences based on feature dependencies by allowing just one positive feature in the rule antecedent. Rullo et al. [79] attempted to overcome this disadvantage by using conjunctions of terms (coterms), where conjunction of positive features could be included in the rules

generated. However, the authors again reported that rules that were generated using the improved version could not share common features in the rule antecedent. This meant that rules having the same positive feature in the rule antecedent could not be generated by OlexGreedy. For example, OlexGreedy was not able to generate the following rules [79]:

$$wheat \wedge farm \Rightarrow wheat$$
$$wheat \wedge commodity \Rightarrow wheat$$
$$wheat \wedge agriculture \Rightarrow wheat$$

where, the feature "wheat" occurred positively in more than one rule.

Hence, OlexGA was proposed, which used a genetic algorithm to induce a rule-based classifier. This version overcame the problems associated with OlexGreedy. However, the generated rules still adhered to the fixed template of "one positive feature (or coterm), zero or more negative feature(s) (or coterm)". Therefore, a disadvantage of the Olex systems is that the use of such templates is somewhat restrictive in that rules with flexible combinations of positive and negated features could not be generated. Despite that, the Olex systems performed better than other techniques such as C4.5, SVM, RIPPER and NB.

A number of alternative methods for incorporating negated features into text classification have been reported in the literature, though not all of these methods are intended for IRL. Antonie and Zaïane [3] and Wu et al. [93] used both positive and negative association rules in their work on classification association rule mining. Galvotti et al. [37] used a novel variant of k-nearest neighbour with "negative evidence". Zheng and Srihari [101] combined positive and negative features in their feature selection method for text classification and used a Naive Bayes classifier. Baralis and Garza [6] used negated words in their associative classifier for text classification.

The use of negation in previous work, both non-IRL and IRL-based, has served to provide empirical evidence of the effectiveness of the incorporation of negation for text classification. This provides further motivation for the investigation into the use of negation in IRL. In particular, this thesis aims to incorporate a method to dynamically include negated features when needed rather than negating all the features in the feature space (as done in RIPPER). In addition, the proposed IRL mechanism also aims to be able to generate rules which are not based on a fixed template like that of Olex, through the use of various rule refinement strategies. Both OlexGreedy and OlexGA will be used as techniques with which the proposed IRL mechanism can be compared.

2.7 Evaluation Measures

The primary aim of this thesis is to establish whether the use of negation in IRL is effective for the text classification task, experimenting with both the keyword and phrase

representations. Therefore, a number of evaluation measures were used to determine if the research question posed in Section 1.2 in Chapter 1 was answered. For the binary classification task, a contingency table (also known as a confusion matrix) [92] was used to evaluate the classifiers generated for each class. An example is given in Table 2.3 where TP, FP, FN and TN represent the number of true positive, false positive, false negative and true negative results respectively. True positives and true negatives are documents that have been correctly classified. False positives are documents that have been wrongly classified as positive when they are actually negative, while false negatives are documents that have been wrongly classified as negative when they are actually positive.

Class c		Classifier	
		Yes	No
Expert	Yes	TP	FN
	No	FP	TN

Table 2.3: Confusion matrix for the binary classification task

From the confusion matrix, performance measures that can be computed include precision (P), recall (R), accuracy (Acc) and the F_1-measure (F_1).

Precision: Is defined as the ratio of the number of documents correctly classified as positive to the total number of documents that have been identified as positive. It is given as:

$$P = \frac{TP}{TP + FP} \tag{2.8}$$

Recall: Is defined as the ratio of the number of documents correctly classified as positive to the total number of documents that belong to the positive class. It is given as:

$$R = \frac{TP}{TP + FN} \tag{2.9}$$

Accuracy: Is defined as the ratio of the number of documents correctly classified, as either positive or negative, to the total number of documents. It is given as:

$$Acc = \frac{TP + TN}{TP + FP + FN + TN} \tag{2.10}$$

F_1-measure: Is the combination of both precision and recall to obtain a single value to measure the performance of a classifier. It is given as:

$$F_1 = \frac{2PR}{P + R} \tag{2.11}$$

In binary classification, precision, recall and the F_1-measure are common measures used to evaluate the classifier performance. The use of accuracy is not a desirable option because it does not take into account the problem of unbalanced class distributions. As such, a high accuracy value can be obtained just by classifying the examples for the majority class, but this will not signify that the classifier is a good classifier. This issue is particularly relevant in the case of binary classification where only two classes are considered. Therefore, the preferred evaluation measure used in this thesis for binary classification is the F_1-measure, whose calculation includes both precision and recall. Another measure worth mentioning, but not used in this thesis, is the Receiver Operating Characteristic (ROC) curve [92]. The curve plots the true positive rate (recall) against the false positive rate (1 - recall) for the classifier used. The area under the curve (AUC) can then be used as an evaluation measure for the classifier. In the case of multi-class classification, with respect to this thesis, the accuracy measure is used to evaluate the performance of the classifier because the problem of unbalanced class distributions is not as prevalent. In addition, the accuracy measure was used in the TFPC algorithm, which was used as the technique for comparison with the proposed IRL mechanism with respect to the multi-class classification task. In both binary and multi-class classification, ten fold cross validation was employed so as to obtain an even average performance and prevent biased results. For evaluating overall performance across all the classes in a dataset, micro-averaging was used. Micro-averaged scores are obtained by first summing up all the corresponding cells in the confusion matrix for all the classes and then computing the values from the global confusion matrix. Micro-averaged scores give equal weighting to every document in the dataset.

In this thesis, the performance measures described above are used to evaluate the effectiveness of the proposed IRL mechanism and compare this performance with that of other machine learning techniques.

2.8 Summary

This chapter has provided the background and related work for text classification. A general description of data mining was given, followed by the classification task. Text classification was then described in detail, providing the background information and related work on the use of keywords and phrases, feature selection techniques and techniques for text classification. Reviews of previous work have shown that the use of phrases in text classification is able to improve classification results. In particular, statistical phrases have been shown to be better than linguistic phrases in most cases. Therefore, the work in this thesis will include the use of statistical phrases, in addition to keywords, as features in the text representation. χ^2 and IG are two feature selection techniques which have been used extensively in previous work to reduce the dimensionality of the feature space in text classification. These techniques have been described

in this chapter and were used, as will be described later in this thesis. Many techniques for text classification have been described in the literature. In particular, SMO, NB, JRip, OlexGreedy, OlexGA and the TFPC algorithm were used for the comparative study reported in this thesis. SMO was chosen to be included because SVM algorithms have been shown to be one of the best text classification techniques and therefore, can be used as a benchmark in this study. NB was included as it is a widely used technique in text classification. JRip, OlexGreedy, OlexGA and the TFPC algorithm are all rule-based techniques and therefore, included for comparison with the proposed IRL mechanism, which is also a rule-based technique.

Chapter 3

The Case for Negation in Inductive Rule Learning

Rules in the form of *antecedent* $\Rightarrow$ *conclusion* are the common output of Inductive Rule Learning (IRL) algorithms. The antecedent part of a rule is usually a conjunction of positive features, where positive features are defined as items that exist in the records (documents in the case of text classification) associated with the class for which a rule is being learnt. Rules in this form are often sufficient for effective classification. However, there are cases where rules with negation are required in order to form a more effective classifier. Intuitively, a classifier consisting of both rules with and without negation should be more effective with respect to classification than one that does not include negation. However, are there situations that can only be "best" classified using classifiers that include the negation of features, or is it always possible to achieve the "best" classification using rules that comprise only positive features?

This chapter firstly sets out to establish that there are indeed situations that can only be resolved using rules that include the negation of features in the antecedent. This is demonstrated using a collection of synthetically generated datasets. The generation of this collection of datasets is described in Section 3.1. Section 3.2 describes the evaluation of the experiments conducted to establish the need for IRL with negation. This includes Sub-sections 3.2.1, 3.2.2 and 3.2.3, which highlight three different scenarios that may occur when learning rules from the synthetic datasets. Section 3.3 summarizes the chapter.

3.1 Experimental Setup to Establish Whether Inductive Rule Learning with Negation is a Necessity or Not

This section describes the experimental setup used to establish whether the inclusion of negation when generating rule-based classifiers is a necessity. The experiment was conducted using a collection of synthetic datasets. These datasets were generated by considering every possible combination of the set of features, $A = \{a, b, c\}$, and the

set of classes, $C = \{x, y, z\}$, within a three record dataset ($N = 3$). The number of possible combinations of a set of features A is given by $2^{|A|} - 1$ (minus one to remove the null set). Thus, in the example considered here, the number of possible combinations is $2^3 - 1 = 7$. Assuming that the consequent is always a single class feature, there are $7 \times 3 = 21$ possible combinations for each record. Table 3.1 shows the 21 possible combinations of features and classes for each record. Given that $N = 3$, the number of different three record datasets that can be generated is $21^3 = 9,261$.

Possible combinations of features and classes
{a} - class x
{b} - class x
{c} - class x
{a, b} - class x
{a, c} - class x
{b, c} - class x
{a, b, c} - class x
{a} - class y
{b} - class y
{c} - class y
{a, b} - class y
{a, c} - class y
{b, c} - class y
{a, b, c} - class y
{a} - class z
{b} - class z
{c} - class z
{a, b} - class z
{a, c} - class z
{b, c} - class z
{a, b, c} - class z

Table 3.1: Possible combinations of $A = \{a, b, c\}$ and $C = \{x, y, z\}$.

For the experiments conducted, a standard *covering* algorithm (see Algorithm 2 in Chapter 2) was applied to the synthetic datasets to produce classifiers. Each classifier was then tested on the same input data used to generate it and the accuracy noted. Two variations of the *covering* algorithm were used: the standard variation without the use of negation, and an alternative that incorporated a simple form of negation (strategies for incorporating negation into the IRL process are described in Section 4.3 in Chapter 4).

3.2 Evaluation of the Experiments to Establish Whether Inductive Rule Learning with Negation is a Necessity or Not

This section reports on the evaluation of the experiment described in Section 3.1. An overview of the results obtained is presented in Table 3.2. The initial experiment showed that when negation is included in the IRL process, many more datasets that are accurately classified can be obtained. Note that the experiment and its analysis have been reported previously in [15]. Table 3.3 gives a summary of the results obtained using the synthetic datasets.

Accuracy on training set	Number of datasets	
	IRL without negation	IRL with negation
100% (3 of 3 records correctly classified)	4,503	6,825
67% (2 of 3 records correctly classified)	3,324	2,316
33% (1 of 3 records correctly classified)	1,434	120
Total	9,261	9,261

Table 3.2: Results for the experiment using synthetic datasets

From Table 3.3, it can be seen that 4,503 of the synthetic datasets are accurately classified using standard IRL. This figure rises to 6,825 when negation is introduced into the IRL process. This is a 25.1% increase in the number of datasets accurately classified.

IRL	Number of datasets accurately classified	% of total
Without negation	4,503	48.6
With negation	6,825	73.7

Table 3.3: Summary of results for the experiment using the synthetic datasets

Inspection of the results lead to the identification of three different scenarios. They are as follows:

1. Inductive rule learning without negation, where the generated classifier is sufficiently accurate;

2. Inductive rule learning where negation is required, if a sufficiently accurate classifier is to be generated;

3. Inductive rule learning where, regardless of whether rules with or without negation are learnt, a suitably accurate classifier cannot be built.

An exemplar dataset for each scenario is presented in Table 3.4. With respect to this table, it should be noted that the features in each record are ordered in a decreasing manner according to their discriminative value, indicating that the first feature is the most significant feature in the record, the second (if present) is the second most significant, and so on. In each case, the class label is the last item in the record. These three scenarios are discussed in more detail in the following three sub-sections.

Exemplar datasets:
Dataset 1
$\{b, c, a, y\}$
$\{a, x\}$
$\{c, z\}$

Dataset 2
$\{a, b, y\}$
$\{a, b, y\}$
$\{c, b, a, x\}$

Dataset 3
$\{a, b, x\}$
$\{a, b, y\}$
$\{a, b, z\}$

Table 3.4: Exemplar datasets used to describe the three identified scenarios

3.2.1 Scenario 1: Inductive rule learning without negation, where the generated classifier is sufficiently accurate

Scenario 1 is concerned with the situation where a given dataset can be classified, with an accuracy of 100%, using IRL without the need for negation. An exemplar dataset where this is the case is Dataset 1 in Table 3.4. Considering this dataset, rule learning using the covering algorithm, will first learn the rule $b \Rightarrow y$ to "cover" the first record. This rule will accurately classify the first record and not apply to the second and third records. The first record is then removed from further consideration. The rule learning

process continues with the second record, where the rule $a \Rightarrow x$ is learnt, which will accurately classify the second record. The second record is thus removed from further consideration and the rule learning moves on to the third record. The rule $c \Rightarrow z$ is then learnt, which correctly classifies the third record. Therefore, on completion, the rule learning process has produced a classifier with three rules as follows:

$$b \Rightarrow y$$
$$a \Rightarrow x$$
$$c \Rightarrow z$$

This classifier gives a classification accuracy of 100% when applied to the input dataset when the rules are fired in the order that they appear in the ruleset. A common rule ordering strategy is rule length ordering, whereby rules with more features in their antecedents are ranked higher than rules with fewer features in their antecedents. This is so that more specific rules are fired first. If there is a tie in rule length, other rule measures, for example, rule coverage or rule accuracy, will be used for ordering the rules. In this case, all three rules have the same rule length, rule coverage and rule accuracy, and therefore are sorted in the order that they are learnt.

This scenario depicts a straightforward and simple rule learning situation where rule learning with negation is not required to obtain an accurate classifier. With reference to Table 3.2, $4,503$ of the $9,261$ synthetic datasets generated fall into this category. Table 3.5 presents a text classification example of the above scenario. As in the case of Table 3.4, the last item in each record is the class label. Three rules are learnt that each correctly classifies the three documents. Learning rules from these three documents are straightforward and simple, without needing any feature to be negated.

Documents
Document 1 = {*bike*, *ride*, *seat*, *motorcycle*}
Document 2 = {*credit*, *bank*, *rates*, *finance*}
Document 3 = {*oil*, *barrel*, *price*, *gallon*, *crudeoil*}

Rules learnt without negation
Rule 1: *bike* $\Rightarrow$ *motorcycles*
Rule 2: *credit* $\Rightarrow$ *finance*
Rule 3: *oil* $\Rightarrow$ *crudeoil*

Table 3.5: Text classification example for Scenario 1

3.2.2 Scenario 2: Inductive rule learning where negation is required, if a sufficiently accurate classifier is to be generated

Scenario 2 describes the situation where the dataset can only be effectively classified if negation is included in the IRL process. A typical exemplar dataset where this is the case is Dataset 2 in Table 3.4. In this case, the rule learning process will start with the first record, learning the rule $a \Rightarrow y$. This rule is not 100% accurate as it correctly classifies the first and second records but incorrectly classifies the third record. Refining the rule with another positive feature b gives the rule $a \wedge b \Rightarrow y$; but again, this rule only correctly classifies the first and second records, and incorrectly classifies the third record. Since there are no more positive features available to refine the rule with, it is added into the classifier and the covered records are removed from further consideration. In this case, only one rule is learnt from this dataset as follows:

$$a \wedge b \Rightarrow y$$

However, when the ruleset is used to classify the dataset, it only gives a 67% accuracy; it only correctly classifies two out of the three records. This scenario represents an example situation where an accurate classifier can only be generated using rules with negation.

Given an IRL system that supports rule learning with negation, and again referring to Dataset 2 in Table 3.4, the rule learning process will be as follows. The rule $a \Rightarrow y$ is first learnt (as before) and it correctly classifies the first and second records but incorrectly classifies the third record. As it is not 100% accurate, it is specialized further to $a \wedge b \Rightarrow y$. This refined rule is also not 100% accurate but since there are no more positive features to specialize the rule with, negated features can be added to improve the rule accuracy. In this case, the rule can be refined by adding the negated feature c to obtain $a \wedge b \wedge \neg c \Rightarrow y$. This rule now becomes 100% accurate and the first and second records, which are covered by the rule, are removed from further consideration. The rule $c \Rightarrow x$ is then learnt from the third record. The resulting classifier then comprises the two following rules:

$$a \wedge b \wedge \neg c \Rightarrow y$$
$$c \Rightarrow x$$

Note that this classifier gives a 100% accuracy when applied to the dataset as opposed to only 67% when rules without negation are used. This scenario clearly demonstrates that rules with negation are necessary in certain situations if accurate classifiers are to be generated. With reference to Table 3.2, (6,825 - 4,503) = 2,322 of the 9,261 synthetic datasets fall into this category.

The example presented in Table 3.6 shows the rule learning process with and without negation when applied to a text classification task. When rules without negation are

learnt, the ruleset can only classify two out of three documents correctly. However, with the use of negation in the rule learning process, the rules learnt can correctly classify all three documents. This scenario thus provides evidence that rule learning with negation is indeed necessary.

Documents
Document 1 = {*bike*, *ride*, *seat*, *bicycle*}
Document 2 = {*seat*, *bike*, *ride*, *bicycle*}
Document 3 = {*clutch*, *seat*, *bike*, *ride*, *motorcycle*}

Rules without negation learnt
Rule 1: $bike \wedge ride \wedge seat \Rightarrow bicycle$

Rules with negation learnt
Rule 1: $bike \wedge ride \wedge seat \wedge \neg clutch \Rightarrow bicycle$
Rule 2: $clutch \Rightarrow motorcycle$

Table 3.6: Text classification example for Scenario 2

It can be argued that in this case, if the order of the documents are interchanged (by placing Document 3 in the first position), then it is possible to learn a ruleset which is 100% accurate without the need for negation. However, with so many possibilities of different scenarios occurring in different datasets, the scenario described in this subsection can still occur and when it does, the advantage of including negation in the IRL process will be beneficial.

3.2.3 Scenario 3: Inductive rule learning where, regardless of whether rules with or without negation are learnt, a suitably accurate classifier cannot be built

Scenario 3 presents the situation where a dataset contains contradictory information and therefore cannot be resolved regardless of whether IRL with negation is used or not. An exemplar dataset where this is the case is Dataset 3 in Table 3.4. Note that the dataset contains contradictions. In other words, a ruleset that accurately classifies the dataset can never be learnt. Rules that can be learnt from this dataset are $a \Rightarrow x$ or $a \wedge b \Rightarrow x$ which will serve to accurately classify only one record. With reference to Table 3.2, (2,316 + 120) = 2,436 of the 9,261 synthetic datasets fall into this category.

Table 3.7 gives an example of the rule learning process when applied to a text classification task and indicates that, regardless of whether rules with or without negation are learnt, IRL can never produce a ruleset that can accurately classify the (contradictory) documents.

Documents
Document 1 = {*bike*, *ride*, *seat*, *motorcycle*}
Document 2 = {*ride*, *bike*, *seat*, *bicycle*}
Document 3 = {*seat*, *ride*, *bike*, *tricycle*}

Rules learnt
Rule 1: $bike \Rightarrow motorcycle$
Alternative possible rules : $bike \wedge ride \Rightarrow motorcycle$, $bike \wedge ride \wedge seat \Rightarrow motorcycle$

Table 3.7: Text classification example for Scenario 3

3.3 Summary

This chapter has sought to establish the requirement for IRL with negation. This has been established using all possible combinations that may exist in a three record synthetic dataset. As a result, three separate scenarios were identified. Scenario 1 described a situation where 100% accuracy in classification could be obtained using a rule learning process that did not include negation. Scenario 2 showed a situation where 100% accuracy in classification could only be attained when using a rule learning process that included negation. Scenario 3 described a situation that included contradictions, which could not be satisfactorily resolved regardless of the nature of the rule learning process adopted. For the sake of simplicity, the number of features used to generate the synthetic datasets was deliberately kept small. Obviously, in real life, the datasets to be considered may contain thousands of records and many features, especially in the case of text mining applications. However, the initial experiment conducted clearly indicates that there is a real need for IRL with negation if the aim is to generate significantly accurate classifiers. The experiment results show a 25.1% increase in the number of datasets accurately classified when negation is included in the IRL process. With this point as motivation, the next chapter details the proposed framework for IRL with negation.

Chapter 4

Framework for an Inductive Rule Learning Based Text Classification

The proposed framework for inductive rule learning (IRL)-based text classification and its components are described in this chapter. This framework is designed with the aim of providing a platform to support the evaluation of ideas concerning the use of negation and phrases in IRL. The chapter is organized as follows. An overview of the proposed text classification framework is given in Section 4.1. This is followed by detailed discussion of the key processes incorporated into the framework. Section 4.2 describes the document preprocessing, which includes a discussion of data cleaning and the adopted techniques for keyword and phrase extraction. In addition, the feature selection techniques, which are used in the keyword and phrase extraction processes, and the text representation formats used, are also discussed. Section 4.3 then describes the proposed IRL mechanism, which incorporate the use of negation. This section includes descriptions on the feature space division process and the rule refinement strategies. Finally, Section 4.4 describes two approaches to classification to which the proposed framework can be applied, namely binary classification and multi-class classification. Section 4.5 concludes the chapter with a summary.

4.1 Text Classification Framework

The proposed framework for the IRL-based text classification consists of a number of processes. The three main processes are: (i) document preprocessing, (ii) IRL and (iii) classification. Each of these processes is discussed in more detail in the following three sections. Figure 4.1 presents an overview of the proposed IRL-based text classification framework.

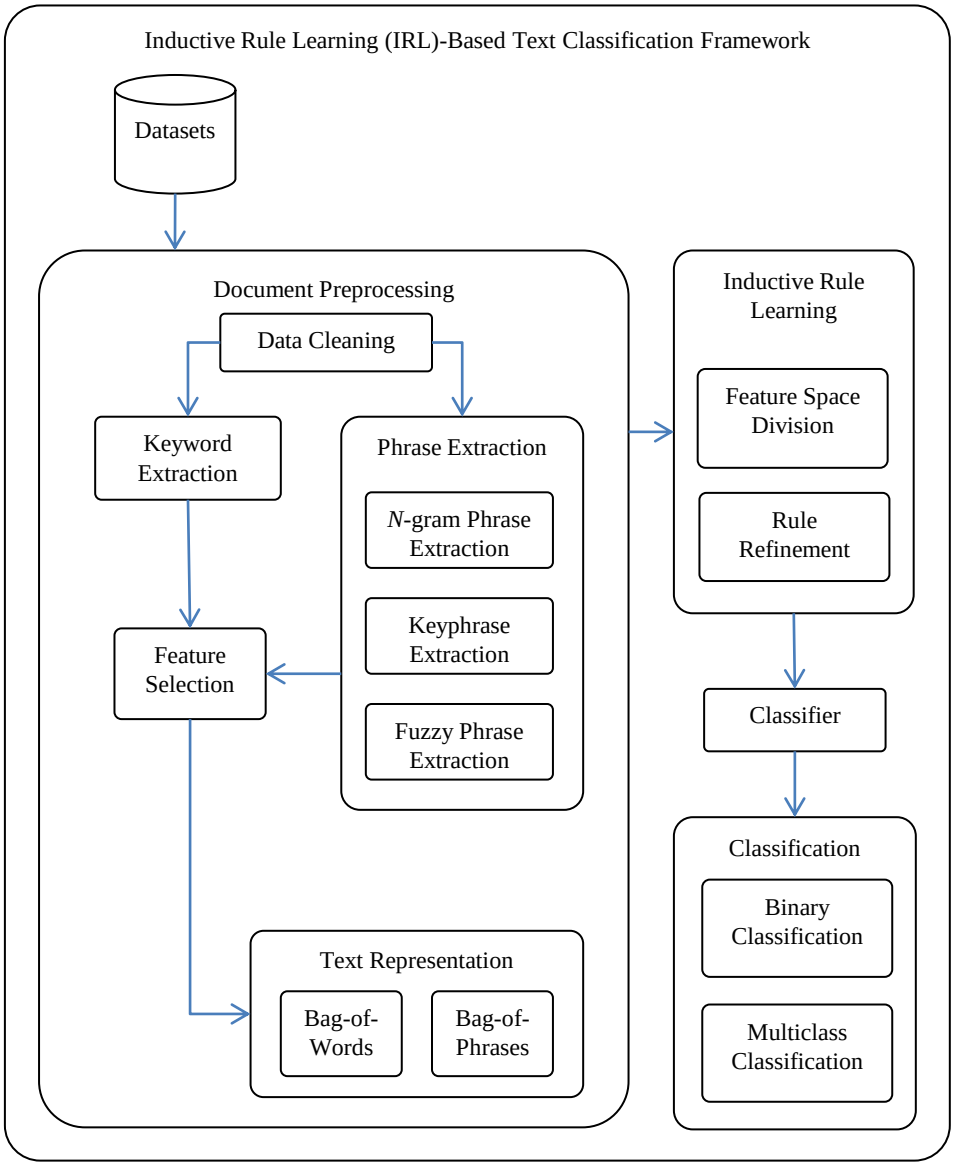

Figure 4.1: Proposed IRL-based text classification framework

4.2 Document Preprocessing

In document preprocessing, the dataset used is first prepared to conform to the desired input format. With respect to the framework implemented to test ideas expressed in this thesis, each document in the input dataset comprised a text file (suffix *.txt*). Both a naming and a format convention was used. For the naming convention, each text file was named with the dataset name followed by a sequential number. The format convention used was that each text file was formatted so that the first line in each file included a class tag, indicated by the string @*Class*, followed by the class label. Subsequent lines then gave the text file contents. Figure 4.2 shows a sample of the format of a typical text file (taken from the 20 Newsgroups dataset) used for evaluation purposes in this thesis. This example will be used throughout this chapter to illustrate various aspects of the operation of the proposed framework.

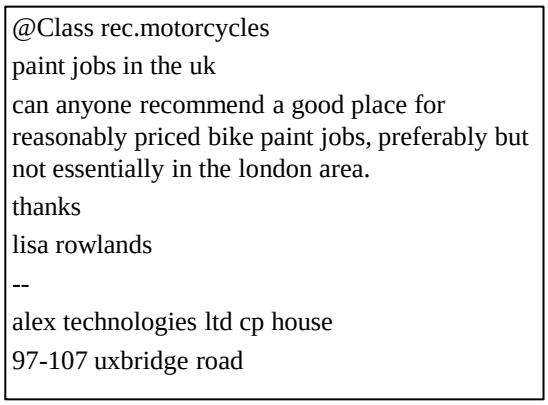

Figure 4.2: Sample text file

There are five sub-processes involved in the IRL-based text classification document preprocessing stage. These are: (i) data cleaning, (ii) keyword extraction (iii) phrase extraction (iv) feature selection, and finally (v) text representation. In data cleaning, a number of operations can be performed. They include: stop word, symbol, number and email address removal. The data cleaning sub-process is described in more detail in Sub-section 4.2.1. The extraction of keywords and phrases are described in Sub-sections 4.2.2 and 4.2.3 respectively. The aim is to identify suitable keywords and phrases that can be used to represent a document collection. Feature selection, described in Sub-section 4.2.4, is the process of selecting a subset of the identified keywords and/or phrases from the full feature space; note that the number of selected keywords and/or phrases can be significant depending on the nature of the dataset under consideration. Two feature selection techniques are used; Chi Square (χ^2) and Information Gain (IG). The final sub-process is text representation, described in Sub-section 4.2.5, where the document collection is "recast", according to the identified keywords and/or phrases,

into a representation suited to the application of text classification algorithms as required by the framework.

4.2.1 Data Cleaning

Data cleaning, as already noted, may include stop word, symbol, number and email address removal. Stop word removal is the process of removing non-informative words, such as articles, prepositions and conjunctions. The stop word list used in this thesis was that provided by Webconfs[1] augmented by an additional 24 "words". The Webconfs list was designed to be used in conjunction with search engine optimization tools, but is well suited to text classification. The additional 24 "words" are the alphabetic letters from 'a' to 'z' which were added to the list, with the exception of 'k' and 'v' which were already in the original list. The complete list, containing 659 stop words, is included in Appendix A. This stop word list was found to work well with respect to the text classification scenarios considered in this thesis. However, the appropriateness of stop word lists depends on the nature of the text classification task that a user intends to carry out. While general stop word lists, such as that used in this thesis, are suitable for many classification tasks, some domain specific applications, such as medical document classification, may require specialized stop word lists. The removal of stop words is optional with respect to the proposed framework, as stop words are required with respect to the extraction of some of the phrases (as discussed in Sub-section 4.2.3).

In common with the work conducted by many other practitioners, symbols and numbers are also removed from the text content in this thesis. Symbols, such as punctuations, are usually deemed to be uninformative with respect to discriminating between documents. Numbers are removed because, generally speaking, numbers (such as telephone numbers, dates and quantities) also tend to be poor class discriminators, although it is possible to envisage applications where this is not the case. In addition, where appropriate, email addresses are also removed, on the grounds that they are uninformative in the context of general text classification. Again however, some applications, such as email classification systems, may require that email addresses be retained as features. An advantage of removing stop word, symbol, numbers and email addresses, in addition to the desire to obtain a set of features that are good class discriminators, is that it reduces the overall size of the input dataset to be considered, thus making it more manageable.

4.2.2 Keyword Extraction

After data cleaning is done, keyword extraction can be performed. A keyword is essentially a single word selected from the feature space because it is considered to be a good class discriminator. In the case of the proposed framework, so as to facilitate keyword

[1] http://www.webconfs.com/stop-words.php

extraction, all stop words, symbols, numbers and email addresses are removed from the input document collection during the data cleaning process. The discriminative value for each word left in the documents for each class is then calculated using the χ^2 and IG formulas. The words will then be sorted in descending order according to their discriminative values. For each class in the dataset, a subset of words is selected to form a set of keywords based on a pre-determined percentage for each class.

The keyword extraction process is summarized as follows:

1. Calculate the discriminative value (using feature selection techniques) of the words left in the documents for each class after data cleaning.
2. Sort the words from each class in descending order according to their discriminative values.
3. Select a subset of words from each class (using a pre-determined percentage) to be used as keyword features for the text representation.

4.2.3 Phrase Extraction

An alternative to using keywords for text representation is using phrases. In this framework, three different kinds of phrase are extracted:

1. n-gram phrase
2. keyphrase
3. fuzzy phrase

The technique adopted for phrase extraction, with respect to the proposed framework, is the Shingling algorithm of Broder [8], which is specifically directed at the extraction of phrases to be used for detecting document similarity. The Shingling algorithm has also been used to extract phrases for email classification [14]. In the Shingling algorithm, contiguous words are extracted in an "overlap" manner. This is demonstrated in the following example. Given the sentence:

The house by the river is on fire.

According to the Shingling algorithm, this sentence contains the following phrases of length two: *The house*, *house by*, *by the*, *the river*, *river is*, *is on* and *on fire*. Phrases of size three and so on can be extracted in the same manner. In this thesis, phrases of various lengths were considered. Phrases of length two and three were considered with respect to n-gram phrase (2-grams and 3-grams) and keyphrase (KP-2 and KP-3) extraction, while phrases of length three and four were considered for fuzzy phrase (FP-1 and FP-2) extraction. Phrases of length one are essentially single words and

not phrases. Phrases longer than four were not experimented with in this thesis as previous works [34, 67] had shown them to be less effective for text representation. The extraction of n-gram phrases, keyphrases and fuzzy phrases is described below.

N-gram Phrase Extraction

With respect to the proposed framework, an n-gram is a sequence of n words ignoring stop word, symbol, number and email address. For the experiments conducted, n-gram phrases of length two (2-grams) and three (3-grams) ($n = 2, 3$) are extracted. All stop words, symbols, numbers and email addresses are first removed from the documents in the data cleaning process. This means that the words left in the documents are those which can potentially be selected as keywords. Using the Shingling algorithm, n-gram phrases of length two and three are extracted. The discriminative value for each n-gram phrase is calculated using the χ^2 and IG formulas. These extracted n-gram phrases are then sorted in descending order according to their discriminative values. A pre-determined percentage is used to select a subset of the extracted n-gram phrases from each class to be represented as features.

The rationale of extracting n-gram phrases in this manner is that words that are potential keywords and appear closely to each other in a particular order in a document, may serve as a more promising feature than a single keyword if they are bound together as a phrase. For example, if the following sentence is found in a document for the class "bicycle": "There is a bike sale tomorrow.", a possible 2-gram phrase that can be extracted is "bike sale". In another document from the same class, the following sentence may be found: "Anyone has a bike for sale?", whereby if stop words are removed, the 2-gram phrase "bike sale" can also be extracted. While the keyword "bike" alone may be a good feature for the class "bicycle", the word "sale" alone is more general and could likely occur in multiple classes. In this case, "bike sale" would appear as a more promising feature than "sale" alone for text classification.

The n-gram phrase extraction process is summarized as follows:

1. Extract n-gram phrases from the words left in the documents for each class after data cleaning.

2. Calculate the discriminative value (using feature selection techniques) of the n-gram phrases for each class.

3. Sort the n-gram phrases from each class in descending order according to their discriminative values.

4. Select a subset of n-gram phrases from each class (using a pre-determined percentage) to be used as phrase features for the text representation.

Table 4.1 gives an example of the n-gram phrase extraction process using the example text presented in Figure 4.2.

Sample text content with stop words, symbols, numbers and emails removed: {paint jobs uk recommend good place priced bike ...}
2-grams extracted: {paint jobs, jobs uk, uk recommend, recommend good, good place, place priced, priced bike}
3-grams extracted: {paint jobs uk, jobs uk recommend, uk recommend good, recommend good place, good place priced, place priced bike}

Table 4.1: N-gram phrase extraction

Keyphrase Extraction

A keyphrase in the context of this thesis is a sequence of two or more words that includes at least one keyword. A keyword here refers to a single word that is not a stop word, extracted using the keyword extraction process and selected as a keyword according to its discriminative value as described in Sub-section 4.2.2. In the extraction of keyphrases, stop words are not removed but symbols, numbers and email addresses are excluded. Keyphrase extraction is therefore aimed at extracting contiguous sequences of words from documents in the dataset, including stop words, using the Shingling algorithm.

The main difference between an n-gram phrase and a keyphrase is the presence of stop words in the documents during the extraction process. The idea here is that by retaining stop words, phrases may better maintain their underlying meaning (in comparison with the use of n-gram phrases). For example, suppose "fire" is a keyword and the two keyphrases "on fire" and "fire at" are extracted. These two phrases carry different underlying meaning because of the use of the prepositions "on" and "at". These two phrases could very well occur in two different classes and could serve to differentiate those classes.

The proposed framework supports the extraction of keyphrases of length two and three. As before, all symbols, numbers and email addresses are first removed in the data cleaning process. Stop words are left in the documents and not removed for the reason given above. Keywords are first identified using the keyword extraction process described in Sub-section 4.2.2. A pre-determined percentage of the keywords is selected and using these selected keywords, keyphrases of the length two (KP-2) and three (KP-3) are extracted. The extracted keyphrases are then sorted according to

their discriminative values calculated using the χ^2 and IG formulas. Lastly, the same pre-determined percentage as before is again used to select a subset of the extracted keyphrases from each class to be used as phrase features.

The keyphrase extraction process can be summarized in terms of the following steps:

1. Perform keyword extraction to obtain a list of selected keywords (using a pre-determined percentage).
2. Based on the selected keywords, extract keyphrases from the dataset that contain at least one keyword.
3. Calculate the discriminative value (using feature selection techniques) of the keyphrases for each class.
4. Sort the keyphrases from each class in descending order according to their discriminative values.
5. Select a subset of keyphrases from each class (using the same pre-determined percentage) to be used as phrase features for the text representation.

For keyphrases of length two, there are five different word configurations that can be identified. They are shown in Table 4.2. In the table, a KeyWord is a single keyword associated with a particular class, extracted in the manner described in Sub-section 4.2.2. An OtherWord is a non-keyword associated with this particular class while a StopWord is a stop word from the list shown in Appendix A. Similarly, in the case of keyphrases of length three, 19 different word configurations can be identified as shown in Table 4.3. Table 4.4 shows an example of keyphrase extraction, again in the context of the example document given in Figure 4.2.

(KeyWord_KeyWord)
(KeyWord_OtherWord)
(KeyWord_StopWord)
(OtherWord_KeyWord)
(StopWord_KeyWord)

Table 4.2: Possible keyphrase configurations of length two

(KeyWord_KeyWord_KeyWord)
(KeyWord_KeyWord_OtherWord)
(KeyWord_KeyWord_StopWord)
(KeyWord_OtherWord_Keyword)
(KeyWord_StopWord_Keyword)
(OtherWord_KeyWord_KeyWord)
(StopWord_KeyWord_KeyWord)
(KeyWord_OtherWord_OtherWord)
(KeyWord_OtherWord_StopWord)
(KeyWord_StopWord_StopWord)
(KeyWord_StopWord_OtherWord)
(OtherWord_KeyWord_OtherWord)
(OtherWord_KeyWord_StopWord)
(StopWord_KeyWord_StopWord)
(StopWord_KeyWord_OtherWord)
(OtherWord_OtherWord_KeyWord)
(OtherWord_StopWord_KeyWord)
(StopWord_StopWord_KeyWord)
(StopWord_OtherWord_KeyWord)

Table 4.3: Possible keyphrase configurations of length three

Sample text content with symbols and numbers removed: {paint jobs in the uk can anyone recommend a good place for reasonably priced bike ...}

Keywords: {paint, jobs, uk, recommend, good, place, priced, bike ...}

KP-2 extracted: {paint jobs, jobs in, the uk, uk can, anyone recommend, recommend a, a good, good place, place for, reasonably priced, priced bike}
(Note that the phrases {in the, can anyone, for reasonably} are not extracted as keyphrases because they do not contain at least one keyword.)

KP-3 extracted: {paint jobs in, jobs in the, in the uk, the uk can, uk can anyone, can anyone recommend, anyone recommend a, recommend a good, a good place, good place for, place for reasonably, reasonably priced bike}

Table 4.4: Keyphrase extraction

Fuzzy Phrase Extraction

In the proposed framework, a fuzzy phrase is defined as a phrase that starts and ends with a keyword, but is separated by n other words, which may be KeyWords, OtherWords or StopWords. Similar to keyphrase extraction, a KeyWord in this context is a single word extracted using the keyword extraction process as described in Subsection 4.2.2. These keywords are then used for the identification of fuzzy phrases. A fuzzy phrase that is extracted contains at least two keywords: one at the start of the phrase and the other at the end of the phrase. Two potential values for the parameter n are used with respect to the experiments conducted for this thesis, which signifies the number of words separating the two keywords, i.e. $n = 1$ and 2. When $n = 1$ (FP-1), the two keywords are separated by one word; thus, the fuzzy phrase is of length three. Similarly, when $n = 2$ (FP-2), the two keywords are separated by two words, resulting in the extraction of fuzzy phrases of length four. When using $n = 1$, the fuzzy phrases extracted subscribed to the pattern "KeyWord _ KeyWord", where "_" can be another KeyWord, an OtherWord or a StopWord. As before, an OtherWord is a non-keyword associated with this particular class while a StopWord is a stop word from the list shown in Appendix A. Similarly, when using $n = 2$, the fuzzy phrases subscribe to the pattern of "KeyWord _ _ KeyWord".

What makes a fuzzy phrase "fuzzy" is that the words in between the two keywords are "wild card" words. "Wild card" words can match to any words in a document. Therefore, two fuzzy phrases are considered a match if they have the same starting and ending keywords, regardless of what other words appear in between the two keywords. The main difference between a fuzzy phrase, an n-gram phrase and a keyphrase is that a fuzzy phrase includes the "wild card" notion to make the phrases fuzzy. The rationale for incorporating fuzziness is because finding a pattern where two keywords are separated by n words is expected to be comparatively rarer. However, although rare, a pattern of such form can still be useful in differentiating between classes.

For fuzzy phrase extraction, symbols, numbers and email addresses are removed but stop words are retained. Again, keywords are identified in the manner described in Sub-section 4.2.2. A pre-determined percentage is used to select a subset of keywords and based on the selected keywords, fuzzy phrases are extracted according to the value for the parameter n. As before, the extracted fuzzy phrases are ordered in descending order according to their discriminative values calculated using χ^2 or IG. Again, the same pre-determined percentage is used to select a subset of the extracted fuzzy phrases to be used as phrase features for text representation.

The following steps show how the fuzzy phrase extraction process operates:

1. Perform keyword extraction to obtain a list of selected keywords (using a pre-determined percentage).

2. Based on the selected keywords, extract fuzzy phrases from the dataset.

3. Calculate the discriminative value (using feature selection techniques) of the fuzzy phrases for each class.

4. Sort the fuzzy phrases from each class in descending order according to their discriminative values.

5. Select a percentage of the fuzzy phrases from each class (using the same pre-determined percentage) to be used as phrase features for the text representation.

Table 4.5 shows an example of how the fuzzy phrase extraction is done using the example document presented in Figure 4.2.

Sample text content with symbols and numbers removed: {paint jobs in the uk can anyone recommend a good place for reasonably priced bike ...}
Keywords: {paint, jobs, uk, recommend, good, place, priced, bike ...}
FP-1 extracted: {recommend _ good}
FP-2 extracted: {jobs _ _ uk, uk _ _ recommend, recommend _ _ place, place _ _ priced}

Table 4.5: Fuzzy phrase extraction

4.2.4 Feature Selection

In the proposed framework, feature selection is used in the keyword and phrase extraction process to select a percentage of keywords and phrases to be used for the text representation. The objective of feature selection is to select a subset of features from the full feature space. Two well-known feature selection techniques, namely χ^2 and IG were incorporated into the framework. These two techniques were chosen for their effectiveness and extensive usage in the text classification literature.

The formulas for χ^2 and IG were given in Sub-sections 2.5.1 and 2.5.2 respectively in Chapter 2. Based on these formulas, χ^2 and IG values were calculated for each of the extracted keywords and/or phrases. These keywords and/or phrases were then ranked in descending order according to the χ^2 and IG values and a pre-determined percentage of the keywords and/or phrases were selected as features for the text representation. A percentage was chosen instead of setting a fixed threshold because of the

expected uneven size of the number of features for each class. Setting a fixed threshold could therefore under-represent classes with large number of features and over-represent classes with small number of features. Feature selection is performed in the local context, where χ^2 and IG values are calculated for keywords and/or phrases local to a particular class. This means that a different subset of keywords and/or phrases will be selected as features for each class. In the experiments reported later in this thesis, due to the substantial number of features extracted associated with each class in the datasets used, two pre-determined percentages were used for feature selection; 10% (90% reduction) and 1% (99% reduction). Either one of these pre-determined percentage is used depending on computational limitations, especially in the larger datasets. In this context, it should also be noted that in [95], favourable results were reported when a rigorous reduction of more than 90% was applied when using χ^2 and IG.

Table 4.6 shows an example of how the feature selection is done with respect to the example document presented in Figure 4.2.

Keywords: {paint, jobs, uk, recommend, good, place, priced, bike ...}
Calculate χ^2: paint(13.74), jobs(81.54), uk(6.97), recommend(105.65), good(27.55), place(50.23), priced(99.25), bike(72.88), ...
Sort in descending order: recommend(105.65), priced(99.25), jobs(81.54), bike(72.88), place(50.23), good(27.55), paint(13.74), uk(6.97), ...
Select a subset based on a pre-determined percentage of 10% (assumming there are 20 keywords in this class: recommend(105.65), priced(99.25)

Table 4.6: Example of feature selection

4.2.5 Text Representation

The output of the document preprocessing process is a set of features that have to be represented in a manner such that they can be mapped to each of the documents in the dataset. The most popular methods for text representation are the bag-of-words and bag-of-phrases representations. The experiments conducted with respect to this thesis were directed at both. The framework therefore provides for both representations. The framework also supports the Attribute Relation File Format (ARFF) used with WEKA, which is therefore also discussed in this sub-section.

In the case of the bag-of-phrases representation, the representation may be either of fixed or mixed word length. In the fixed word length representation, only phrases of

a specific length are represented at a time. The mixed word length representation on the other hand, may comprise keywords and phrases of different lengths. This means that the mixed word length representation tends to consist of a lot more features than the fixed word length representation.

Bag-of-Words and Bag-of-Phrases

In the bag-of-words representation, selected keywords associated with each class are used to represent each document in the dataset. As the name implies, the bag-of-words representation is only used for representing documents using keywords as features. The bag-of-phrases representation is similar to the bag-of-words representation, differing only in the use of phrases as features. In the context of the framework, the bag-of-phrases representation is used when any of the following are used as features: 2-gram, 3-gram, KP-2, KP-3, FP-1 and FP-2. The recognized disadvantage of the "bag" representation is that the ordering of words and phrases with respect to other words and phrases in the dataset is lost. However, the text classification literature has mostly favoured the "bag" representation due to it simplicity and acceptable effectiveness.

Using the "bag" representation, the keywords and phrases are interpreted as binary valued features, i.e. they are represented in terms of their presence or absence in a document. Each document is represented in terms of a document vector whose elements represent the keywords and/or phrases which are present in the document. If a feature of a class is present in the document, the document vector will include the feature. However, if the feature is absent from the document, the document vector will exclude the feature. Table 4.7 shows an example of the bag-of-words representation, while Table 4.8 shows an example of the bag-of-phrases representation, with respect to example documents taken from the 20 Newsgroups dataset.

{Doc 1: graphics, polygon, polygons, vga, program, writes, texture, ... , class:comp.graphics}
{Doc 2: bit, information, chip, class:comp.sys.mac.hardware}
{Doc 3: power, class:rec.sport.hockey}
{Doc 4: library, program, writes, precision, write, fact, supports, ... , class:sci.crypt}
{Doc 5: writes, people, article, ac, power, fire, children, faith, ... , class:talk.politics.guns}
{Doc 6: don, dos, pl, class:comp.os.ms-windows.misc}
.
.
.

Table 4.7: Sample bag-of-words representation

{Doc 1: fast_polygon, routine_needed, graphics_program, polygon_routine, ian_romanick, simple_fast, ... , class:comp.graphics}
{Doc 2: class:comp.sys.mac.hardware}
{Doc 3: class:rec.sport.hockey}
{Doc 4: tel_fax, class:sci.crypt}
{Doc 5: class:sci.space}
{Doc 6: ac_uk, class:talk.politics.guns}
.
.
.

Table 4.8: Sample bag-of-phrases representation

Attribute Relation File Format (ARFF)

In addition to the bag-of-words and bag-of-phrases text representation, the framework also supports the creation of Attribute Relation File Format (ARFF) files, the input file format for the Waikato Environment for Knowledge Analysis (WEKA) machine learning workbench [92]. The framework allows ARFF files to be created from either the bag-of-words or the bag-of-phrases representation. One ARFF file is created for each class in the dataset. These ARFF files are then used as input for binary classification using the WEKA machine learning workbench.

The ARFF format includes: (i) the name of the relation (class); (ii) the attributes (features) representing the class and (iii) the instances (documents) in the dataset. Lines that begin with "%" are comments. The name of the class must be placed in the first line of the file, preceded by the "@relation" tag. Following this is the list of features representing the class. Each feature is preceded by an"@attribute" tag. The feature values can be either nominal or numeric. Nominal values are specified in curly braces while numeric ones are followed by the keyword "numeric" The last item in the feature list is the class attribute, where each document is classified as either belonging to this class or not. Following the class attribute, is the "@data" tag, which signifies the start of the document representation. This tag separates the list of features from the document representation. Each document in the dataset is represented as a single line between curly braces as follows:

{(feature_index *value*), (feature_index *value*) ... ... (class_attribute_index *value*)}

where the numbers; feature_index and *value*, are separated by a white space. The feature_index indicates the index of a feature in the feature list while the *value* represents

the presence or absence of the feature (1 for presence and 0 for absence, in the case of the binary representation adopted for the framework). The normal representation is to list the index for all the features in the feature list and to assign the values 1 or 0 to indicate the presence or absence of the term. However, in the case of binary text classification, the number of features is usually large and not many features will occur in all the documents, especially documents from classes other than the one being considered. Therefore, to prevent listing the indexes of all the features whereby most will be assigned the value 0, an alternative way to represent a document is to only list the indexes of features which are present in a document and to assign the value 1 to it. The following is a sample of a single line representing a single document in a dataset.

{0 1, 8 1, 14 1, 15 1, 26 1, 32 1, 39 1, 61 1, 71 1, 118 1, 146 1, 333 1, 337 1, 351 1, 406 1, 435 1, 439 1, 440 1, 447 1, 591 1, 604 1, 617 1, 638 1, 644 1, 802 1, 864 1, 916 1, 917 1, 978 1, 1284 1, 1352 1, 1374 1, 1395 1}

The first pair of numbers (0 1) indicates that the first feature at index 0 is present in the document. Similarly, the remaining feature_index indicate that the particular feature is present in the document. The last number pair (1395 1) represents the class_attribute_index and value pair, where the value "1" indicates that this document belongs to the class stated after the "@relation" tag. A value "0" for the class_attribute_index will signify that the document does not belong to the class stated after the "@relation" tag. Figure 4.3 shows an extract from an ARFF file.

4.3 Inductive Rule Learning with Negation

The proposed IRL mechanism with negation aims to improve the effectiveness of classifiers by using both positive and negated features, while maintaining the simplicity and effectiveness of the covering algorithm. In the covering algorithm, rules are learned sequentially according to the nature of the training set. The documents "covered" by a rule learnt are then removed and the process is repeated until a stopping condition is achieved (typically when all documents in the training set are "covered"). The algorithm for the proposed IRL mechanism with negation is presented in Algorithm 3. The input consists of a dataset of labelled documents, the set of features for class c, the strategy used for rule refinement and the class label. For a given class c, rule learning will start with the first feature in the $Feature\_set_c$ of class c. Rules are learned sequentially one at a time based on a training set (D, in the case of the example shown in Algorithm 3) using the **LearnOneRule** method. The examples "covered" by a rule learnt are then removed and the rule is added to the ruleset for class c, Ruleset_c. The index for the $Feature\_set_c$ is then increased by one and the process is repeated until a stopping condition is met. In this mechanism, the stopping conditions are as follows: (i) when there are no more uncovered documents or (ii) when there are no more unused

```
@relation comp.graphics
@attribute graphics numeric
@attribute image numeric
@attribute gif numeric
@attribute animation numeric
@attribute images numeric
.
.
.
@attribute federal numeric
@attribute attendees numeric
@attribute msstate numeric
@attribute multiply numeric
@attribute nominal numeric
@attribute class {1,0}
@data
{0 1, 8 1, 14 1, 15 1, 26 1, 32 1, 39 1, 61 1, 71 1, 118 1, 146 1, 333 1, 337 1, 351
1, 406 1, 435 1, 439 1, 440 1, 447 1, 591 1, 604 1, 617 1, 638 1, 644 1, 802 1, 864
1, 916 1, 917 1, 978 1, 1284 1, 1352 1, 1374 1, 1395 1}
{220 1, 622 1, 809 1, 1395 0}
{446 1, 1395 0}
{24 1, 26 1, 32 1, 335 1, 439 1, 450 1, 534 1, 583 1, 604 1, 638 1, 643 1, 791 1,
815 1, 823 1, 873 1, 925 1, 1359 1, 1395 0}
.
.
.
```

Figure 4.3: An extract from an ARFF file

features in the *Feature_set*$_c$. The rules in Ruleset$_c$ are then ordered using the **Order** method and are added to the final *Ruleset*.

Many IRL systems that use the covering algorithm to learn rules differ in the way they learn the individual rules, i.e. they have different implementations of the **LearnOneRule** method. Algorithm 4 shows the algorithm for the **LearnOneRule** method proposed with respect to the work described in this thesis. The input consists of: the dataset of labelled documents; the set of features for class c; the strategy used for rule refinement and the class label. The rule learning process starts by creating an empty *Rule* and assigning the class label c as the conclusion of the *Rule*. A feature from the *Feature_set*$_c$ is added to the *Rule* based on the index of the *Feature_set*$_c$. The generated *Rule* so far will then be checked so as to determine whether further refinement is needed. Further refinement is not necessary and the *Rule* will be discarded if:

1. The *Rule* does not cover any documents;
2. The *Rule* does not cover any positive documents (all documents covered are negative documents);
3. The *Rule* covers more negative documents than positive documents.

Where further refinement is needed, the *Rule* will iteratively be refined until one of the following stopping conditions is met:

1. The *Rule* does not cover any negative documents (all documents covered are positive documents);
2. There are no more features to refine the *Rule* with;
3. The previous *Rule* accuracy is greater than or equal to the current *Rule* accuracy.

In the case of condition (1) or (2), the *Rule* will be returned. If condition (3) is met, then the previous *Rule* will be returned.

The method **Order** is used to order the rules in *Ruleset*$_c$. The algorithm for this is shown in Algorithm 5. If there is more than one rule in *Ruleset*$_c$, the rules in *Ruleset*$_c$ will be sorted in descending order according to each rule's length. The length of a rule is defined as the number of features in the *antecedent* of a rule. For rules of the same length, they will be sorted according to each rule's weight; the weight of a rule is defined as the number of documents it covers.

Rule refinement is a significant element of the **LearnOneRule** method. Eight strategies for rule refinement were devised for the IRL mechanism with negation. A number of these strategies generate rules with negation, while the others do not. The latter were devised so that comparisons could be made so as to determine the effectiveness of the proposed strategies that generate rules with negation. The eight strategies

are described in Sub-section 4.3.2. These strategies are based on three different types of features. These will be discussed first in Sub-section 4.3.1.

Algorithm 3: Algorithm for the proposed IRL mechanism with negation based on the covering algorithm

```
   input  : D, a dataset of class-labelled documents
            Feature_set_c, the set of features for class c
            R, the strategy used for rule refinement
            c, the class label
   output: A classifier consisting of a set of IF-THEN rules
 1 Ruleset = { }; //initial set of rules learned is empty;
 2 for each class c do
 3 |   Set Feature_set_c index to 0;
 4 |   while stopping condition is not met do
 5 |   |   Rule = LearnOneRule(Feature_set_c, D, c, R);
 6 |   |   if Rule is not marked to be discarded then
 7 |   |   |   Remove documents covered by Rule from D;
 8 |   |   |   Ruleset_c = Ruleset_c + Rule;
 9 |   |   end
10 |   |   else
11 |   |   |   Discard Rule;
12 |   |   end
13 |   |   Increase Feature_set_c index by 1;
14 |   end
15 |   Ruleset_c = Order(Ruleset_c);
16 |   Ruleset = Ruleset + Ruleset_c;
17 end
18 return Ruleset;
```

4.3.1 Feature Space Division

When a rule is learnt and it covers both *positive* and *negative* documents, then the rule has to be further refined in order to learn a rule that can separate these positive and negative documents. In this context, positive documents are considered to be documents that belong to the class to which the current rule under consideration is directed at, while negative documents are documents that do not belong to the class that the current rule is directed at. These documents consist of a collection of features with different characteristics. This collection of features is referred to as the feature search space Sp, from which features are selected to be used in the rule refinement process. Feature space division is the process of identifying and differentiating the characteristics of the features in Sp. Three types of features are identified in Sp:

Unique Positive (UP) features: The set of features that are found only in the positive documents covered and not in any negative documents covered.

Algorithm 4: Algorithm for LearnOneRule method

input : D, a dataset of class-labelled documents
$Feature\_set_c$, the set of features for class c
R, the strategy used for rule refinement
c, the class label
output: *Rule*

```
Create a new empty rule, Rule;
Set Rule conclusion to c;
Add feature from Feature_set_c to Rule based on index;
if Rule needs refinement then
    while stopping condition is not met do
        Rule = RefineRule(Rule, R, D);
    end
end
else
    Mark Rule to be discarded;
end
return Rule;
```

Algorithm 5: Algorithm for Order method

input : $Ruleset_c$, the ruleset for class c
output: Ordered $Ruleset_c$

```
if Ruleset_c has more than one rule then
    Order rules by their lengths;
    if rules have the same length then
        Order rules by their weight;
    end
end
return Ruleset_c;
```

Unique Negative (UN) features: The set of features that are found only in the negative documents covered and not in any positive documents covered.

Overlapping (Ov) features: The set of features which are found in at least one positive and one negative document covered.

These three types of feature divide Sp into three sub-spaces, namely UP, UN and Ov, as shown in Figure 4.4. Thus Sp = {UP, UN, Ov}. Such a division will allow for effective and efficient identification of features to be negated when generating rules with negation. This is a more desirable option, as opposed to the notion of negating every single feature in the feature space, as in the case of RIPPER.

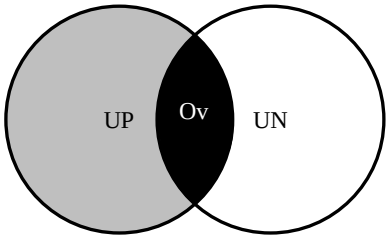

Figure 4.4: Feature space division

During rule refinement, an appropriate feature is selected from Sp to be added to the rule. As a rule is being refined, the documents it covers also change. Thus, the features in Sp change accordingly. It should be noted that any one of the UP, UN and Ov sub-spaces may be empty, as the existence of these features is dependent upon the content of the documents covered by the current rule under consideration.

When refining a rule, a feature from either the UP, UN and Ov feature sub-spaces is selected to be added to the rule. If a rule is refined with a UP or Ov feature, then a rule with no negation is generated. If a rule is refined with a UN feature, then a rule with negation is generated. When refining a rule with a UP or UN feature, the feature with the highest document frequency, i.e. the feature that appears in the most covered documents, is selected. This ensures that the refined rule will cover the maximum possible number of positive documents at every round of refinement. When refining a rule with an Ov feature, the feature with the highest document frequency difference (i.e. positive document frequency minus negative document frequency) is selected to be added to the rule. This is because an Ov feature occurs in both positive and negative documents and the feature that appears in the highest number of positive documents and the least number of negative documents is considered to be the most desirable as its selection will result in a refined rule covering a maximum possible number of positive documents.

4.3.2 Rule Refinement Strategies

Based on the division of Sp into UP, UN and Ov, eight strategies for rule refinement may be devised as follows:

1. UP
2. UN
3. Ov
4. UP-UN-Ov
5. UN-UP-Ov
6. BestStrategy
7. BestPosRule
8. BestRule

The first five strategies are named after the sub-spaces used. Strategies (1), (2) and (3) are directed at utilizing only a single sub-space. Given that a sub-space may be empty, having an empty set using the UP, UN and Ov strategies would mean that refinement may be prematurely halted in the absence of any features to be added. Strategies (4) and (5), which are the UP-UN-Ov and UN-UP-Ov strategies, are designed to address this issue in that they use a sequence of sub-space combinations. The two strategies are labelled in the order that the sub-spaces are considered. Thus, the sequence combination of UP-UN-Ov entails the use of UP features first. If the UP sub-space is empty, the UN features will be considered instead, and then the Ov features if the UN sub-space is also empty. The UN-UP-Ov sequence combination strategy works in a similar manner, only inter-changing the order of UP and UN. In both cases, Ov is used last because using Ov features will always result in the coverage of at least one negative document. In both cases, if the first sub-space is not empty, then only the first sub-space will be used for every round of rule refinement. This would mean that the UP-UN-Ov sequence combination strategy may produce the same results as the UP strategy, and similarly the UN-UP-Ov sequence combination strategy may produce the same results as the UN strategy. Other sequence combinations like UP-UN, UN-UP, UP-Ov, UN-Ov and so on, are not considered as these are subsets of both UP-UN-Ov and UN-UP-Ov and would have been covered by these two.

A common drawback associated with the first five strategies is that in each round of rule refinement, only one type of sub-space is used. This means that in each round of rule refinement, there is only one sub-space to consider. This may result in "forcing" a rule to be refined by using one particular sub-space when using another sub-space

may generate a better rule. Strategy (6), the BestStrategy strategy, partially addresses this problem by selecting the best out of the first five strategies, thus, involving the selection of rules refined by using different sub-spaces. The best rule here is defined as the rule with the highest rule accuracy, calculated using Laplace estimation. A more comprehensive approach to using more than one sub-space is offered by strategies (7) and (8). Strategy (7) is the BestPosRule strategy, which generates two refined versions of a rule; one version using a feature from the UP sub-space and the other using a feature from the Ov sub-space. Both versions are repeatedly refined in the same manner until a stopping condition is met. The rule with the best rule accuracy with Laplace estimation is then selected. Thus, this strategy uses two sub-spaces in conjunction during each round of rule refinement. Note that this strategy will only generate rules without negation as it uses the UP and/or Ov sub-spaces. Strategy (8) is the BestRule strategy. It is an extension of the BestPosRule strategy, where a third version of the rule to be refined is generated using a feature from the UN sub-space. Thus, this strategy uses all three sub-spaces in each round of rule refinement and may generate rules with negation. All three versions of the rule are repeatedly refined in the same manner until a stopping condition is met. Intuitively, using more than one sub-space in conjunction in each round of rule refinement should generate better rules, as several versions of the rule to be refined are generated (until a stopping condition is met) and the best rule selected according to the rule accuracy measure (rule accuracy with Laplace estimation). Each strategy is discussed further in the remainder of this section.

Strategy 1 - UP

Algorithm 6 shows the algorithm for the UP strategy. This strategy refines a rule by adding a feature from the UP sub-space to the rule antecedent if the UP sub-space is not empty. The **addUPFeature** method will add the UP feature with the highest document frequency from the UP sub-space to the rule antecedent. A refined rule with no negation will then be generated. If the UP sub-space does not contain any features, then the rule will not be refined. As mentioned earlier, this prematurely halts the rule refinement but this problem can be addressed using the UP-UN-Ov strategy. One point to note is that a rule refined using the UP feature will only contain two features in its antecedent. This is because the UP feature only appears in positive documents and therefore, a rule refined with the UP feature will not cover any negative documents. Not covering any negative documents is a stopping condition for rule refinement, so rule refinement will stop after adding a UP feature to a rule, resulting in the rule having only two features in its antecedent.

Algorithm 6: Algorithm for the UP strategy

input : R, a rule that needs refinement
output: R', refined rule

```
if UP is not empty then
    R' = addUPFeature(R);
end
else
    R' = R;
end
return R';
```

Strategy 2 - UN

The algorithm for the UN strategy is shown in Algorithm 7. This strategy refines a rule by adding a feature from the UN sub-space to the rule antecedent if the UN sub-space is not empty. The **addUNFeature** method will add the UN feature with the highest document frequency from the UN sub-space to the rule antecedent. Thus, a refined rule with negation will be generated. If the UN sub-space does not contain any feature, then the rule will not be refined. Again, this prematurely halts rule refinement but, as in the case of the UP strategy, this problem can be addressed using the UN-UP-Ov strategy.

Algorithm 7: Algorithm for the UN strategy

input : R, a rule that needs refinement
output: R', refined rule

```
if UN is not empty then
    R' = addUNFeature(R);
end
else
    R' = R;
end
return R';
```

Strategy 3 - Ov

Algorithm 8 shows the algorithm for the Ov strategy. Rule refinement is done by adding an Ov feature from the Ov sub-space to the rule antecedent if the Ov sub-space is not empty. The **addOvFeature** method will add the Ov feature with the highest document frequency difference (i.e. positive document frequency minus negative document frequency) to the rule antecedent. The refined rule generated will be a rule with no negation. One could argue that the Ov feature could be negated if the highest document frequency difference was inverted (i.e. negative document frequency minus positive document frequency). However, if this were done, the refined rule with negation

will negate the positive documents which contain the Ov feature. This of course is contrary to the aim of having a rule cover as many positive documents as possible and therefore, should not be considered. If the Ov sub-space does not contain any feature, then the rule will not be refined.

Algorithm 8: Algorithm for the Ov strategy

```
   input  : R, a rule that needs refinement
   output: R', refined rule
1  if Ov is not empty then
2  |  R' = addOvFeature(R);
3  end
4  else
5  |  R' = R;
6  end
7  return R';
```

Strategy 4 - UP-UN-Ov

Algorithm 9 shows the algorithm for the sequence combination of the UP-UN-Ov strategy. This strategy addresses the empty sub-space problem that may be encountered when using the UP, UN and Ov strategies. Using the UP-UN-Ov strategy, if the UP sub-space is not empty, the **addUPFeature** method will be used to add a UP feature to the rule. However, if the UP sub-space has no features, a UN feature will be added using the **addUNFeature** method. Similarly, if the UN sub-space has no features, then an Ov feature will be added using the **addOvFeature**. Otherwise, the rule will not be refined. In the sequence, Ov comes last because it covers at least one negative document and thus should be considered last if the aim is to generate a rule that can distinguish between classes. However, if the UP sub-space is never empty for each round of the rule refinement, then only the UP sub-space will be used and this will produce the same set of rules as using the UP strategy. Depending on which feature is used for rule refinement, a refined rule generated using the UP-UN-Ov strategy may or may not contain negation.

Strategy 5 - UN-UP-Ov

The algorithm for the UN-UP-Ov strategy is shown in Algorithm 10. This strategy works in the same manner as the UP-UN-Ov strategy, only interchanging the order of UN and UP. Thus, if the UN sub-space is never empty for every round of the rule refinement, then only the UN sub-space will be used and this will produce the same set of rules as using the UN strategy. Again, depending on which feature is used for rule refinement, a refined rule generated using the UN-UP-Ov strategy may or may not contain negation.

Algorithm 9: Algorithm for the UP-UN-Ov strategy

input : R, a rule that needs refinement
output: R', refined rule

```
1  if UP is not empty then
2  |  R' = addUPFeature(R);
3  end
4  else
5  |  if UN is not empty then
6  |  |  R' = addUNFeature(R);
7  |  end
8  |  else
9  |  |  if Ov is not empty then
10 |  |  |  R' = addOvFeature(R);
11 |  |  end
12 |  |  else
13 |  |  |  R' = R;
14 |  |  end
15 |  end
16 end
17 return R';
```

Algorithm 10: Algorithm for the UN-UP-Ov strategy

input : R, a rule that needs refinement
output: R', refined rule

```
1  if UN is not empty then
2  |  R' = addUNFeature(R);
3  end
4  else
5  |  if UP is not empty then
6  |  |  R' = addUPFeature(R);
7  |  end
8  |  else
9  |  |  if Ov is not empty then
10 |  |  |  R' = addOvFeature(R);
11 |  |  end
12 |  |  else
13 |  |  |  R' = R;
14 |  |  end
15 |  end
16 end
17 return R';
```

Strategy 6 - BestStrategy

Algorithm 11 shows the BestStrategy algorithm. The rule to be refined is refined separately, to produce five different versions, using the UP, UN, Ov, UP-UN-Ov and UN-UP-Ov strategies. This strategy then chooses the best rule out of the five rules generated. The **ChooseBestRule** method is used to choose the best rule. The best rule in this case is defined as the rule with the highest rule accuracy using Laplace estimation. This strategy may or may not generate a rule with negation, depending on which rule is chosen as the best rule.

Algorithm 11: Algorithm for the BestStrategy strategy

input : R, a rule that needs refinement
output: R', refined rule

```
1  R1 = RefineWithUP(R);
2  R2 = RefineWithUN(R);
3  R3 = RefineWithOv(R);
4  R4 = RefineWithUP-UN-Ov(R);
5  R5 = RefineWithUN-UP-Ov(R);
6  while R1, R2, R3, R4, R5 need refinement do
7  |   R1 = RefineWithUP(R1);
8  |   R2 = RefineWithUN(R2);
9  |   R3 = RefineWithOv(R3);
10 |   R4 = RefineWithUP-UN-Ov(R4);
11 |   R5 = RefineWithUN-UP-Ov(R5);
12 end
13 R' = ChooseBestRule(R1, R2, R3, R4, R5);
14 return R';
```

Strategy 7 - BestPosRule

The algorithm for the BestPosRule strategy is shown in Algorithm 12. This strategy generates two refined versions of a rule, one version using a feature from the UP sub-space and the other using a feature from the Ov sub-space. Thus, this strategy uses two sub-spaces in conjunction during each rule refinement round and will only generate rules without negation. Both versions are recursively refined in the same manner until a stopping condition is met. The stopping conditions for rule refinement include one of the following:

1. The current rule accuracy (with Laplace estimation) is less than the previous rule accuracy (with Laplace estimation). In this case, the previous rule is selected.
2. The current rule no longer covers any negative documents.
3. There are no more features in the sub-space to be added.

Note that because of the characteristic of the UP feature, which only occurs in positive documents, any rule that is refined with a UP feature will no longer cover any negative documents. This satisfies one of the stopping conditions for rule refinement and, therefore, this rule will not be further refined. Refinement in this recursive manner will stop when a stopping condition is met and the rule with the highest possible accuracy using Laplace estimation is selected. Figure 4.5 graphically demonstrates how the BestPosRule strategy works. In the figure, the rule R is the initial rule to be refined. R is refined with a UP feature to produce R_1 and refined with an Ov feature to produce R_2. R_1 is not further refined as it is refined with a UP feature for the reason described above. R_2 is further refined in the same manner until a stopping condition is met. Each refinement will produce a different rule, R_n (where n = 1, 2, 3, ...).

Algorithm 12: Algorithm for the BestPosRule strategy

input : R, a rule that needs refinement
output: R', refined rule

R_1 = **RefineWithUP**(R);
R_2 = **RefineWithOv**(R);
for $i = 1 \rightarrow 2$ **do**
  while R_i *needs refinement* **do**
    R_i = **recurse**(R_i);
  end
  if *Accuracy*(R_i) > *Accuracy*(R_{best}) **then**
    $R_{best} = R_i$;
  end
end
R'=R_{best};
return R';

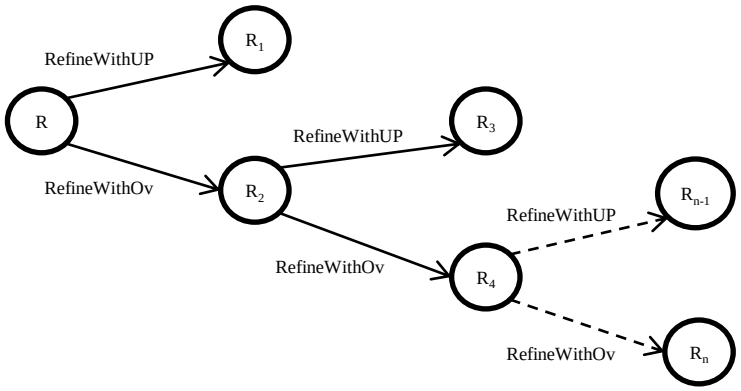

Figure 4.5: Graph conceptualization of rule refinement using the BestPosRule strategy

Strategy 8 - BestRule

Algorithm 13 shows the algorithm for the BestRule strategy. This strategy is an extension of the BestPosRule strategy, where a third version of the rule to be refined is generated using a feature from the UN sub-space. Thus, this strategy uses all three subspaces in conjunction for each round of rule refinement and thus, may generate rules with negation. All three versions are recursively refined until a stopping condition is met. The stopping conditions for rule refinement in this strategy are the same as the stopping conditions for the BestPosRule strategy. The rule with the highest accuracy using Laplace estimation is then selected. Again, any rule that is refined with a UP feature will no longer cover any negative documents and therefore, will not be further refined. Figure 4.6 demonstrates, in a graphical manner, how the BestRule strategy works. Similar to Figure 4.5, R is the initial rule to be refined. It is refined with a UP feature to produce R_1, a UN feature to produce R_2 and an Ov feature to produce R_3. Refinement for each rule is continued in the same manner until one of the stopping conditions is met.

Algorithm 13: Algorithm for the BestRule strategy

```
   input  : R, a rule that needs refinement
   output: R', refined rule
 1 R_1 = RefineWithUP(R);
 2 R_2 = RefineWithOv(R);
 3 R_3 = RefineWithUN(R);
 4 for i = 1 → 3 do
 5 |   while R_i needs refinement do
 6 |   |   R_i = recurse(R_i);
 7 |   end
 8 |   if Accuracy(R_i) >Accuracy(R_best) then
 9 |   |   R_best = R_i;
10 |   end
11 end
12 R'=R_best;
13 return R';
```

Example of the Rule Refinement Strategies

The example presented in Table 4.9 is used to illustrate the rules learnt using the eight rule refinement strategies proposed. From Table 4.9, it is given that the feature set for class *motorcycle* contains three features; *bike*, *ride* and *honda* and that the initial rule learnt is $bike \Rightarrow motorcycle$. In this example, the rule learnt covers three records: two of them are positive documents (from class *motorcycle*), while the other is a negative document (from class *bicycle*). The search space Sp is then divided into three subspaces by identifying the UP, UN and Ov features. The rules learnt using each of the

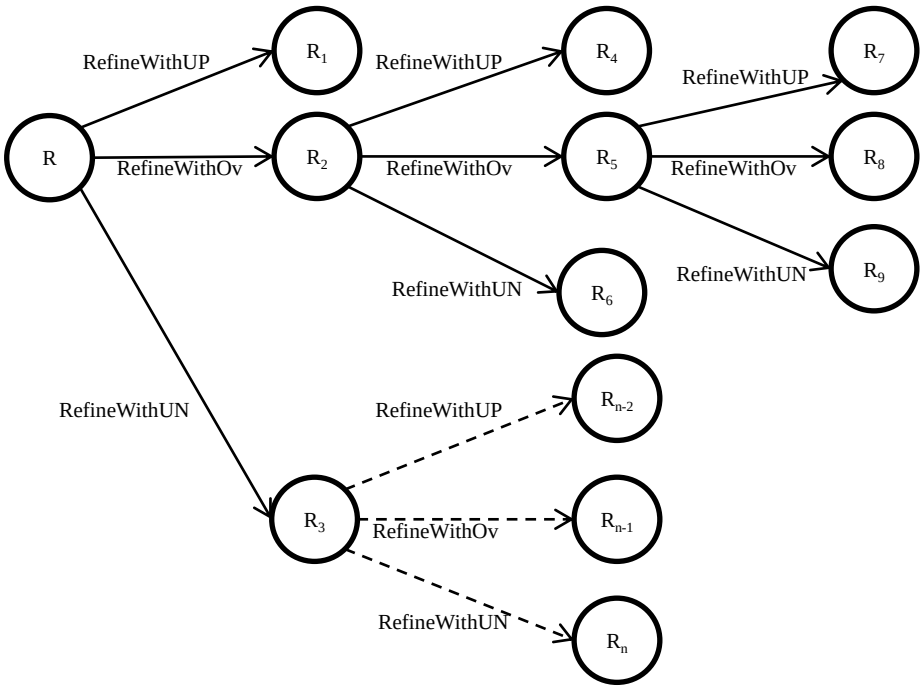

Figure 4.6: Graph conceptualization of rule refinement using the BestRule strategy

rule refinement strategies are also shown in Table 4.9.

4.3.3 Summary of the Rule Refinement Strategies

Eight rule refinement strategies have been proposed based on the division of Sp into three sub-spaces. These strategies were incorporated into the IRL mechanism to learn both rules with and without negation. Strategies UP, UN and Ov used only one sub-space in each round of rule refinement. The specific features in each of these sub-spaces depends on the content of the documents that a rule covers. Therefore, it is possible that a sub-space may be empty. To address the effects of rule refinement stopping prematurely because of the empty sub-space problem, the strategies UP-UN-Ov and UN-UP-Ov were devised. While these two strategies used all three sub-spaces in turn whenever the former was found to be empty, they still used only one sub-space in each round of rule refinement. The BestStrategy strategy chose the rule with the highest accuracy (using Laplace estimation) from the five rules generated using UP, UN, Ov, UP-UN-Ov and UN-UP-Ov. Two more comprehensive strategies were also devised that used more than one sub-space in each round of rule refinement. The BestPosRule strategy used the UP and Ov sub-spaces to generate two refined version of a rule in each round of rule refinement. The BestRule strategy further extended the former by

Example
Feature set for class $motorcycle = \{bike, ride, honda\}$
Initial rule learnt = $bike \Rightarrow motorcycle$

The rule covers three records (two positive records and one negative record)
Record 1 labelled class $motorcycle = \{bike, ride, honda\}$
Record 2 labelled class $motorcycle = \{bike, ride, honda\}$
Record 3 labelled class $bicycle = \{specialized, cruise, bike, ride\}$

Identify UP, UN and Ov features
UP feature(s) = $\{honda\}$
UN feature(s) = $\{specialized, cruise\}$
Ov feature(s) = $\{ride\}$

Rule refinement
Refine with UP: $bike \wedge honda \Rightarrow motorcycle$
Refine with UN: $bike \wedge \neg specialized \Rightarrow motorcycle$
Refine with Ov: $bike \wedge ride \Rightarrow motorcycle$
Refine with UP-UN-Ov: $bike \wedge honda \Rightarrow motorcycle$
Refine with UN-UP-Ov: $bike \wedge \neg specialized \Rightarrow motorcycle$
Refine with BestStrategy: Choose the best rule from the above five strategies
Refine with BestPosRule: $bike \wedge honda \Rightarrow motorcycle$ or $bike \wedge ride \Rightarrow motorcycle$ (whichever is the best)
Refine with BestRule: $bike \wedge honda \Rightarrow motorcycle$ or $bike \wedge \neg specialized \Rightarrow motorcycle$ or $bike \wedge ride \Rightarrow motorcycle$ (whichever is the best)

Table 4.9: Example of rule refinement using the eight proposed strategies

Strategy	Description	Sample rules
UP	Add a UP feature to refine a rule	$a \wedge b \Rightarrow x$
UN	Add a UN feature to refine a rule	$a \wedge \neg c \Rightarrow x$
Ov	Add an Ov feature to refine a rule	$a \wedge b \wedge d \Rightarrow x$
UP-UN-Ov	If UP is not empty, add a UP feature to refine a rule; Else If UN is not empty, add a UN feature to refine a rule; Else If Ov is not empty, add an Ov feature to refine a rule	$a \wedge b \Rightarrow x$
UN-UP-Ov	If UN is not empty, add a UN feature to refine a rule; Else If UP is not empty, add a UP feature to refine a rule; Else If Ov is not empty, add an Ov feature to refine a rule	$a \wedge \neg c \Rightarrow x$
BestStrategy	Choose the best rule from the five rules generated by each UP, UN, Ov, UP-UN-Ov and UN-UP-Ov	$a \wedge b \wedge d \Rightarrow x$
BestPosRule	Recursively generate two versions of a rule; one refined with a UP feature and the other refined with an Ov feature. Choose the best refined rule	$a \wedge b \wedge d \wedge e \Rightarrow x$
BestRule	Recursively generate three versions of a rule; one refined with a UP feature, one refined with a UN feature and the other refined with an Ov feature. Choose the best refined rule	$a \wedge b \wedge \neg c \wedge \neg f \Rightarrow x$

Table 4.10: Summary of the rule refinement strategies

generating three refined version of a rule in each round of rule refinement using all three sub-spaces. Table 4.10 shows a summary of the rule refinement strategies.

All the proposed strategies generated different kinds of rules (see examples given in Table 4.9). Some strategies like UP, Ov and BestPosRule only generate rules without negation, while it is possible for UN, UP-UN-Ov, UN-UP-Ov, BestStrategy and BestRule to generate rules with negation. This mix of strategies allowed for experimentation using IRL, both with and without negation, so as to investigate the effectiveness of using negation in the context of IRL for text classification.

4.3.4 The Issue of Overfitting

The issue of overfitting is a common cause for concern in the field of classification including text classification. In this context, overfitting is defined as the learning of a classifier that is very specifically tailored to a particular dataset and which, when applied to another dataset from the same domain, will perform very poorly. Most classification systems include a mechanism to handle overfitting. For example, RIPPER uses rule pruning methods to avoid overfitting. The proposed IRL mechanism handles

overfitting in a variety of manners depending on which rule refinement strategy is used. Table 4.11 describes the rule refinement strategies with regards to the issue of overfitting.

4.4 Classification

Classification is the last process in the text classification framework as shown in Figure 4.1. Both binary classification and multi-class classification are conducted in the experiments and are therefore discussed further in Sub-sections 4.4.1 and 4.4.2 respectively. The evaluation of the proposed IRL mechanism and the results obtained are discussed in more detail in Chapters 6 and 7.

4.4.1 Binary Classification

In binary classification for IRL, the induced classifier contains only rules directed at one class c that features in the dataset. These rules are used to classify documents as either belonging to this class or not. Thus, by implication, binary classification equates to two class classification. In order to determine which rule to fire when classifying a document, the rules are ordered. The higher order rules will be fired before the lower order rules. In the proposed framework, the primary ordering was according to descending rule length (the number of features in the rule *antecedent*) so that more specific rules would be fired first; the secondary ordering was according to descending rule weight (the number of documents the rule covered in the learning phase). This means that more specific rules with higher coverage will be ranked higher than less specific and lower coverage rules. A default rule, where the *conclusion* was the default class, was used to classify a document when all the other rules did not cover this document.

Given a binary classification problem, the classification outcomes can be presented in the form of a *confusion matrix* comprising the number of true positive (TP), false positive (FP), false negative (FN) and true negative (TN). This confusion matrix can then be used to evaluate the classifier as a whole. If a rule r in $Ruleset_c$ covers a document d_x, where x is the class label for d and $x = c$, then the classification result is a TP. If a rule r in $Ruleset_c$ covers a document d_x but $x \neq c$, then the classification result is a FP. If no rules in $Ruleset_c$ covers a document d_x and $x = c$, then the classification result is a FN. Lastly, if no rules in $Ruleset_c$ covers a document d_x and $x \neq c$, then the classification result is a TN. Algorithm 14 shows the algorithm for binary classification.

A confusion matrix, with its TP, FP, FN and TN values, is typically used to calculate the precision (P) and recall (R) values for each class in the dataset. P and R are then used to calculate the F_1-measure to evaluate the performance of the overall

Strategy	Description
UP	A rule refined using the UP feature will always have two features in its antecedent. The nature of the UP feature is such that it only appears in positive documents, therefore, any rule refined with a UP feature will no longer cover negative documents. Thus, rule refinement will stop after adding a UP feature to a rule, generating a rule that has an antecedent comprising two features. This prevents the rule from being too precise and overfitting the data.
UN	A rule will not be refined further if the rule no longer covers negative documents. Further stopping conditions for refinement are when there are no more UN features or when the previous rule accuracy is higher than the current rule accuracy. These conditions contribute to the avoidance of overfitting by preventing the rule learnt from being too precise.
Ov	Rule refinement will stop when there are no more Ov features or when the previous rule accuracy is higher than the current rule accuracy. The nature of the Ov feature sub-space is such that it appears in both positive and negative documents. Thus, the use of Ov for refinement will generate a rule that covers at least one negative document. This reduces the likelihood of the rule learnt from overfitting the data.
UP-UN-Ov	This strategy for refinement was devised to overcome the empty sub-space problem. When the UP sub-space is empty, the UN sub-space is used to add a UN feature to a rule. If however, the UN sub-space is also empty, the Ov sub-space will be used instead. Overfitting is consequently avoided for the same reasons as presented in the discussion given above for each of the individual strategies.
UN-UP-Ov	This strategy for refinement was devised to overcome the empty sub-space problem. When the UN sub-space is empty, the UP sub-space is used to add a UP feature to a rule. If however, the UP sub-space is also empty, the Ov sub-space will be used instead. Overfitting is again avoided for the reasons presented in the discussion given above for each of the individual strategies.
BestStrategy	This strategy chooses the rule with the best rule accuracy (with Laplace estimation) from the above five strategies. The issue of overfitting is addressed individually as per each strategy.
BestPosRule	This strategy includes a more exhaustive search than those associated with the previous strategies to find the best rule for refinement using UP and Ov in each round of refinement. Again, the issue of overfitting is addressed individually by the use of UP or Ov.
BestRule	This strategy includes an even more exhaustive search than all of the above strategies to find the best rule refined using UP, UN and Ov for each round of refinement. Again, the issue of overfitting is addressed individually according to the adopted strategy of UP, UN or Ov.

Table 4.11: Summary of the rule refinement strategies with regards to overfitting

Algorithm 14: Algorithm for binary classification

input : $Ruleset_c$, a classifier for class c
$Testset$, the set of documents from dataset, D to test the classifier
output: $Results$, the classification results in the form of confusion matrix

```
1  TP = 0;
2  FP = 0;
3  FN = 0;
4  TN = 0;
5  for each document d_x, where x is the class label for d in Testset do
6  |  Classify d_x using Ruleset;
7  |  if r ∈ Ruleset_c covers d_x and x = c then
8  |  |  TP = TP + 1;
9  |  end
10 |  else
11 |  |  if r ∈ Ruleset_c covers d_x and x ≠ c then
12 |  |  |  FP = FP + 1;
13 |  |  end
14 |  |  else
15 |  |  |  if no rules in Ruleset_c covers d_x and x = c then
16 |  |  |  |  FN = FN + 1;
17 |  |  |  end
18 |  |  |  else
19 |  |  |  |  if no rules in Ruleset_c covers d_x and x ≠ c then
20 |  |  |  |  |  TN = TN + 1;
21 |  |  |  |  end
22 |  |  |  end
23 |  |  end
24 |  end
25 end
26 Results = ConfusionMatrix(TP, FP, FN, TN);
27 return Results;
```

classification. Where binary classification is used to achieve multi-class classification, micro-averaging is used to evaluate the classification of the dataset as a whole. The formulas for calculating P, R and F_1 were shown in Section 2.7 in Chapter 2 and are repeated as follows for the benefit of the reader:

$$P = \frac{TP}{TP + FP} \tag{4.1}$$

$$R = \frac{TP}{TP + FN} \tag{4.2}$$

$$F_1 = \frac{2PR}{P + R} \tag{4.3}$$

4.4.2 Multi-Class Classification

In contrast to binary classification, in multi-class classification, the induced classifier consists of rules for all the classes in the dataset. These rules are used to classify documents as belonging to one of all the available classes. As already noted, multi-class classification can be handled by using separate binary classifiers individually, one for each class. However, it is also possible for multi-class classification to be handled directly by classifying all the documents in a dataset using a single classifier that includes rules for all the classes (including a default rule with the most numerous class as its conclusion). This is a more challenging approach and is adopted with respect to multi-class classification in the proposed framework. The same rule ordering strategy as described in Sub-section 4.4.1 was used to order the rules in the classifier. If a rule r_c in *Ruleset* covers a document d_x and $x = c$, then it is a correct classification. If a rule r_c in *Ruleset* covers a document d_x and $x \neq c$, then it is an incorrect classification. If no rules in *Ruleset* covers a document d_x and $x = defaultClass$, then it is a correct classification. Lastly, if no rules in *Ruleset* cover a document d_x and $x \neq defaultClass$, then it is an incorrect classification. Algorithm 15 shows the algorithm for multi-class classification evaluation adopted with respect to the proposed framework.

The numbers of correct and incorrect classifications are used to calculate the accuracy of the classification of the dataset. This accuracy measure is then used as the evaluation measure for the multi-class classifier.

4.5 Summary of Framework

This chapter has described the proposed framework for IRL-based text classification. The text classification framework was described with respect to the three principal processes that make up the overall IRL-based text classification procedure. The three

Algorithm 15: Algorithm for multi-class classification

input : $Ruleset_c$, a classifier for class c
$Testset$, the set of documents from dataset, D to test the classifier
output: $Results$, the classification results in the form of accuracy

```
1  correctClassification = 0;
2  incorrectClassification = 0;
3  for each document d_x, where x is the class label for d in Testset do
4  |  Classify d_x using Ruleset;
5  |  if r_c ∈ Ruleset covers d_x and x = c then
6  |  |  correctClassification = correctClassification + 1;
7  |  end
8  |  else
9  |  |  if r_c ∈ Ruleset covers d_x and x ≠ c then
10 |  |  |  incorrectClassification = incorrectClassification + 1;
11 |  |  end
12 |  |  else
13 |  |  |  if no rules in Ruleset covers d_x and x = defaultClass then
14 |  |  |  |  correctClassification = correctClassification + 1;
15 |  |  |  end
16 |  |  |  else
17 |  |  |  |  if no rules in Ruleset covers d_x and x ≠ defaultClass then
18 |  |  |  |  |  incorrectClassification = incorrectClassification + 1;
19 |  |  |  |  end
20 |  |  |  end
21 |  |  end
22 |  end
23 end
24 Results = Accuracy(correctClassification, incorrectClassification);
25 return Results;
```

processes were: (i) document preprocessing, (ii) IRL and (iii) classification. In document preprocessing, five further sub-processes were identified, namely: (i) data cleaning, (ii) keyword extraction, (iii) phrase extraction, (iv) feature selection and (v) text representation. In phrase extraction, three different kinds of phrase were extracted: (i) n-grams, (ii) keyphrases and (iii) fuzzy phrases. The discussion concerning feature selection also included the consideration of two well known techniques, namely, χ^2 and IG. These techniques were used to calculate the discriminative values of keywords and/or phrases and a pre-determined percentage was used to select keywords and/or phrases as features for text representation. In the context of text representation, both the bag-of-words and the bag-of-phrases representations were discussed. In addition, the ARFF format used with respect to the WEKA machine learning workbench, was also discussed. The core of the proposed framework is the IRL mechanism. This was thus described in more detail. The description of the proposed IRL mechanism included consideration of the feature space division process and eight rule refinement strategies. Lastly, the classification process was described in the context of two classification formulations: binary classification and multi-class classification. In the following chapter, Chapter 5, the datasets used to evaluate the proposed IRL mechanism are described. The experimental setup, results and evaluation of the proposed IRL mechanism is then discussed in Chapters 6 and 7, whereby keywords and phrases are used as the text representation respectively.

Chapter 5

Datasets

This chapter describes the datasets used for evaluating the proposed framework. In total, eight datasets were used to evaluate the work described. The first two were the popular 20 Newsgroups dataset [50] and the Reuters-21578, Distribution 1.0[1] dataset [56]. These two datasets have been widely used in the text classification literature. The third dataset was taken from the Veterinary Science domain, the Small Animals Veterinary Surveillance Network (SAVSNET) dataset [74], which comprises a set of questionnaire returns. This is a private dataset from the SAVSNET project[2]. The last five datasets were taken from the UCI Machine Learning repository[3] [33]. Although the UCI datasets are not unstructured text-based, they were selected so as to evaluate the proposed IRL mechanism in terms of general data mining. Sections 5.1, 5.2, 5.3 and 5.4 describe the 20 Newsgroups, Reuters-21578, SAVSNET and UCI datasets respectively. Section 5.5 summarizes this chapter.

5.1 20 Newsgroups Dataset

The 20 Newsgroups dataset is a collection of 19,997 documents, comprising news articles from 20 classes. Each class corresponds to a different topic, some are closely related while others are distinct. There are 1,000 documents in each class with the exception of one class that contains 997 documents. The list of classes in the 20 Newsgroups dataset and the number of documents in each class is shown in Table 5.1.

In the experiments conducted for this thesis, this dataset was split into two non-overlapping datasets (hereafter, referred to as 20NG-A and 20NG-B), each comprising 10 classes (20NG-A has 10,000 documents and 20NG-B has 9,997 documents). This dataset is split only for computational efficiency reasons, as reported in Wang [89], by taking 10 classes for 20NG-A and the remaining 10 classes for 10NG-B. Therefore, 20NG-A and 20NG-B should be viewed as two separate datasets and the results should

[1] available at http://www.daviddlewis.com/resources/testcollections/reuters21578/
[2] http://www.liv.ac.uk/savsnet/
[3] available at http://archive.ics.uci.edu/ml/

be considered in this context. The classes for 20NG-A and 20NG-B are shown in Table 5.2.

Class name	Number of documents
comp.graphics	1,000
comp.os.ms-windows.misc	1,000
comp.sys.ibm.pc.hardware	1,000
comp.sys.mac.hardware	1,000
comp.windows.x	1,000
rec.autos	1,000
rec.motorcycles	1,000
rec.sport.baseball	1,000
rec.sport.hockey	1,000
sci.crypt	1,000
sci.electronics	1,000
sci.med	1,000
sci.space	1,000
misc.forsale	1,000
talk.politics.misc	1,000
talk.politics.guns	1,000
talk.politics.mideast	1,000
talk.religion.misc	1,000
alt.atheism	1,000
soc.religion.christian	997
total	19,997

Table 5.1: The classes and number of documents in the 20 Newsgroups dataset

20NG-A	20NG-B
alt.atheism	comp.graphics
comp.sys.ibm.pc.hardware	comp.os.ms-windows.misc
comp.windows.x	comp.sys.mac.hardware
misc.forsale	rec.autos
rec.motorcycles	rec.sport.hockey
rec.sport.baseball	sci.crypt
sci.electronics	sci.space
sci.med	soc.religion.christian
talk.politics.mideast	talk.politics.guns
talk.religion.misc	talk.politics.misc

Table 5.2: The classes in 20NG-A and 20NG-B

5.2 Reuters-21578 Dataset

The Reuters-21578, Distribution 1.0 dataset is a collection of news stories. These documents came from the Reuters newswire service and were collected in 1987. There are five category sets in this collection, namely, "Exchanges", "Orgs", "People", "Places" and "Topics". The category set that has been used in almost all previous research is the "Topics" set, which contains general economic subjects. There are 135 classes and 21,578 documents in the "Topics" set. While the ModApté split [4] and ModLewis [55] split are two popular approaches used to split the dataset for testing purposes, the dataset is not split according to them in the context of the work conducted in this thesis. Rather, Ten-fold Cross Validation (TCV) is applied to split the dataset in ten almost equal parts and each part is used in turn for testing, as mentioned in Section 2.2 in Chapter 2.

The preprocessing of this dataset follows that of Wang [89]. First, the top ten most populated classes were selected. From these classes, the multi-labelled and non-text documents were removed. This left the dataset with only eight classes and 6,643 documents. Hereafter, this dataset is referred to as Reuters-8. The class labels and the number of documents in each class is shown in Table 5.3.

Class name	Number of documents
acq	2108
crude	444
earn	2736
grain	108
interest	216
money-fx	432
ship	174
trade	425

Table 5.3: The classes and number of documents in the Reuters-8 dataset

5.3 SAVSNET Dataset

The SAVSNET dataset is a set of questionnaires collected by the SAVSNET project. The aim of the project is to determine the disease status of small animals (mostly cats and dogs). Each questionnaire comes from a single veterinary consultation. The questionnaires comprise both tabular (closed-ended questions) and free text (open-ended questions) data. The data that is of interest in this thesis is the free text data. These are notes from the veterinary practitioner which include diagnostics, treatments and prescriptions. Many of the questionnaire returns have been hand-annotated by domain experts. For the purpose of constructing an evaluation dataset for use with the

proposed IRL mechanism, four such annotations (class labels) were selected: aggression, diarrhoea, pruritus and vomit.

Originally, 27,072 questionnaires were collected. However, 26,039 of them did not include the identified class labels, leaving 1,033 questionnaires. Out of these, 89 questionnaires were found to be repetitions and were also removed. The remaining 944 questionnaires each had one of the four identified class labels. Out of these, 116 questionnaires had their open-ended question left blank, thus, leaving only 828 questionnaires with free text data [39]. The distribution of the class labels across these 828 questionnaires is shown in Table 5.4.

Class name	Number of questionnaires
aggression	34
diarrhoea	308
pruritus	350
vomit	136
total	828

Table 5.4: The classes and number of questionnaires with free text in the SAVSNET dataset

5.4 UCI Machine Learning Repository Datasets

The UCI Machine Learning Repository datasets are well-known in the data mining community. These datasets have been used for numerous data mining evaluations. In the experiments reported in this thesis, five datasets from the UCI Machine Learning Repository were used, namely, "Iris", "Adult", "Wine" "Breast" and "Car". They were chosen because they are the top five most popular datasets, based on the statistics shown on the UCI Machine Learning Repository webpage[4] as of the end of 2011.

The "Iris" dataset is used to predict the type of iris plant. It has three classes (*Iris Setosa, Iris Versicolour, Iris Virginica*) with 50 instances each. The "Adult" dataset is used to predict whether income of an adult exceeds \$50,000 a year based on census data. The class labels are thus: >\$50,000 and ≤\$50,000. There are 48,842 instances in this dataset. The "Wine" dataset is used to predict the type of wine. It has three classes $(1, 2, 3)$ and 178 instances. The "Breast" dataset [61] contains data to predict the diagnosis of breast cancer. There are two classes (*Malignant, Benign*) and 699 instances. The "Car" dataset contains data to evaluate car acceptability. There are four classes (*unacc, acc, good, vgood*) and 1,728 instances. Some statistical information about each dataset is shown in Table 5.5.

[4] http://archive.ics.uci.edu/ml/

In their original form, the five datasets are in tabular form. Despite the experiments being directed at unstructured text data, these five datasets were used for further evaluation of the proposed framework (but using tabular data). The five datasets in their original form were first discretized using the LUCS-KDD Data Discretization/Normalization Software [18] so that the format is suitable for input into the proposed framework. Discretization here refers to the process of converting continuous values into discrete values. The datasets and details of discretization can be found here [17]. Tables 5.6, 5.7, 5.8, 5.9 and 5.10 show the class distribution for the "Iris", "Adult", "Wine", "Breast" and "Car" discretized datasets respectively.

Dataset name	Number of attributes	Number of instances	Number of classes
Iris	4	150	3
Adult	14	48,842	2
Wine	13	178	3
Breast	10	699	2
Car	6	1,728	4

Table 5.5: The statistical information concerning the five UCI Machine Learning Repository datasets

Class	Number of instances	% of total
Iris Setosa	50	33.33
Iris Versicolour	50	33.33
Iris Virginica	50	33.33
Total	150	100.00

Table 5.6: The class distribution for "Iris" dataset

Class	Number of instances	% of total
>\$50,000	11,687	23.93
≤\$50,000	37,155	76.07
Total	48,842	100.00

Table 5.7: The class distribution for "Adult" dataset

Class	Number of instances	% of total
1	59	33.15
2	71	39.89
3	48	26.97
Total	178	100.00

Table 5.8: The class distribution for "Wine" dataset

Class	Number of instances	% of total
Benign	458	65.52
Malignant	241	34.48
Total	699	100.00

Table 5.9: The class distribution for "Breast" dataset

Class	Number of instances	% of total
unacc	1,210	70.02
acc	384	22.22
good	69	3.99
vgood	65	3.76
Total	1728	100.00

Table 5.10: The class distribution for "Car" dataset

5.5 Summary

This chapter has described the datasets used in the experiments conducted with respect to the proposed framework and reported later in this thesis. Three of the datasets used comprised documents with unstructured text: 20 Newsgroups, Reuters-8 and SAVSNET. A further five datasets taken from the UCI Machine Learning Repository comprised tabular data, which was discretized: "Iris", "Adult", "Wine", "Breast" and "Car". All these datasets were used in the experiments reported in Chapter 6, where the use of keywords as the text representation is considered. Only the 20 Newsgroups, Reuters-8 and SAVSNET datasets were used in the experiments reported in Chapter 7, which discussed the use of phrases as the text representation. The UCI Machine Learning Repository datasets were used only with the keywords representation, as in terms of data mining, the discretized attributes could be treated as "keywords" but not phrases.

Chapter 6

Experimental Setup, Results and Evaluation Using Keywords for Inductive Rule Learning with Negation

This chapter describes the experimental setup, results and evaluation using single keywords in the bag-of-words representation with respect to the proposed Inductive Rule Learning (IRL) mechanism. The text classification task was performed on the 20 Newsgroups, Reuters-8, SAVSNET and UCI Machine Learning Repository datasets as described in Chapter 5.

In all the experiments, Ten-fold Cross Validation (TCV) was adopted. Chi-Square (χ^2) and Information Gain (IG) were used as the feature selection techniques to select a subset of keywords to be used as features. A pre-determined percentage of 10% (90% reduction) of the total number of features in each class was used for the selection. For each dataset, both binary and multi-class classification were performed. In the binary classification task, the eight strategies in the proposed IRL mechanism were compared with one another and the best strategy was then compared with: (i) the Sequential Minimal Optimization (SMO) algorithm; (ii) Naive Bayes (NB); (iii) JRip; (iv) OlexGreedy and (v) OlexGA. The evaluation measure used was the micro-averaged F_1-measure, as an averaging measure across all classes in a dataset. In the context of multi-class classification, the eight strategies in the proposed IRL mechanism were again compared with one another, in addition to comparison with the Total From Partial Classification (TFPC) algorithm (with default thresholds: support = 0.1% and confidence = 35.0%), which is a Classification Association Rule Mining (CARM) algorithm, using its own keyword selection strategy (hereafter referred to as TFPC-Keywords). The evaluation measure used in this case was accuracy (Acc). Sections 6.1, 6.2, 6.3 and 6.4 discuss the classification results obtained using the 20 Newsgroups, Reuters-8, SAVSNET and UCI Machine Learning Repository datasets respectively.

6.1 Experiments Using the 20 Newsgroups Dataset

This section discusses the evaluation of the eight strategies in the proposed IRL mechanism using the 20 Newsgroups dataset. As described in Section 5.1 in Chapter 5, the 20 Newsgroups dataset was split into two non-overlapping datasets for computational efficiency purposes. Keywords were extracted as described in Sub-section 4.2.2 in Chapter 4 except in the case of the TFPC-Keywords algorithm where the keyword selection strategy built into the TFPC algorithm was used. The number of features used for the representation of each class (10% of the total number of features) with respect to the 20 Newsgroups dataset is shown in Table 6.1. Sub-sections 6.1.1 and 6.1.2 discuss the results for the binary and multi-class classification of the 20 Newsgroups dataset respectively.

Class name	Number of features
20NG-A	
alt.atheism	1,442
comp.sys.ibm.pc.hardware	1,069
comp.windows.x	1,713
misc.forsale	1,150
rec.motorcycles	1,205
rec.sport.baseball	1,086
sci.electronics	1,208
sci.med	1,837
talk.politics.mideast	1,922
talk.religion.misc	1,597
total	14,229
20NG-B	
comp.graphics	1,395
comp.os.ms-windows.misc	2,709
comp.sys.mac.hardware	1,062
rec.autos	1,231
rec.sport.hockey	1,250
sci.crypt	1,539
sci.space	1,629
soc.religion.christian	1,640
talk.politics.guns	1,676
talk.politics.misc	1,769
total	15,900

Table 6.1: The number of features used for the representation of each class in the 20 Newsgroups dataset

6.1.1 Binary Classification

This sub-section details the evaluation of the results from the binary classification task. The strategies in the proposed IRL mechanism are denoted as RL with the identifier for the different rule refinement strategies used appended. Table 6.2 shows the micro-averaged F_1-measure and the average number of rules generated by the eight strategies in the proposed IRL mechanism using χ^2 and IG for the 20NG-A dataset. Table 6.4 shows the same results but for the 20NG-B dataset. The micro-averaged F_1-measure for the other machine learning techniques in comparison with the best RL strategy for classification of the 20NG-A and 20NG-B datasets are shown in Tables 6.3 and 6.5 respectively.

Strategies	F_1	Average # of rules	Average # of rules with negation	% of rules with negation
χ^2				
RL + UP	0.800	1783.8	0.0	0.0
RL + UN	0.810	997.6	366.1	36.7
RL + Ov	0.803	1601.7	0.0	0.0
RL + UP-UN-Ov	0.800	1786.4	5.7	0.3
RL + UN-UP-Ov	0.810	1011.8	372.7	36.8
RL + BestStrategy	**0.830**	1065.0	224.5	21.1
RL + BestPosRule	0.824	1226.0	0.0	0.0
RL + BestRule	0.821	1112.1	402.3	36.2
IG				
RL + UP	0.759	1122.6	0.0	0.0
RL + UN	0.794	652.5	321.0	49.2
RL + Ov	0.785	998.0	0.0	0.0
RL + UP-UN-Ov	0.759	1124.3	0.6	0.0
RL + UN-UP-Ov	0.794	657.9	322.5	49.0
RL + BestStrategy	**0.815**	649.8	205.6	31.6
RL + BestPosRule	0.802	798.9	0.0	0.0
RL + BestRule	0.803	666.5	329.7	49.5

Table 6.2: Micro-averaged F_1-measure and the average number of rules for the classification of the 20NG-A dataset using χ^2 and IG for feature selection and the keyword representation

Inspection of Table 6.2 indicates that RL + BestStrategy was the best overall strategy with a F_1-measure value that was slightly higher than the rest of the strategies. This strategy is one of the strategies that could generate rules with negation. Out of the rules generated using the RL + BestStrategy, 21.1% (when χ^2 was used) and 31.6% (when IG was used) of the rules incorporated negation. With regards to the first three strategies, which utilized only one sub-space each, the best out of the three was the RL + UN strategy, which also generated rules with negation. In addition, RL +

UN had the smallest ruleset, an average of 997.6 rules, out of which 36.7% were rules with negation. Similarly, when IG was used, RL + UN also had a higher F_1-measure when comparing the first three strategies, in addition to having the smallest ruleset, an average of 652.5 rules, out of which 49.2% were rules with negation. Regardless of the feature selection technique used, the worst strategies were RL + UP and RL + UP-UN-Ov, which also featured the largest rulesets.

Techniques	χ^2	Rank	IG	Rank
RL	0.830	2	0.815	3
SMO	0.849	1	0.842	1
NB	0.636	6	0.661	6
JRip	0.760	5	0.756	5
OlexGreedy	0.824	3	0.821	2
OlexGA	0.817	4	0.813	4

Table 6.3: Micro-averaged F_1-measure for the classification of the 20NG-A dataset using χ^2 and IG for feature selection and the keyword representation for the best RL strategy in comparison with the other machine learning techniques

When the best RL strategy was compared with the other machine learning techniques, as shown in Table 6.3, it ranked second behind SMO when χ^2 was used and ranked third behind SMO and OlexGreedy when IG was used. While the first top four techniques had a F_1-measure value of more than 0.800, both JRip and NB did not perform well, with NB performing the worst out of all the techniques. When comparing only the rule-based techniques (not including SMO and NB), the RL strategy came in best and second best when χ^2 and IG were used respectively. Recall that rule-based techniques offer the advantage that they are more readily understandable by the end user.

In the classification of the 20NG-B dataset, the best strategy was RL + BestRule, followed closely by RL + BestStrategy when χ^2 was used as shown in Table 6.4. Both these strategies generated rules with negation. In the case of RL + BestRule, 37.7% of the generated rules incorporated negation, while in the case of RL + BestStrategy, 19.3% of the rules incorporated negation, as can be seen from Table 6.4. The worst strategy was RL + UN-UP-Ov when χ^2 was used. Looking at the first three strategies that utilized only one sub-space, the best strategy was RL + UP, which did not generate any rules with negation. RL + UP had a slightly higher F_1-measure value compared to RL + UN but this was achieved with a much larger ruleset; an average of 1707.0 rules as compared to RL + UN, which had an average of 837.7 rules (less than half of RL + UP's). When IG was used, the best strategy was RL + BestStrategy, which generated 25.7% of rules with negation. This was followed closely by RL + BestRule,

which generated a ruleset comprising 50.6% of rules with negation. The worst strategy was RL + Ov. With respect to the first three strategies which used only one sub-space, RL + UN was again slightly better, with an average number of rules of only 625.2 with 47.3% of the rules incorporating negation.

Strategies	F_1	Average # of rules	Average # of rules with negation	% of rules with negation
χ^2				
RL + UP	0.844	1707.0	0.0	0.0
RL + UN	0.825	837.7	305.4	36.5
RL + Ov	0.824	1481.1	0.0	0.0
RL + UP-UN-Ov	0.844	1706.3	1.9	0.1
RL + UN-UP-Ov	0.823	848.6	310.9	36.6
RL + BestStrategy	0.861	1062.7	205.2	19.3
RL + BestPosRule	0.858	1197.9	0.0	0.0
RL + BestRule	**0.862**	1058.7	399.1	37.7
IG				
RL + UP	0.819	1238.7	0.0	0.0
RL + UN	0.821	625.2	295.6	47.3
RL + Ov	0.809	1089.6	0.0	0.0
RL + UP-UN-Ov	0.819	1238.5	1.6	0.1
RL + UN-UP-Ov	0.821	625.2	295.6	47.3
RL + BestStrategy	**0.852**	698.6	179.8	25.7
RL + BestPosRule	0.842	911.6	0.0	0.0
RL + BestRule	0.850	713.7	360.8	50.6

Table 6.4: Micro-averaged F_1-measure and the average number of rules for the classification of the 20NG-B dataset using χ^2 and IG for feature selection and the keyword representation

Techniques	χ^2	Rank	IG	Rank
RL	0.862	2	0.852	2
SMO	0.892	1	0.891	1
NB	0.656	6	0.672	6
JRip	0.808	5	0.812	5
OlexGreedy	0.845	3	0.844	3
OlexGA	0.844	4	0.837	4

Table 6.5: Micro-averaged F_1-measure for the classification of the 20NG-B dataset using χ^2 and IG for feature selection and the keyword representation for the best RL strategy in comparison with the other machine learning techniques

In comparison with the other machine learning techniques as shown in Table 6.5, the best RL strategy came in second behind SMO. It performed quite similarly to the

Olex systems. NB was again the worst performing technique in this case. Comparison of only the rule-based techniques (not including SMO and NB) showed that the RL strategy was the best.

Overall, with regards to the binary classification task for the 20 Newsgroups datasets, the better RL strategies were ones which generated rules with negation. With respect to the feature selection techniques used, when IG was adopted, the percentage of rules with negation was higher, except for RL + UP-UN-Ov in the 20NG-A dataset. Moreover, it was observed that the average number of rules generated was much smaller for the RL strategies when IG was used as compared to when χ^2 was used. χ^2 and IG both used different computations to determine the significance of a feature as a keyword. Therefore, the choice of features ranked in the top 10% was different. Generally, when a keyword that occurs in more documents is used as a feature for rule learning, the rule learnt has a wider coverage (covering more documents), resulting in fewer rules needed to cover all the documents in a dataset. This suggested that when IG was used, there were more rules learnt which covered two or more documents whereas the features deemed more significant by χ^2 led to the learning of more rules that covered only one document, thus resulting in a bigger ruleset. However, the use of χ^2 led to better classification results in terms of higher F_1-measure values obtained. Comparison with the other machine learning techniques showed that the best RL strategy outperformed the other machine learning techniques and was closely competitive with SMO.

6.1.2 Multi-Class Classification

This sub-section discusses the results of evaluating the eight strategies in the proposed IRL mechanism in the context of multi-class classification for the 20 Newsgroups dataset. The eight strategies in the proposed IRL mechanism were compared with the TFPC-Keywords algorithm. Table 6.6 shows the average accuracy obtained and the average number of rules generated with respect to the 20NG-A dataset for the eight strategies in the proposed IRL mechanism using χ^2 and IG in comparison with the TFPC-Keywords algorithm. Tables 6.7 shows the same information for the classification of the 20NG-B dataset.

The results from the multi-class classification of the 20NG-A dataset presented in Table 6.6 showed that the best strategy was RL + Ov, which was a strategy that did not generate rules with negation, regardless of the feature selection technique used. The worst RL strategy was RL + UP-UN-Ov (when χ^2 was used) and both RL + UP and RL + UP-UN-OV (when IG was used). The difference in accuracies between the best and worst strategies was 7.7% (when χ^2 was used) and 10.8% (when IG was used), indicating that an additional average of 770.0 and 1,080.0 (out of 10,000) documents were correctly classified by RL + Ov. In comparison with TFPC-Keywords, RL + Ov was 5.2% more accurate when χ^2 was used and 2.1% more accurate when IG

was used, indicating that it covered an additional average of 520.0 and 210.0 (out of 10,000) documents. It should be noted however, that the TFPC-Keywords algorithm did not determine keywords in the same way as used in the IRL mechanism, as it used its own keyword selection strategy. Despite the RL strategies which generated rules with negation not performing as well as RL + Ov, they were also better than TFPC-Keywords in terms of the average accuracy and the average number of rules. The RL + UP and RL + UP-UN-Ov strategies were less effective in comparison.

Strategies	Average Acc (%)	Average # of rules	Average # of rules with negation	% of rules with negation
χ^2				
RL + UP	73.6	1783.8	0.0	0.0
RL + UN	79.3	997.6	366.1	36.7
RL + Ov	**81.2**	1601.7	0.0	0.0
RL + UP-UN-Ov	73.5	1786.4	5.7	0.3
RL + UN-UP-Ov	79.1	1011.8	372.7	36.8
RL + BestStrategy	78.6	1065.0	224.5	21.1
RL + BestPosRule	79.8	1226.0	0.0	0.0
RL + BestRule	77.8	1112.1	402.3	36.2
IG				
RL + UP	67.3	1122.6	0.0	0.0
RL + UN	76.7	652.5	321.0	49.2
RL + Ov	**78.1**	998.0	0.0	0.0
RL + UP-UN-Ov	67.3	1124.3	0.6	0.1
RL + UN-UP-Ov	76.7	657.9	322.5	49.0
RL + BestStrategy	76.5	649.8	205.6	31.6
RL + BestPosRule	76.4	798.9	0.0	0.0
RL + BestRule	75.1	666.5	329.7	49.5
TFPC-Keywords	76.0	1582.9	0.0	0.0

Table 6.6: Average accuracy and the average number of rules for the classification of the 20NG-A dataset using χ^2 and IG for feature selection and the keyword representation in comparison with the TFPC-Keywords algorithm

Results similar to that of the 20NG-A dataset were observed for the 20NG-B dataset as shown in Table 6.7. RL + Ov was again the best performing RL strategy while the worst RL strategies were RL + UP and RL + UP-UN-Ov. The difference in accuracies between the best and worst RL strategies was 5.7% (when χ^2 was used) and 7.4% (when IG was used), indicating that an additional average of 569.8 and 739.8 (out of 9,997) documents were correctly classified by RL + Ov. The best RL strategy was 5.2% better than TFPC-Keywords, which translated to an additional average of 519.8 (out of 9,997) documents being correctly classified. In fact, all the RL strategies except for RL + UP and RL + UP-UN-Ov performed better than TFPC-Keywords in terms of the average accuracy and the average number of rules.

Strategies	Average Acc (%)	Average # of rules	Average # of rules with negation	% of rules with negation
χ^2				
RL + UP	78.7	1707.0	0.0	0.0
RL + UN	79.6	837.7	305.4	36.5
RL + Ov	**84.4**	1481.1	0.0	0.0
RL + UP-UN-Ov	78.7	1706.3	1.9	0.1
RL + UN-UP-Ov	79.5	848.6	310.9	36.6
RL + BestStrategy	82.0	1062.7	205.2	19.3
RL + BestPosRule	83.0	1197.9	0.0	0.0
RL + BestRule	82.8	1058.7	399.1	37.7
IG				
RL + UP	75.5	1238.7	0.0	0.0
RL + UN	79.4	625.2	295.6	47.3
RL + Ov	**82.9**	1089.6	0.0	0.0
RL + UP-UN-Ov	75.5	1238.5	1.6	0.1
RL + UN-UP-Ov	79.4	625.2	295.6	47.3
RL + BestStrategy	81.0	698.6	179.8	25.7
RL + BestPosRule	81.1	911.6	0.0	0.0
RL + BestRule	81.4	713.7	360.8	50.6
TFPC-Keywords	79.2	1546.1	0.0	0.0

Table 6.7: Average accuracy and the average number of rules for the classification of the 20NG-B dataset using χ^2 and IG for feature selection and the keyword representation in comparison with the TFPC-Keywords algorithm

Overall, RL + Ov, which generated rules without negation was the best RL strategy in terms of average accuracy in the context of multi-class classification of the 20 Newsgroups dataset. Another observation was that the use of χ^2 as a feature selection technique enabled the RL strategies to produce better classification accuracy compared to the use of IG.

6.2 Experiments Using the Reuters-8 Dataset

This section discusses the evaluation of the eight strategies in the proposed IRL mechanism using the Reuters-8 dataset. In the experiments using the Reuters-8 dataset, keywords were extracted as described in Section 4.2.2, except in the case of the TFPC-Keywords algorithm where (as noted previously) the built-in keyword selection strategy was used. The number of features used for the representation of each class (10% of the total number of features) in the Reuters-8 dataset is shown in Table 6.8. The evaluation of the results generated using both binary and multi-class classification with respect to the Reuters-8 dataset are discussed in Sub-sections 6.2.1 and 6.2.2 respectively.

Class name	Number of features
acq	1,283
crude	630
earn	1,040
grain	263
interest	340
money-fx	526
ship	365
trade	597
total	5,044

Table 6.8: The number of features used for the representation of each class in the Reuters-8 dataset

6.2.1 Binary Classification

The results from the binary classification of the Reuters-8 dataset is reported and discussed in this sub-section. Again, the strategies in the proposed IRL mechanism is denoted as RL with the identifier for the different rule refinement strategies used appended. Table 6.9 shows the micro-averaged F_1-measure and the average number of rules generated by the eight strategies in the proposed IRL mechanism using χ^2 and IG for the Reuters-8 dataset. The micro-averaged F_1-measure for the best RL strategy in comparison with the other machine learning techniques for classification of the Reuters-8 dataset is shown in Table 6.10.

The results presented in Table 6.9 demonstrated that both RL + BestPosRule (which did not generate rules with negation) and RL + BestRule (which did generate rules with negation) had the highest F_1-measure when χ^2 was used. However, RL + BestRule achieved this result with a smaller ruleset, an average of 33.0 less rules. The worst result was that of RL + UP with the lowest F_1-measure and the highest average number of rules. When IG was used, RL + BestStrategy had the best result, followed closely by RL + BestRule. Both strategies generated rules with negation. RL + UP and RL + UP-UN-Ov performed worst, with the lowest recorded F_1-measure and the highest average number of rules. Comparing the first three strategies, which utilized only one sub-space, the best of the three was the RL + Ov strategy, followed by the RL + UN strategy, which generated a far lower average number of rules, regardless of the feature selection technique used.

From Table 6.10, in comparison with the other machine learning techniques, the best RL strategy only managed to come out in fourth place irrespective of the feature selection technique used. Comparison of only the rule-based techniques (not including SMO and NB) showed that the RL strategy was in third place. Despite this, the recorded performance of the proposed IRL mechanism was only very slightly worse than that of OlexGreedy and JRip, and slightly better than OlexGA and NB. SMO, as usual, recorded the best results. The rather similar results recorded for the best RL

strategy, OlexGreedy and OlexGA suggested that these rule-based techniques were, in one way or another, responding similarly when classifying the Reuters-8 dataset.

Strategies	F_1	Average # of rules	Average # of rules with negation	% of rules with negation
		χ^2		
RL + UP	0.821	616.3	0.0	0.0
RL + UN	0.842	175.1	104.8	59.9
RL + Ov	0.860	304.2	0.0	0.0
RL + UP-UN-Ov	0.822	616.3	0.1	0.0
RL + UN-UP-Ov	0.848	232.8	132.8	57.0
RL + BestStrategy	0.877	285.7	73.5	25.7
RL + BestPosRule	**0.882**	310.8	0.0	0.0
RL + BestRule	**0.882**	277.8	155.7	56.0
		IG		
RL + UP	0.791	507.4	0.0	0.0
RL + UN	0.840	125.7	82.1	65.3
RL + Ov	0.858	227.5	0.0	0.0
RL + UP-UN-Ov	0.791	507.4	0.0	0.0
RL + UN-UP-Ov	0.841	140.9	92.5	65.6
RL + BestStrategy	**0.881**	181.6	68.6	37.8
RL + BestPosRule	0.878	233.8	0.0	0.0
RL + BestRule	0.880	183.6	116.7	63.6

Table 6.9: Micro-averaged F_1-measure and the average number of rules for the classification of the Reuters-8 dataset using χ^2 and IG for feature selection and the keyword representation

Techniques	χ^2	Rank	IG	Rank
RL	0.882	4	0.881	4
SMO	0.932	1	0.932	1
NB	0.775	6	0.748	6
JRip	0.896	2	0.900	2
OlexGreedy	0.883	3	0.892	3
OlexGA	0.872	5	0.880	5

Table 6.10: Micro-averaged F_1-measure for the classification of the Reuters-8 dataset using χ^2 and IG for feature selection and the keyword representation for the best RL strategy in comparison with the other machine learning techniques

Overall, with respect to the binary classification of the Reuters-8 dataset, when IG was used, the average number of rules generated was smaller and the percentage of rules with negation was higher compared to when χ^2 was used. However, the use of χ^2 enabled the RL strategies to produce higher F_1-measure values than the use of

IG, except in the case of the RL + BestStrategy, where it was vice versa. The best RL strategy only performed averagely when compared to the other machine learning techniques, ranking fourth out of six, regardless of the feature selection technique used.

6.2.2 Multi-Class Classification

This sub-section discusses the results of the evaluation conducted in the context of multi-class classification where the operation of the eight strategies in the proposed IRL mechanism was compared to that of the TFPC-Keywords algorithm for the Reuters-8 dataset. Table 6.11 shows the average accuracy results obtained and the average number of rules generated using the eight strategies in the proposed IRL mechanism in comparison with the TFPC-Keywords algorithm, using χ^2 and IG when classifying the Reuters-8 dataset.

Strategies	Average Acc (%)	Average # of rules	Average # of rules with negation	% of rules with negation
χ^2				
RL + UP	79.6	616.3	0.0	0.0
RL + UN	84.0	175.1	104.8	59.9
RL + Ov	86.7	304.2	0.0	0.0
RL + UP-UN-Ov	79.6	616.3	0.1	0.0
RL + UN-UP-Ov	83.7	232.8	132.8	57.0
RL + BestStrategy	86.1	285.7	73.5	25.7
RL + BestPosRule	**87.3**	310.8	0.0	0.0
RL + BestRule	87.1	277.8	155.7	56.0
IG				
RL + UP	75.6	507.4	0.0	0.0
RL + UN	83.4	125.7	82.1	65.3
RL + Ov	85.9	227.5	0.0	0.0
RL + UP-UN-Ov	75.6	507.4	0.0	0.0
RL + UN-UP-Ov	83.1	140.9	92.5	65.6
RL + BestStrategy	85.9	181.6	68.6	37.8
RL + BestPosRule	**86.3**	233.8	0.0	0.0
RL + BestRule	86.2	183.6	116.7	63.6
TFPC-Keywords	75.2	3235.6	0.0	0.0

Table 6.11: Average accuracy and the average number of rules for the classification of the Reuters-8 dataset using χ^2 and IG for feature selection and the keyword representation in comparison with the TFPC-Keywords algorithm

The results shown in Table 6.11 indicated that RL + BestPosRule outperformed all the other strategies, with RL + BestRule very close behind. However, RL + BestPosRule achieved the result with an average number of rules which was 11.9% (when χ^2 was used) and 27.3% (when IG was used) more than obtained using RL + BestRule.

RL + UP and RL + UP-UN-Ov were again the worst among the RL strategies. The difference between the best and worst RL strategies was 7.7% (when χ^2 was used) and 10.7% (when IG was used), translating to an additional average of 511.5 and 710.8 (out of 6,643) documents being correctly classified by RL + BestPosRule. TFPC-Keywords had the lowest average accuracy of all. A very high average number of rules for TFPC-Keywords only served to aggravate the fact that it has the lowest average accuracy. The best RL strategy was 12.1% more accurate than TFPC-Keywords, which translated to an additional average of 803.8 (out of 6,643) documents correctly classified by the best RL strategy.

Overall, RL + BestPosRule, which generated rules without negation, was better in terms of average accuracy. Again, the use of χ^2 produced higher classification accuracy compared to the use of IG. When compared to the TFPC-Keywords, the RL strategies produced much better results.

6.3 Experiments Using the SAVSNET Dataset

This section discusses the evaluation of the eight strategies in the proposed IRL mechanism using the SAVSNET dataset. For the experiments using the SAVSNET dataset, keywords were again extracted as described in Section 4.2.2 (except when using TFPC-Keywords). The number of features used for the representation of each class (10% of the total number of features) in the SAVSNET dataset is shown in Table 6.12. The evaluation using binary and multi-class classification applied to the SAVSNET dataset are discussed in Sub-sections 6.3.1 and 6.3.2 respectively.

Class name	Number of features
aggression	69
diarrhoea	248
pruritus	269
vomit	192
total	778

Table 6.12: The number of features used for the representation of each class in the SAVSNET dataset

6.3.1 Binary Classification

This sub-section discusses the results for the binary classification of the SAVSNET dataset. Table 6.13 shows the micro-averaged F_1-measure and the average number of rules associated with the eight strategies in the IRL mechanism using χ^2 and IG for feature selection. Table 6.14 shows the micro-averaged F_1-measure for the best RL

strategy in comparison with the other machine learning techniques considered for the classification of the SAVSNET dataset.

Inspection of Table 6.13 indicates that the best results were obtained using RL + UN. This strategy not only generated rules with negation but also generate the lowest average number of rules (smallest ruleset). The worst of the strategies appeared to be RL + UP with the lowest F_1-measure value and the highest average number of rules.

Strategies	F_1	Average # of rules	Average # of rules with negation	% of rules with negation
χ^2				
RL + UP	0.794	182.0	0.0	0.0
RL + UN	**0.832**	90.7	47.0	51.8
RL + Ov	0.823	123.0	0.0	0.0
RL + UP-UN-Ov	0.794	182.0	0.2	0.1
RL + UN-UP-Ov	0.826	98.1	48.4	49.3
RL + BestStrategy	0.822	105.9	34.1	32.2
RL + BestPosRule	0.829	134.4	0.0	0.0
RL + BestRule	0.827	105.5	51.7	49.0
IG				
RL + UP	0.769	158.7	0.0	0.0
RL + UN	**0.823**	85.1	50.0	58.8
RL + Ov	0.818	108.7	0.0	0.0
RL + UP-UN-Ov	0.769	158.7	0.2	0.1
RL + UN-UP-Ov	**0.823**	86.6	50.7	58.5
RL + BestStrategy	0.814	93.1	36.9	39.6
RL + BestPosRule	0.816	115.3	0.0	0.0
RL + BestRule	0.816	91.6	56.8	62.0

Table 6.13: Micro-averaged F_1-measure and the average number of rules for the classification of the SAVSNET dataset using χ^2 and IG for feature selection and the keyword representation

Techniques	χ^2	Rank	IG	Rank
RL	0.832	2	0.823	1
SMO	0.852	1	0.813	4
NB	0.829	3	0.804	5
JRip	0.796	5	0.779	6
OlexGreedy	0.765	6	0.820	3
OlexGA	0.820	4	0.823	1

Table 6.14: Micro-averaged F_1-measure for the classification of the SAVSNET dataset using χ^2 and IG for feature selection and the keyword representation for the best RL strategy in comparison with the other machine learning techniques

In comparison with the other machine learning techniques as shown in Table 6.14, the best RL strategy was reasonably effective, coming in second behind SMO, with OlexGreedy producing the worst result when χ^2 was used. The best RL strategy produced the equal best performance, together with OlexGA, while JRip produced the worst result, when IG was used. Comparison of only the rule-based techniques (not including SMO and NB) showed that the best RL strategy was the most effective, regardless of the feature selection technique used.

Overall, the RL strategy that generated rules with negation produced the best performance. The use of IG again enabled the RL strategies to generate a much smaller ruleset and a higher percentage of rules with negation compared to the use of χ^2. However, using χ^2 for feature selection enabled the RL strategies to produce higher F_1-measure values compared to using IG. Compared to the other machine learning techniques, the best RL strategy was among the best and competitive with SMO.

6.3.2 Multi-Class Classification

This sub-section discusses the evaluation results obtained in the context of multi-class classification using the eight strategies in the proposed IRL mechanism in comparison with the TFPC-Keywords algorithm when applied to the SAVSNET dataset. Table 6.15 shows the average accuracy obtained and the number of rules generated using χ^2 and IG when classifying the SAVSNET dataset using the eight strategies for the proposed IRL mechanism in comparison with the TFPC-Keywords algorithm.

The best recorded result with respect to the multi-class classification evaluation when χ^2 was used, as shown in Table 6.15, was obtained using RL + UN, which generated rules with negation. The worst result was obtained using RL + UP and RL + UP-UN-Ov. The difference of 8.4% in average accuracy between the best and worst RL strategies shows that the best RL strategy was able to generate rules that correctly classify an additional average of 69.6 (out of 828) documents. RL + Ov produced the best result among those strategies which did not generate rules with negation. It achieved an accuracy of 82.4% which was closely comparable to RL + UN. However, this result was achieved using a much larger ruleset, 35.6% more rules than RL + UN.

When IG was used, the results as shown in Table 6.15 indicated that the performance of RL + UN-UP-Ov was slightly better than RL + UN. Again, both strategies generated rules with negation and had the smallest rulesets. The worst results were again obtained using RL + UP and RL + UP-UN-Ov. The margin between the best and worst was 10.0%, indicating that RL + UN generated rules that could correctly classified an additional average of 82.8 (out of 828) documents when compared to RL + UP. RL + Ov was again trailing closely behind, but this result was also achieved using a ruleset that was 28.0% larger than that generated using RL + UN.

Strategies	Average Acc (%)	Average # of rules	Average # of rules with negation	% of rules with negation
χ^2				
RL + UP	74.3	182.0	0.0	0.0
RL + UN	**82.7**	90.7	47.0	51.8
RL + Ov	82.4	123.0	0.0	0.0
RL + UP-UN-Ov	74.3	182.0	0.2	0.1
RL + UN-UP-Ov	82.3	98.1	48.4	49.3
RL + BestStrategy	79.4	105.9	34.1	32.2
RL + BestPosRule	79.7	134.4	0.0	0.0
RL + BestRule	79.6	105.5	51.7	49.0
IG				
RL + UP	72.1	158.7	0.0	0.0
RL + UN	81.9	85.1	50.0	58.8
RL + Ov	81.3	108.9	0.0	0.0
RL + UP-UN-Ov	72.1	158.7	0.2	0.1
RL + UN-UP-Ov	**82.1**	86.6	50.7	58.5
RL + BestStrategy	78.9	93.1	36.9	39.6
RL + BestPosRule	78.1	115.3	0.0	0.0
RL + BestRule	78.1	91.6	56.8	62.0
TFPC-Keywords	36.1	2318.2	0.0	0.0

Table 6.15: Average accuracy and the average number of rules for the classification of the SAVSNET dataset using χ^2 and IG for feature selection and the keyword representation in comparison with the TFPC-Keywords algorithm

TFPC-Keywords performed very badly with respect to the classification of the SAVSNET dataset, achieving only 36.1% average accuracy with a very large ruleset comprising an average of 2,318.2 rules. The best RL strategy was 46.6% more accurate than TFPC-Keywords, which translated to an additional average of 385.8 (out of 828) documents correctly classified by the best RL strategy. This poor performance could be attributed to the fact that the documents in the SAVSNET dataset, consisting of free text questionnaire data, tended to be very small (compared to the 20 Newsgroups and Reuters datasets), and often featured misspellings, poor grammar and punctuation, which the keyword selection strategy embedded in the TFPC-Keywords algorithm could not easily cope with. However, it is also possible that alternative support and confidence threshold values to the default settings might produce slightly better results, although the author of the TFPC algorithm suggested that the default values provided worked best [89].

Overall, the RL strategies which generated rules with negation had the best results compared to the RL strategies which generated rules without negation in terms of the average accuracy. It was again observed that the use of χ^2 enabled the RL strategies to produce higher classification accuracies than when IG was used. All the RL strategies outperformed the TFPC-Keywords substantially.

6.4 Experiments Using the UCI Machine Learning Repository Datasets

The five datasets taken from the UCI Machine Learning Repository comprised tabular data and not text data. In the case of these five datasets, the number of attributes (features) was small compared to the number of features in the text datasets. The number of attributes after discretization is shown in Table 6.16. Feature selection was not performed on these datasets but the attributes were treated as if they were keywords and sorted according to their χ^2 and IG values. Evaluation using the UCI datasets was conducted because it was deemed desirable to determine how the proposed IRL mechanism might operated with respect to more general non-text classification problems. Of course, if a dataset features very few attributes, as in the case of the UCI datasets, it is entirely feasible to include the negation of each attribute as part of the input data. However, it was still considered worthwhile to see how well the proposed IRL mechanism operated with respect to these UCI tabular datasets. Sub-sections 6.4.1 and 6.4.2 discuss the results for the binary and multi-class classification of the UCI datasets respectively.

Dataset name	Number of discretized attributes
Iris	19
Adult	97
Wine	68
Breast	20
Car	25

Table 6.16: The number of discretized attributes for the five UCI Machine Learning Repository datasets

6.4.1 Binary Classification

This sub-section discusses the results obtained using binary classification applied to the selected five UCI datasets. The datasets used were: "Iris", "Adult", "Wine", "Breast" and "Car". Table 6.17 shows the micro-averaged F_1-measure obtained using the eight strategies in the proposed IRL mechanism in comparison with the other machine learning techniques for the classification of the UCI datasets using χ^2 and IG for feature selection.

The first observation that can be made with respect to the results presented in Table 6.17 is that both sets of results (χ^2 and IG) featured a similar trend and an almost identical outcome. This leads to the conclusion that it is not relevant whether χ^2 or IG is used. This was due to the fact that no feature selection was performed and therefore, the full set of attributes were used as features for the representation.

With all attributes present during rule learning, there could only be a subtle difference between rules that came from the different ordering of attributes, sorted by χ^2 and IG respectively. In the case of the "Iris" dataset, the best result was produced using RL + UN, followed closely by RL + BestRule and RL + BestStrategy. All three were strategies which generated rules with negation. The best of these was also better than all the other machine learning techniques. RL + UP and RL + UP-UN-Ov produced the worst results among all the strategies. For the remaining four datasets, RL + Ov, producing only rules without negation, was the best strategy. In comparison with the other machine learning techniques, RL + Ov was best at classifying the "Wine" dataset but was worst at classifying the "Breast" dataset. It came in second worst for both the "Adult" and "Car" datasets.

Strategies/UCI datasets	Iris	Adult	Wine	Breast	Car
χ^2					
RL + UP	0.864	0.767	0.850	0.839	0.750
RL + UN	**0.954**	0.802	0.870	0.918	0.752
RL + Ov	0.942	0.822	**0.925**	0.920	0.755
RL + UP-U-Ov	0.864	0.769	0.850	0.853	0.750
RL + UN-UP-Ov	0.920	0.789	0.871	0.853	0.750
RL + BestStrategy	0.952	0.673	0.838	0.857	0.750
RL + BestPosRule	0.949	0.785	0.906	0.870	0.750
RL + BestRule	0.953	0.772	0.861	0.878	0.750
SMO	0.942	**0.848**	0.917	0.927	0.846
JRip	0.939	0.845	0.899	0.928	**0.905**
NB	0.950	0.804	0.917	**0.941**	0.791
OlexGreedy	0.943	0.833	0.901	0.930	0.765
OlexGA	0.941	0.837	0.873	0.928	0.661
IG					
RL + UP	0.868	0.731	0.850	0.839	0.750
RL + UN	**0.958**	0.804	0.870	0.918	0.752
RL + Ov	0.944	0.815	**0.925**	0.920	0.755
RL + UP-U-Ov	0.868	0.732	0.850	0.853	0.750
RL + UN-UP-Ov	0.920	0.789	0.871	0.853	0.750
RL + BestStrategy	0.957	0.737	0.838	0.857	0.750
RL + BestPosRule	0.955	0.785	0.906	0.870	0.750
RL + BestRule	0.955	0.769	0.861	0.878	0.750
SMO	0.942	**0.848**	0.917	0.927	0.847
JRip	0.939	0.844	0.912	0.928	**0.906**
NB	0.950	0.804	0.917	**0.941**	0.791
OlexGreedy	0.943	0.833	0.901	0.930	0.765
OlexGA	0.944	0.839	0.829	0.929	0.666

Table 6.17: Micro-averaged F_1-measure for the classification of the UCI datasets using χ^2 and IG for feature selection and the keyword representation

Overall, it was observed that the results were inconclusive as to whether the use of negation in IRL made a difference in non-text datasets. However, it was noted that when χ^2 was used, the F_1-measure values for the "Iris" dataset were slightly lower than when IG was used. There was no definite trend for the F_1-measure values for the "Adult" dataset while the F_1-measure values were equal for the "Wine", "Breast" and "Car" datasets regardless of the feature selection technique used. The use of χ^2 and IG also did not affect the results for the other machine learning techniques much, as they recorded almost identical results.

6.4.2 Multi-Class Classification

This sub-section discusses the results of the evaluation conducted using multi-class classification and the eight strategies in the proposed IRL mechanism with respect to the UCI datasets. TFPC-Keywords could not be used for the evaluation as this system was designed for text classification only. Table 6.18 shows the average accuracy obtained when classifying the UCI datasets using χ^2 and IG for feature selection.

Strategies	Avg accuracy (%)				
	Iris	Adult	Wine	Breast	Car
χ^2					
RL + UP	**96.0**	77.1	86.0	91.6	**70.0**
RL + UN	95.3	76.7	84.3	91.4	**70.0**
RL + Ov	**96.0**	**80.4**	**93.3**	90.8	**70.0**
RL + UP-U-Ov	**96.0**	76.3	86.0	91.4	**70.0**
RL + UN-UP-Ov	**96.0**	76.4	84.3	91.4	**70.0**
RL + BestStrategy	**96.0**	76.4	86.0	90.3	**70.0**
RL + BestPosRule	**96.0**	77.3	89.9	90.8	**70.0**
RL + BestRule	**96.0**	77.3	87.1	**92.9**	**70.0**
IG					
RL + UP	**96.0**	77.1	86.0	91.6	**70.0**
RL + UN	**96.0**	77.0	84.3	91.4	**70.0**
RL + Ov	95.3	**79.8**	**93.3**	90.8	**70.0**
RL + UP-U-Ov	**96.0**	76.3	86.0	91.4	**70.0**
RL + UN-UP-Ov	**96.0**	76.4	84.3	91.4	**70.0**
RL + BestStrategy	**96.0**	76.2	86.0	90.3	**70.0**
RL + BestPosRule	**96.0**	77.3	89.9	90.8	**70.0**
RL + BestRule	**96.0**	77.4	87.1	**92.9**	**70.0**
TFPC-Keywords	-	-	-	-	-

Table 6.18: Average accuracy for the classification of the UCI datasets using χ^2 and IG for feature selection and the keyword representation

When the RL strategies were used to perform multi-class classification on the five UCI datasets, an almost identical set of results was obtained. The results for both the

"Iris" and "Car" datasets showed that all the strategies produced the same results, apart from a slight difference for RL + UN in the case of the "Iris" dataset. For the "Adult" and "Wine" datasets, RL + Ov produced the best results. For the "Breast" dataset, RL + BestRule was the best.

Overall, it was noted that the choice of feature selection technique did not have a significant effect with respect to the classification of the UCI datasets. As already mentioned in the previous sub-section, no subset of attributes was selected and the feature selection techniques were just used for ordering the attributes.

6.5 Summary

This chapter has reported on the evaluation of the proposed strategies in the IRL mechanism using both binary and multi-class classification with respect to the 20 Newsgroups, Reuters-8, SAVSNET and UCI datasets. The eight strategies in the proposed IRL mechanism were compared with one another to see whether the strategies with negation were better than the strategies without negation. The best RL strategy in each case was then used for comparison with the other machine learning techniques, namely SMO, JRip, NB, OlexGreedy and OlexGA in the binary classification setting. The TFPC-Keywords algorithm was used to compare with the RL strategies in the multi-class classification setting. Two feature selection techniques, χ^2 and IG, were used and it was observed that the use of χ^2 enabled the RL strategies to produce better classification results than when IG was used. However, the use of IG enabled the RL strategies to learn a smaller ruleset. The strategies which generated rules with negation (RL + UN, RL + UN-UP-OV, RL + BestStrategy, RL + BestRule) also, in general, had a smaller average number of rules compared to strategies which did not generate rules with negation (RL + UP, RL + Ov, RL + UP-UN-Ov, RL + BestPosRule). This is deemed desirable, particularly if the RL strategies could also produce better results in addition to having a smaller average number of rules. In some cases, RL + UP-UN-Ov generated a very small percentage of rules with negation ($< 1\%$). However, this RL strategy was not considered to be a strategy which was expected to significantly generate rules with negation.

Tables 6.19 and 6.20 present the results summary of the experiments conducted as described in this chapter. The tables indicate whether the RL strategies that generated rules with or without negation were best with respect to each dataset in the context of binary and multi-class classification respectively (a tick in a column indicates the best performance).

Datasets	χ^2		IG	
	Without Neg	With Neg	Without Neg	With Neg
20NG-A		√		√
20NG-B		√		√
Reuters-8	√	√		√
SAVSNET		√		√
UCI - Iris		√		√
UCI - Adult	√		√	
UCI - Wine	√		√	
UCI - Breast	√		√	
UCI - Car	√		√	

Table 6.19: Summary of the best RL strategy for binary classification with respect to all the datasets

Datasets	χ^2		IG	
	Without Neg	With Neg	Without Neg	With Neg
20NG-A	√		√	
20NG-B	√		√	
Reuters-8	√		√	
SAVSNET		√		√
UCI - Iris	√	√	√	√
UCI - Adult	√		√	
UCI - Wine	√		√	
UCI - Breast		√		√
UCI - Car	√	√	√	√

Table 6.20: Summary of the best RL strategy for multi-class classification with respect to all the datasets

With respect to binary classification, the RL strategies which generated rules with negation were found to produce better results, especially in the case of the text datasets. The RL strategies which generated rules without negation were better at classifying the UCI datasets (the non-text datasets). In the multi-class classification setting the use of negation was found to produce a better performance only with respect to the SAVSNET and "Breast" datasets. Equal performances across all the RL strategies were seen with respect to the "Iris" and "Car" datasets.

Overall, it was observed that the RL strategies which generated rule with negation were more prevalent in the context of the text datasets and binary classification, than with respect to the tabular datasets and multi-class classification when keywords were used as the text representation. In the following chapter, which is Chapter 7, the evaluation of the IRL mechanism with respect to the use of phrases as the text representation is considered.

Chapter 7

Experimental Setup, Results and Evaluation Using Phrases for Inductive Rule Learning with Negation

This chapter describes the experimental setup, results and evaluation using phrases in the text representation for the proposed Inductive Rule Learning (IRL) mechanism. Three different types of phrases were used in the evaluation: (i) n-gram phrases; (ii) keyphrases and (iii) fuzzy phrases. The evaluation of the text classification task was performed on the 20 Newsgroups, Reuters-8 and SAVSNET datasets as described in Chapter 5. The UCI Machine Learning Repository datasets were not used as it did not make sense to treat attributes in tabular datasets as phrases in the context of this study.

In the experiments conducted, Ten-fold Cross Validation (TCV) was adopted. The two feature selection techniques considered previously, Chi-Square (χ^2) and Information Gain (IG), were again used for selecting a subset of the phrases to be used as features. For each of the datasets used, both binary and multi-class classification were performed. For the binary classification, the eight strategies in the proposed IRL mechanism were first compared with one another and the best RL strategy was then compared with the Sequential Minimal Optimization (SMO) algorithm, Naive Bayes (NB), JRip, OlexGreedy and OlexGA. The evaluation was conducted using the micro-averaged F_1-measure as an averaging measure across all the classes in a dataset. In multi-class classification, the eight strategies in the proposed IRL mechanism were again compared with one another in addition to comparison with the Total From Partial Classification (TFPC) algorithm (with default thresholds: support = 0.1% and confidence = 35.0%), and its associated four phrase selection strategies. The measure used for evaluation in the case of multi-class classification was the accuracy (Acc) measure. The results for the multi-class classification are reported and discussed in Appendix B, C and D for

the n-gram phrase, keyphrase and fuzzy phrase representations respectively while the results from the binary classification are presented in this chapter. Both sets of results were not presented in the body of this chapter because the results corroborated one another and thus, only one set of results was presented to facilitate ease of reading.

In this chapter, Sections 7.1, 7.2 and 7.3 discuss the evaluation using the n-gram phrase, keyphrase and fuzzy phrase representations respectively with respect to the binary classification of the 20 Newsgroups, Reuters-8 and SAVSNET datasets.

7.1 Evaluation Using the N-gram Phrase Representation

This section describes the evaluation using n-gram phrases for the text representation in the proposed Inductive Rule Learning (IRL) mechanism. Three different n-gram representations were used in the evaluation:

1. 2-gram
2. 3-gram
3. Mixed (Keyword, 2-gram, 3-gram)

A pre-determined percentage of 10% (90% reduction) of the total number of features in each class for both the 2-gram and 3-gram representations was adopted. When the mixed representation was used, keywords and n-gram phrases of different lengths, in this case, 2-grams and 3-grams, were all pooled together and used to represent the documents. Therefore, due to the high number of features (especially for the 20 Newsgroups dataset) and computational limitations, only 1% of the total number of features available were used for each class in a dataset.

In general, longer phrases tended to be more unique and occurred less frequently across all documents in a dataset. Therefore, due to their uniqueness and frequency, the use of 3-grams for text representation was expected to be less effective in text classification than the use of 2-grams. The classification results obtained from the experiments conducted had shown this to be so. Thus, the discussion of the results will be focused on the use of 2-grams and the mixed representation. Should the reader be interested in the results produced using the 3-gram representation, these are shown in Appendix B.

7.1.1 Experiments Using the 20 Newsgroups Dataset

This sub-section discusses the evaluation of the eight strategies in the proposed IRL mechanism using the 20 Newsgroups dataset using n-gram phrases in the text representation. The number of n-gram phrase (2-grams, 3-grams, mixed) features used for the representation of each class in the 20 Newsgroups dataset is shown in Table 7.1.

Class name	2-gram	3-gram	Mixed
20NG-A			
alt.atheism	7,522	8628	1,759
comp.sys.ibm.pc.hardware	4,929	5,892	1,189
comp.windows.x	8,377	9,934	2,002
misc.forsale	4,256	4,908	1,031
rec.motorcycles	4,666	5,197	1,106
rec.sport.baseball	5439	6,369	1,289
sci.electronics	5,231	5,824	1,226
sci.med	8,408	9,524	1,977
talk.politics.mideast	11,433	13,270	2,662
talk.religion.misc	7,743	8,836	1,817
total	68,004	78,382	16,058
20NG-B			
comp.graphics	6,638	7,723	1575
comp.os.ms-windows.misc	10,945	13,001	2,665
comp.sys.mac.hardware	4,685	5,421	1,116
rec.autos	5,318	5,986	1,253
rec.sport.hockey	6,738	8,295	1,628
sci.crypt	8,000	9,333	1,887
sci.space	8,287	9,487	1,940
soc.religion.christian	9,868	11,344	2,285
talk.politics.guns	8,282	9,536	1,949
talk.politics.misc	10,312	11,937	2,401
total	79,073	92,063	18,699

Table 7.1: The number of n-gram features used for the representation of each class in the 20 Newsgroups dataset

As in the case of the previous chapter, the eight strategies in the proposed IRL mechanism are again denoted as RL with the identifier for the different rule refinement strategies used appended. The micro-averaged F_1-measure and the average number of rules generated by the IRL mechanism using χ^2 and IG for the 20NG-A dataset is shown in Table 7.2 for the 2-gram representation and Table 7.3 for the mixed representation. Tables 7.5 and 7.6 show the same results but for the 20NG-B dataset. The micro-averaged F_1-measure for the other machine learning techniques in comparison with the best RL strategy for the classification of the 20NG-A and 20NG-B datasets are shown in Tables 7.4 and 7.7 respectively.

Recall that when the 2-gram representation was used only phrases of length two were used to represent the documents. In the case of the 20NG-A dataset, the binary classification results presented in Table 7.2 shows that RL + BestPosRule had the best F_1-measure among all the strategies regardless of whether χ^2 or IG was used. This strategy did not generate any rules with negation. The worst strategy was RL + UP-UN-Ov, which also generated the highest average number of rules. It was observed that

the use of χ^2 produced higher F_1-measure values for the strategies as compared to the use of IG. When IG was used, a slightly higher percentage of rules with negation and average number of rules was produced, compared to the use of χ^2. However, all the strategies generated a very high average number of rules regardless of whether χ^2 or IG was used; on average more than 2000 rules. In addition to the fact that a high number of n-gram phrases was extracted, a phrase was by itself more unique (than a keyword) and therefore could be expected to appear in fewer documents. Therefore, a rule made up of a phrase or conjunction of phrases would cover a smaller number of documents, resulting in the generation of more rules in order to "cover" all the documents in a dataset.

Strategies	F_1	Average # of rules	Average # of rules with negation	% of rules with negation
		χ^2		
RL + UP	0.828	2918.3	0.0	0.0
RL + UN	0.832	2267.1	257.5	11.4
RL + Ov	0.836	2517.4	0.0	0.0
RL + UP-UN-Ov	0.826	2922.8	20.5	0.7
RL + UN-UP-Ov	0.832	2381.5	267.7	11.2
RL + BestStrategy	0.833	2276.5	173.9	7.6
RL + BestPosRule	**0.837**	2293.9	0.0	0.0
RL + BestRule	0.831	2283.7	263.2	11.5
		IG		
RL + UP	0.795	3070.4	0.0	0.0
RL + UN	0.796	2300.2	425.3	18.5
RL + Ov	0.795	2634.3	0.0	0.0
RL + UP-UN-Ov	0.794	3078.7	19.4	0.6
RL + UN-UP-Ov	0.797	2398.6	437.6	18.2
RL + BestStrategy	0.805	2319.0	244.8	10.6
RL + BestPosRule	**0.810**	2344.1	0.0	0.0
RL + BestRule	0.801	2324.3	423.3	18.2

Table 7.2: Micro-averaged F_1-measure and average number of rules for the classification of the 20NG-A dataset using χ^2 and IG for feature selection and the 2-gram representation

When binary classification was performed on the 20NG-A dataset using the mixed representation, the best RL strategy, regardless of the feature selection technique used, was RL + BestStrategy, which generated rules with negation, as shown in Table 7.3. The second best strategy was RL + BestRule, also a strategy which generated rules with negation. The worst strategies were RL + UN-UP-Ov (when χ^2 was used) and RL + UP-UN-Ov (when IG was used). The use of IG enabled the RL strategies to produce a higher percentage of rules with negation compared to the use of χ^2 (except

for RL + UP-UN-Ov). However, the use of χ^2 enabled the RL strategies to obtain higher F_1-measure values than when IG was used, albeit with bigger rulesets.

Strategies	F_1	Average # of rules	Average # of rules with negation	% of rules with negation
χ^2				
RL + UP	0.818	1533.7	0.0	0.0
RL + UN	0.822	820.8	293.2	35.7
RL + Ov	0.819	1305.6	0.0	0.0
RL + UP-UN-Ov	0.817	1535.8	7.7	0.5
RL + UN-UP-Ov	0.816	867.8	310.3	35.8
RL + BestStrategy	**0.836**	904.5	209.7	23.2
RL + BestPosRule	0.832	1027.6	0.0	0.0
RL + BestRule	0.833	954.6	350.5	36.7
IG				
RL + UP	0.800	1400.7	0.0	0.0
RL + UN	0.808	741.3	342.3	46.2
RL + Ov	0.805	1172.8	0.0	0.0
RL + UP-UN-Ov	0.799	1403.0	4.8	0.3
RL + UN-UP-Ov	0.808	751.0	342.4	45.6
RL + BestStrategy	**0.830**	764.5	226.7	29.7
RL + BestPosRule	0.824	938.3	0.0	0.0
RL + BestRule	0.828	789.3	366.7	46.5

Table 7.3: Micro-averaged F_1-measure and average number of rules for the classification of the 20NG-A dataset using χ^2 and IG for feature selection and the mixed representation

Techniques	χ^2	Rank	IG	Rank
2-gram				
RL	0.837	1	0.810	1
SMO	0.814	2	0.807	2
NB	0.603	6	0.581	6
JRip	0.665	5	0.658	5
OlexGreedy	0.729	4	0.694	4
OlexGA	0.735	3	0.734	3
Mixed				
RL	0.836	2	0.830	2
SMO	0.847	1	0.854	1
NB	0.664	6	0.689	6
JRip	0.771	5	0.773	5
OlexGreedy	0.817	4	0.820	3
OlexGA	0.819	3	0.819	4

Table 7.4: Micro-averaged F_1-measure for the classification of the 20NG-A dataset using χ^2 and IG for feature selection for the best RL strategy in comparison with the other machine learning techniques using the 2-gram and mixed representations

Table 7.4 shows that when the best RL strategy was compared to the other machine learning techniques, it was the best in the context of the 2-gram representation of the documents. SMO was second best, while the worst technique was NB. When the mixed representation was used, SMO was found to be the best technique followed by the best RL strategy; NB was again the worst technique. When comparing only rule-based techniques (not including SMO and NB), the RL strategy was the best identified technique while the worst was JRip, regardless of which feature selection technique or representation was used.

Overall, with respect to the classification of the 20NG-A dataset, it was noted that RL + UP-UN-Ov generated the highest average number of rules while RL + UN generated the least average number of rules, regardless of the feature selection or text representation used. When the 2-gram representation was used, the best RL strategy was one which did not generate any rules with negation. However, when the mixed representation was used, the best RL strategy was one which generated rules with negation. In comparison with the other machine learning techniques, the best RL strategy performed well; it was generally better than all the other techniques and was competitive with SMO.

For the classification of the 20NG-B dataset using the 2-gram representation, RL + UP and RL + UP-UN-Ov were identified as equally best when χ^2 was used as shown in Table 7.5. While RL + UP did not generate any rules with negation, RL + UP-UN-Ov generated a very small percentage of rules with negation. The worst strategies were RL + UN and RL + UN-UP-Ov. When IG was used, the best strategy was RL + BestStrategy, which generated rules with negation. The worst strategy was RL + Ov. Again, it appeared that very large rulesets were generated when using the 2-gram representation, similar to the case in the 20NG-A dataset. It was also observed that the use of IG enabled the RL strategies to produce a higher percentage of rules with negation compared to the use of χ^2. However, when χ^2 was used, the F_1-measure values generated by the RL strategies were higher than when IG was used. These observations were also noted in the case of the 20NG-A dataset.

Table 7.6 shows that the use of the mixed representation with respect to the 20NG-B dataset resulted in RL + UP and RL + UP-UN-Ov being identified as the equal best strategies when χ^2 was used. RL + BestPosRule, which did not generate rules with negation, and RL + BestRule, which generated rules with negation, were both very closely behind with both generating fewer rules than RL + UP and RL + UP-UN-Ov. In fact, RL + BestRule generated 614.3 fewer rules than RL + UP-UN-Ov. The worst strategy was RL + UN-UP-Ov. When IG was used, RL + BestStrategy, which generated rules with negation, was found to be the best strategy. This was closely followed by RL + BestRule, which also generated rules with negation. The worst strategy was RL + UN. The percentage of rules with negation generated by the

RL strategies was also higher when IG was used compared to when χ^2 was used (except for RL + UP-UN-Ov). The average number of rules generated by the RL strategies was lower when IG was used compared to when χ^2 was used. However, the use of χ^2 enabled the RL strategies to produce higher F_1-measure values compared to when IG was used, with the exception of RL + UN-UP-Ov in this case.

Strategies	F_1	Average # of rules	Average # of rules with negation	% of rules with negation
χ^2				
RL + UP	**0.873**	2751.0	0.0	0.0
RL + UN	0.862	2129.2	258.0	12.1
RL + Ov	0.866	2407.8	0.0	0.0
RL + UP-UN-Ov	**0.873**	2753.9	9.4	0.3
RL + UN-UP-Ov	0.862	2223.3	265.6	11.9
RL + BestStrategy	0.869	2138.1	168.2	7.9
RL + BestPosRule	0.869	2159.9	0.0	0.0
RL + BestRule	0.869	2138.7	265.1	12.4
IG				
RL + UP	0.844	2826.2	0.0	0.0
RL + UN	0.833	2121.8	441.6	20.8
RL + Ov	0.830	2460.4	0.0	0.0
RL + UP-UN-Ov	0.844	2828.5	11.5	0.4
RL + UN-UP-Ov	0.833	2183.3	452.2	20.7
RL + BestStrategy	**0.849**	2131.9	230.3	10.8
RL + BestPosRule	0.846	2161.1	0.0	0.0
RL + BestRule	0.844	2130.6	443.6	20.8

Table 7.5: Micro-averaged F_1-measure and average number of rules for the classification of the 20NG-B dataset using χ^2 and IG for feature selection and the 2-gram representation

With reference to Table 7.7, the best RL strategy was compared to the other machine learning techniques for the binary classification of the 20NG-B dataset. It was noted that using the 2-gram representation, the best RL strategy was found to produce the best overall performance and therefore ranked first, while NB produced the worst performance. In the case of the mixed representation, SMO was found to produce the best overall performance and therefore ranked first, with the best RL strategy coming in second. NB again produced the overall worst performance. Comparing only the rule-based techniques (not including SMO and NB), the RL strategy was the best technique regardless of the feature selection technique or representation used.

Strategies	F_1	Average # of rules	Average # of rules with negation	% of rules with negation
χ^2				
RL + UP	**0.875**	1551.0	0.0	0.0
RL + UN	0.832	696.5	230.2	33.1
RL + Ov	0.845	1254.3	0.0	0.0
RL + UP-UN-Ov	**0.875**	1552.9	5.0	0.3
RL + UN-UP-Ov	0.826	779.5	256.1	32.9
RL + BestStrategy	0.871	949.7	214.0	22.5
RL + BestPosRule	0.874	1022.9	0.0	0.0
RL + BestRule	0.874	938.6	319.9	34.1
IG				
RL + UP	0.856	1436.6	0.0	0.0
RL + UN	0.830	662.0	294.0	44.4
RL + Ov	0.833	1202.3	0.0	0.0
RL + UP-UN-Ov	0.856	1438.0	2.4	0.2
RL + UN-UP-Ov	0.831	663.4	294.1	44.3
RL + BestStrategy	**0.870**	815.3	224.2	27.5
RL + BestPosRule	0.866	979.2	0.0	0.0
RL + BestRule	0.869	805.1	368.6	45.8

Table 7.6: Micro-averaged F_1-measure and average number of rules for the classification of the 20NG-B dataset using χ^2 and IG for feature selection and the mixed representation

Techniques	χ^2	Rank	IG	Rank
2-gram				
RL	0.873	1	0.849	1
SMO	0.858	2	0.849	1
NB	0.654	6	0.632	6
JRip	0.754	5	0.747	5
OlexGreedy	0.780	4	0.780	4
OlexGA	0.786	3	0.786	3
Mixed				
RL	0.875	2	0.870	2
SMO	0.891	1	0.897	1
NB	0.679	6	0.699	6
JRip	0.813	5	0.820	5
OlexGreedy	0.841	3	0.837	4
OlexGA	0.841	3	0.841	3

Table 7.7: Micro-averaged F_1-measure for the classification of the 20NG-B dataset using χ^2 and IG for feature selection for the best RL strategy in comparison with the other machine learning techniques using the 2-gram and mixed representations

Overall, with respect to the classification of the 20NG-B dataset, again, the RL strategy which generated the highest average number of rules was RL + UP-UN-Ov while RL + UN generated the least average number of rules, regardless of the feature selection technique or text representation used. It was observed that the best RL strategy was one which generated rules with negation when IG was used, regardless of the text representation used. However, when χ^2 was used, the best RL strategy was one which did not generate rules with negation, regardless of the text representation used. When compared to the other machine learning techniques, the best RL strategy again was better than all of the other techniques except for SMO, whereby their performance were closely competitive.

7.1.2 Experiments Using the Reuters-8 Dataset

This sub-section discusses the evaluation of the eight strategies in the proposed IRL mechanism with respect to the Reuters-8 dataset. Table 7.8 shows the number of n-gram phrase (2-grams, 3-grams and mixed) features used for the representation of each class in the Reuters-8 dataset.

Class name	2-gram	3-gram	Mixed
acq	9,236	12,724	2,324
crude	3,451	4,397	847
earn	5,607	8,132	1,478
grain	820	925	200
interest	1,473	1,830	364
money-fx	2,885	3,632	704
ship	1,164	1,310	284
trade	3,488	4,451	853
total	28,124	37,401	7,054

Table 7.8: The number of n-gram features used for the representation of each class in the Reuters-8 dataset

For the binary classification task, the micro-averaged F_1-measure and the average number of rules generated by the eight strategies in the IRL mechanism using χ^2 and IG for the Reuters-8 dataset are shown in Table 7.9 for the 2-gram representation, and in Table 7.10 for the mixed representation. The best RL strategy identified was then compared with the other machine learning techniques for the classification of the Reuters-8 dataset. The results obtained from this comparison are shown in Table 7.11.

When the 2-gram representation was used with χ^2 as the feature selection technique, RL + BestRule was the best RL strategy, as shown in Table 7.9. This strategy generated rules of which, on average, 41.1% were rules with negation. The worst RL strategies were found to be RL + Ov and RL + UN-UP-Ov. When IG was used as the feature

selection technique, RL + BestPosRule, which did not generate rules with negation, was found to be the best RL strategy. The next best RL strategy, RL + BestStrategy, generated rules with negation and achieved a slightly lower F_1-measure value, with on average 78.4 fewer rules compared to the best RL strategy. The identified worst RL strategies were RL + UP and RL + UP-UN-Ov. The use of IG enabled the RL strategies to produce slightly bigger rulesets than when χ^2 was used. However, the use of χ^2 enabled the RL strategies to obtain higher F_1-measure values compared to when using IG.

In the mixed representation, regardless of which feature selection technique was used, the best RL strategy was RL + BestRule, which generated rules with negation, as shown in Table 7.10. The worst RL strategy was RL + UN. The use of χ^2 enabled the RL strategies to obtain slightly higher F_1-measure values compared to the use of IG, although its use generated bigger rulesets in comparison. The percentage of rules with negation was higher for the RL strategies with the use of IG than the use of χ^2 (except for RL + UP-UN-Ov).

Strategies	F_1	Average # of rules	Average # of rules with negation	% of rules with negation
χ^2				
RL + UP	0.879	1532.8	0.0	0.0
RL + UN	0.873	742.5	297.5	40.1
RL + Ov	0.871	1193.3	0.0	0.0
RL + UP-UN-Ov	0.879	1532.8	2.6	0.2
RL + UN-UP-Ov	0.871	754.6	297.7	39.5
RL + BestStrategy	0.884	785.3	189.5	24.1
RL + BestPosRule	0.885	869.9	0.0	0.0
RL + BestRule	**0.887**	819.0	336.2	41.1
IG				
RL + UP	0.800	1843.0	0.0	0.0
RL + UN	0.844	796.2	321.4	40.4
RL + Ov	0.846	1372.3	0.0	0.0
RL + UP-UN-Ov	0.800	1843.0	3.5	0.2
RL + UN-UP-Ov	0.841	817.9	324.3	39.7
RL + BestStrategy	0.860	855.7	168.6	19.7
RL + BestPosRule	**0.861**	934.1	0.0	0.0
RL + BestRule	0.856	859.8	358.3	41.7

Table 7.9: Micro-averaged F_1-measure and average number of rules for the classification of the Reuters-8 dataset using χ^2 and IG for feature selection and the 2-gram representation

Strategies	F_1	Average # of rules	Average # of rules with negation	% of rules with negation
χ^2				
RL + UP	0.886	818.1	0.0	0.0
RL + UN	0.851	208.0	114.0	54.8
RL + Ov	0.872	371.1	0.0	0.0
RL + UP-UN-Ov	0.886	818.1	1.4	0.2
RL + UN-UP-Ov	0.862	266.6	144.4	54.2
RL + BestStrategy	0.898	335.0	91.5	27.3
RL + BestPosRule	0.897	416.5	0.0	0.0
RL + BestRule	**0.903**	345.0	191.7	55.6
IG				
RL + UP	0.860	654.3	0.0	0.0
RL + UN	0.847	162.0	101.6	62.7
RL + Ov	0.869	306.0	0.0	0.0
RL + UP-UN-Ov	0.860	654.3	0.9	0.1
RL + UN-UP-Ov	0.852	172.4	107.6	62.4
RL + BestStrategy	0.895	246.5	92.1	37.4
RL + BestPosRule	0.893	333.1	0.0	0.0
RL + BestRule	**0.901**	251.7	165.3	65.7

Table 7.10: Micro-averaged F_1-measure and average number of rules for the classification of the Reuters-8 dataset using χ^2 and IG for feature selection and the mixed representation

Techniques	χ^2	Rank	IG	Rank
2-gram				
RL	0.887	2	0.861	4
SMO	0.911	1	0.911	1
NB	0.802	6	0.802	6
JRip	0.844	5	0.832	5
OlexGreedy	0.875	4	0.875	3
OlexGA	0.879	3	0.879	2
Mixed				
RL	0.903	2	0.901	3
SMO	0.935	1	0.935	1
NB	0.832	6	0.787	6
JRip	0.899	3	0.913	2
OlexGreedy	0.886	5	0.892	5
OlexGA	0.895	4	0.896	4

Table 7.11: Micro-averaged F_1-measure for the classification of the Reuters-8 dataset using χ^2 and IG for feature selection for the best RL strategy in comparison with the other machine learning techniques using the 2-gram and mixed representations

In comparison with the other machine learning techniques, as shown in Table 7.11, the best RL strategy was second behind SMO when χ^2 was used regardless of the representation adopted. When IG was used, the best RL strategy ranked fourth using the 2-gram representation and third using the mixed representation. The worst technique was again found to be NB. Comparison of the best RL strategy with only the rule-based techniques (not including SMO and NB) showed that it performed best using χ^2 for both representations, while no difference in ranking was seen when IG was used.

Overall, the RL strategies that generated rules with negation were best at classifying the Reuters-8 dataset in all cases except for when IG was used with the 2-gram representation. The use of χ^2 again resulted in the RL strategies obtaining higher F_1-measure values compared to the use of IG, regardless of the representation used. When the best RL strategy was compared to the other machine learning techniques, it performed better than all the other techniques except SMO when χ^2 was used. However, its performance was just average when IG was used.

7.1.3 Experiments Using the SAVSNET Dataset

This sub-section describes the evaluation of the eight strategies in the proposed IRL mechanism using the SAVSNET dataset. Table 7.12 shows the number of n-gram phrase (2-grams, 3-grams, mixed) features used for representation of each class in the SAVSNET dataset.

Class name	2-gram	3-gram	Mixed
aggression	108	112	29
diarrhoea	829	1,017	209
pruritus	947	1,145	236
vomit	533	624	135
total	2,417	2,898	609

Table 7.12: The number of n-gram features used for the representation of each class in the SAVSNET dataset

For the binary classification task, the micro-averaged F_1-measure and the average number of rules generated by the strategies in the IRL mechanism using χ^2 and IG for the SAVSNET dataset are shown in Tables 7.13 and 7.14 for the 2-gram and mixed representations respectively. The best RL strategy in each case was then compared with the other machine learning techniques, with the results shown in Table 7.15.

When χ^2 was used with the 2-gram representation, the best strategy was RL + Ov, which did not generate rules with negation, as shown in Table 7.13. Although the results of the eight RL strategies were fairly close, the worst RL strategy in this case was RL + BestPosRule, which also did not generate rules with negation. When

IG was used instead, the best RL strategy was RL + BestRule, which generated rules with negation. The worst of the eight RL strategies were RL + UP and RL + UP-UN-Ov. It was observed that the percentage of rules with negation generated by the RL strategies was lower when χ^2 was used than when IG was used. However, similar to the 20 Newsgroups and Reuters-8 datasets, the use of χ^2 in this case enabled the RL strategies to produce higher F_1-measure values compared to the use of IG.

Strategies	F_1	Average # of rules	Average # of rules with negation	% of rules with negation
χ^2				
RL + UP	0.795	334.1	0.0	0.0
RL + UN	0.800	232.4	62.7	27.0
RL + Ov	**0.814**	279.7	0.0	0.0
RL + UP-UN-Ov	0.795	334.1	0.8	0.2
RL + UN-UP-Ov	0.799	237.6	62.9	26.5
RL + BestStrategy	0.800	243.0	34.2	14.1
RL + BestPosRule	0.794	258.0	0.0	0.0
RL + BestRule	0.803	243.1	64.1	26.4
IG				
RL + UP	0.738	344.6	0.0	0.0
RL + UN	0.783	214.7	68.4	31.9
RL + Ov	0.790	264.6	0.0	0.0
RL + UP-UN-Ov	0.738	344.6	0.8	0.2
RL + UN-UP-Ov	0.785	216.5	68.4	31.6
RL + BestStrategy	0.790	221.4	40.4	18.2
RL + BestPosRule	0.776	251.1	0.0	0.0
RL + BestRule	**0.795**	221.4	72.9	32.9

Table 7.13: Micro-averaged F_1-measure and average number of rules for the classification of the SAVSNET dataset using χ^2 and IG for feature selection and the 2-gram representation

In the mixed representation using χ^2, the two RL strategies with the highest recorded F_1-measure were RL + UN, which generated rules with negation; and RL + BestPosRule, which did not generate rules with negation, as shown in Table 7.14. RL + UN could be considered better in that it generated a smaller average number of rules, 34.5 less rules compared to RL + BestPosRule. The worst RL strategy was RL + UP-UN-Ov. When IG was used, both, RL + Ov, which did not generate rules with negation, and RL + BestRule, which generated rules with negation, recorded the best results. RL + BestRule, however, achieved this result with 14.7 less rules. The worst RL strategy was RL + UP. The use of IG enabled the RL strategies to produce rulesets with higher percentages of rules with negation and smaller rulesets compared to the use of χ^2. However, the use of χ^2 enabled the RL strategies to generate higher F_1-measure

values as compared to the use of IG with the exception of RL + BestStrategy (equal) and RL + BestRule (lower).

Strategies	F_1	Average # of rules	Average # of rules with negation	% of rules with negation
χ^2				
RL + UP	0.804	159.9	0.0	0.0
RL + UN	**0.831**	84.5	40.0	47.3
RL + Ov	0.830	108.9	0.0	0.0
RL + UP-UN-Ov	0.802	159.9	0.1	0.1
RL + UN-UP-Ov	0.824	91.6	42.9	46.8
RL + BestStrategy	0.824	97.0	31.7	32.7
RL + BestPosRule	**0.831**	119.0	0.0	0.0
RL + BestRule	0.823	96.8	48.9	50.5
IG				
RL + UP	0.791	145.5	0.0	0.0
RL + UN	0.822	78.1	43.6	55.8
RL + Ov	**0.827**	97.8	0.0	0.0
RL + UP-UN-Ov	0.792	145.5	1.9	1.3
RL + UN-UP-Ov	0.820	80.5	43.8	54.4
RL + BestStrategy	0.824	83.4	34.8	41.7
RL + BestPosRule	0.815	106.1	0.0	0.0
RL + BestRule	**0.827**	83.1	50.3	60.5

Table 7.14: Micro-averaged F_1-measure and average number of rules for the classification of the SAVSNET dataset using χ^2 and IG for feature selection and the mixed representation

Techniques	χ^2	Rank	IG	Rank
2-gram				
RL	0.814	1	0.795	1
SMO	0.784	2	0.747	2
NB	0.752	3	0.730	5
JRip	0.634	6	0.627	6
OlexGreedy	0.742	4	0.734	4
OlexGA	0.740	5	0.745	3
Mixed				
RL	0.831	4	0.827	3
SMO	0.839	1	0.837	2
NB	0.835	2	0.841	1
JRip	0.803	5	0.792	5
OlexGreedy	0.779	6	0.778	6
OlexGA	0.832	3	0.819	4

Table 7.15: Micro-averaged F_1-measure for the classification of the SAVSNET dataset using χ^2 and IG for feature selection for the best RL strategy in comparison with the other machine learning techniques using the 2-gram and mixed representations

In comparison to the other machine learning techniques, as shown in Table 7.15, the best RL strategy was ranked first when the 2-gram representation was considered, regardless of the feature selection technique adopted. This was followed by SMO. The worst identified technique was JRip. However, when the mixed representation was used, the best RL strategy was ranked fourth when χ^2 was used, and third when IG was used. When comparing only rule-based techniques (not including SMO and NB), the best RL strategy was the best technique when the 2-gram representation was used. When the mixed representation was used, the best RL strategy was ranked second (using χ^2) and was the best (using IG).

Overall, with respect to the binary classification of the SAVSNET dataset, the best RL strategy was one that generated rules with negation, except for when χ^2 was used in the 2-gram representation. The use of χ^2 also enabled the RL strategies to produce higher F_1-measure values compared to the use of IG. Comparison with the other machine learning techniques showed that the best RL strategy performed well, especially when the 2-gram representation was used and averagely when the mixed representation was used.

7.1.4 Summary of Evaluation Using the N-gram Phrase Representation

The evaluation of the eight strategies in the proposed IRL mechanism using binary classification with respect to the 20 Newsgroups, Reuters-8 and SAVSNET datasets and the n-gram phrase representation was discussed in this section. The eight strategies were first compared against one another to see which was the best strategy. The best RL strategy in each case was then compared with the other machine learning techniques, namely SMO, JRip, NB, OlexGreedy and OlexGA in the binary classification task. The discussion on multi-class classification can be found in Appendix B. In the multi-class classification task, the best RL strategy in the mixed representation was compared to the TFPC algorithm with its associated four phrase selection strategies. Both χ^2 and IG were used as the feature selection technique for the IRL mechanism in the experiments conducted. The use of χ^2 generally enabled the RL strategies to produce higher F_1-measure values (in binary classification) and higher accuracies (in multi-class classification) compared to the use of IG, regardless of the text representation used. When IG was used in the mixed representation, smaller rulesets were learnt by the RL strategies. This however was not observed when IG was used in the 2-gram representation. In general, the RL strategies which generated rules with negation (RL + UN, RL + UN-UP-Ov, RL + BestStrategy, RL + BestRule) produced smaller rulesets compared to the RL strategies which did not generate rules with negation (RL + UP, RL + Ov, RL + UP-UN-Ov, RL + BestPosRule). Although RL + UP-UN-Ov generated a very small percentage of rules with negation ($< 1\%$) in some cases, this

RL strategy was not considered to be a strategy which was expected to generate a significant number of rules with negation.

Tables 7.16, 7.17, 7.18 and 7.19 present a summary of the conducted experiments. The tables indicate whether the RL strategies which generated rules with or without negation were best with respect to each dataset in the context of binary and multi-class classification for both the 2-gram and mixed representations (a tick in a column indicates the best performance). With respect to binary classification using the 2-gram representation, as shown in Table 7.16, the RL strategies which generated rules without negation were found to produce better results. The RL strategies that generated rules with negation were only better when χ^2 was used in the Reuters-8 dataset and when IG was used in the 20NG-B and SAVSNET datasets. When using the mixed representation, as shown in Table 7.17, the RL strategies which generated rules with negation produced better results. The only case where the RL strategy which did not produce rules with negation was better was when χ^2 was used with respect to the 20NG-B dataset. In the context of multi-class classification, the RL strategies that generated rules without negation were found to produce better results, largely agreeing with the results reported in Chapter 6 for the multi-class classification task. The only case where the RL strategy which generated rules with negation was better with respect to multi-class classification was in the classification of the Reuters-8 dataset using the mixed representation.

Datasets	χ^2		IG	
	Without Neg	With Neg	Without Neg	With Neg
20NG-A	√		√	
20NG-B	√			√
Reuters-8		√	√	
SAVSNET	√			√

Table 7.16: Summary of the best RL strategy for binary classification using the 2-gram representation with respect to all the datasets

Datasets	χ^2		IG	
	Without Neg	With Neg	Without Neg	With Neg
20NG-A		√		√
20NG-B	√			√
Reuters-8		√		√
SAVSNET	√	√	√	√

Table 7.17: Summary of the best RL strategy for binary classification using the mixed representation with respect to all the datasets

Datasets	χ^2		IG	
	Without Neg	With Neg	Without Neg	With Neg
20NG-A	√		√	
20NG-B	√		√	
Reuters-8	√		√	
SAVSNET	√		√	

Table 7.18: Summary of the best RL strategy for multi-class classification using the 2-gram representation with respect to all the datasets

Datasets	χ^2		IG	
	Without Neg	With Neg	Without Neg	With Neg
20NG-A	√		√	
20NG-B	√		√	
Reuters-8		√		√
SAVSNET	√		√	

Table 7.19: Summary of the best RL strategy for multi-class classification using the mixed representation with respect to all the datasets

Overall, it was found that there was a mix of both RL strategies that generated rules with and without negation that performed the best when the n-gram phrase features were used in the text representation. In the case of binary classification, it was found that the RL strategies that generated rules with negation performed better when the mixed representation was used than when the 2-gram representation was used. In the case of multi-class classification, the RL strategies that generated rules without negation performed better. In the following two sections, the evaluation of the proposed IRL mechanism with respect to the use of keyphrases and fuzzy phrases respectively as the text representation are presented.

7.2 Evaluation Using the Keyphrase Representation

The experimental setup, results and evaluation using keyphrases as the text representation in the proposed Inductive Rule Learning (IRL) mechanism are described in this section. Three different keyphrase representations were used in the evaluation:

1. Keyphrase-2 (KP-2)
2. Keyphrase-3 (KP-3)
3. Mixed (Keyword, Keyphrase-2, Keyphrase-3)

As with the use of the 3-gram representation in Section 7.1, the use of the KP-3 representation was expected to be less effective than the KP-2 representation. This was shown to be true from the results obtained in the experiments conducted. Therefore, in this chapter, only the results for the KP-2 and mixed representations will be discussed. Should the reader be interested in the results obtained using the KP-3 representation, these are shown in Appendix C.

Recall that the extraction of keyphrases was based upon the keywords selected using χ^2 and IG. Both techniques essentially selected a different subset of keywords and therefore, the number of keyphrases extracted from the use of these keywords was also different for χ^2 and IG. A pre-determined percentage of 10% (90% reduction) of the total number of features in each class for the KP-2 and KP-3 representations, and a percentage of 1% (99% reduction) for the mixed representation was used with χ^2 and IG to select keyphrases to be used for the text representation.

7.2.1 Experiments Using the 20 Newsgroups Dataset

This sub-section discusses the evaluation of the eight strategies in the proposed IRL mechanism using the 20 Newsgroups dataset. Table 7.20 shows the number of keyphrase features (KP-2, KP-3, mixed) used for representation of each class in the 20 Newsgroups dataset, selected using χ^2 and IG.

For the binary classification task, the micro-averaged F_1-measure and the average number of rules generated by the eight strategies in the IRL mechanism using χ^2 and IG for the 20NG-A dataset is shown in Table 7.21 for the KP-2 representation and Table 7.22 for the mixed representation. Tables 7.24 and 7.25 show the same results but for the 20NG-B dataset. The micro-averaged F_1-measure for the other machine learning techniques in comparison with the best RL strategy for the classification of the 20NG-A and 20NG-B datasets are shown in Tables 7.23 and 7.26 respectively.

When the KP-2 representation was used for the 20NG-A dataset, the best RL strategies were RL + UP and RL + UP-UN-Ov, regardless of the feature selection technique used as shown in Table 7.21. While RL + UP did not generate rules with negation, RL + UP-UN-Ov generated a very small percentage of rules with negation. The best strategies, however, generated the largest ruleset. The worst performing strategy was RL + Ov. The percentage of rules with negation generated by the RL strategies was lower when χ^2 was used than when IG was used. However, the F_1-measure values generated by the RL strategies were higher when χ^2 was used than when IG was used. In addition, when χ^2 was used, smaller rulesets were generated by the RL strategies, compared to when IG was used, with the exception of RL + UN-UP-Ov in this case.

Class name	KP-2		KP-3		Mixed	
	χ^2	IG	χ^2	IG	χ^2	IG
20NG-A						
alt.atheism	3,252	3,726	7,530	8,611	330	407
comp.sys.ibm.pc.hardware	2,404	2,722	5,281	5,923	395	416
comp.windows.x	4,260	5,111	8,769	10,507	599	640
misc.forsale	2,003	2,214	3,749	4,134	244	235
rec.motorcycles	1,631	1,767	3,247	3,550	179	205
rec.sport.baseball	2,650	2,973	5,507	6,192	346	357
sci.electronics	1,914	2,426	3,740	4,793	190	204
sci.med	3,250	4,287	6,553	8,803	389	419
talk.politics.mideast	5,613	6,285	12,494	14,061	669	726
talk.religion.misc	3,269	3,919	7,334	8,682	344	422
total	30,246	35,430	64,204	75,256	3,685	4,031
20NG-B						
comp.graphics	2,938	3,012	5,919	6,115	394	414
comp.os.ms-windows.misc	6,427	7,132	13,864	15,349	734	691
comp.sys.mac.hardware	2,036	2,374	4,371	5,033	309	301
rec.autos	1,905	2,469	3,909	5,073	246	255
rec.sport.hockey	3,560	3,802	7,208	7,696	541	550
sci.crypt	3,399	4,179	7,487	9,023	446	493
sci.space	3,198	4,314	6,453	8,646	383	423
soc.religion.christian	4,756	5,187	11,386	12,297	640	699
talk.politics.guns	3,354	4,248	7,417	9,250	414	481
talk.politics.misc	4,090	5,403	9,350	12,245	437	533
total	35,663	42,120	77,364	90,727	4,544	4,840

Table 7.20: The number of keyphrase features used for the representation of each class in the 20 Newsgroups dataset

When the mixed representation was used for the 20NG-A dataset, the best strategies were once again RL + UP and RL + UP-UN-Ov, regardless of the feature selection techniques used, as shown in Table 7.22. RL + BestRule was very close behind, achieving the result with a much smaller ruleset; an average of 175.8 less rules when χ^2 was used, and an average of 245.5 less rules when IG was used. The worst strategy was RL + UN, which had the smallest ruleset. The percentage of rules with negation generated by the RL strategies was much higher when IG was used compared to when χ^2 was used. However, the use of IG in the mixed representation resulted in the generation of slightly more rules than when χ^2 was used, with the exception of RL + UN-UP-Ov. The higher number of rules in each case indicated that the feature selection technique used had identified a higher number of phrases as being significant. As phrases were generally more unique and could occur in less documents, more rules would have to be learnt to "cover" all the documents in the dataset. The use of χ^2 resulted in the RL strategies generating slightly higher F_1-measure values compared to the use of IG, with the exception of RL + Ov (equal).

Strategies	F_1	Average # of rules	Average # of rules with negation	% of rules with negation
		χ^2		
RL + UP	**0.920**	2000.5	0.0	0.0
RL + UN	0.898	1626.2	162.9	10.0
RL + Ov	0.894	1870.3	0.0	0.0
RL + UP-UN-Ov	**0.920**	2000.5	7.9	0.4
RL + UN-UP-Ov	0.901	1708.4	172.0	10.1
RL + BestStrategy	0.911	1677.3	111.3	6.6
RL + BestPosRule	0.907	1677.9	0.0	0.0
RL + BestRule	0.910	1673.1	157.1	9.4
		IG		
RL + UP	**0.874**	2233.0	0.0	0.0
RL + UN	0.838	1631.0	444.6	27.3
RL + Ov	0.824	2101.2	0.0	0.0
RL + UP-UN-Ov	**0.874**	2233.0	6.1	0.3
RL + UN-UP-Ov	0.839	1636.1	445.8	27.2
RL + BestStrategy	0.865	1686.9	246.3	14.6
RL + BestPosRule	0.864	1756.1	0.0	0.0
RL + BestRule	0.861	1692.3	429.8	25.4

Table 7.21: Micro-averaged F_1-measure and the average number of rules for the classification of the 20NG-A dataset using χ^2 and IG for feature selection and the KP-2 representation

Table 7.23 shows that the best RL strategy was the best technique when using χ^2 with the KP-2 representation compared to the other machine learning techniques for the 20NG-A dataset. This strategy was second best behind SMO when IG was used. This strategy was also second best behind SMO when the mixed representation was used regardless of the feature selection technique used. In all the cases, NB was the worst technique. When comparing only the rule-based techniques (not including SMO and NB), the best RL strategy outperformed all the other machine learning techniques regardless of the feature selection technique or the representation used. All the techniques compared had higher F_1-measure values when χ^2 was used compared to when IG was used.

Table 7.24 shows the results when using the KP-2 representation for the 20NG-B dataset. When χ^2 was used, RL + UP-UN-Ov was the best strategy. It generated a very small percentage of rules with negation and had the highest number of rules. The worst technique was RL + Ov. When IG was used, the best strategies were RL + UP and RL + UP-UN-Ov, while the worst was again RL + Ov. Generally, very large rulesets were learnt because of the uniqueness of phrases, which resulted in more rules being generated to "cover" all the documents in the dataset. The percentage of rules with negation that was generated by the RL strategies was lower when χ^2 was used

compared to when IG was used. However, the use of χ^2 enabled the RL strategies to obtain higher F_1-measure values.

Strategies	F_1	Average # of rules	Average # of rules with negation	% of rules with negation
χ^2				
RL + UP	**0.888**	825.9	0.0	0.0
RL + UN	0.804	536.4	156.8	29.2
RL + Ov	0.842	866.6	0.0	0.0
RL + UP-UN-Ov	**0.888**	825.9	0.1	0.0
RL + UN-UP-Ov	0.826	624.7	173.6	27.8
RL + BestStrategy	0.884	645.3	101.9	15.8
RL + BestPosRule	0.882	673.7	0.0	0.0
RL + BestRule	0.885	650.1	127.9	19.7
IG				
RL + UP	**0.880**	900.9	0.0	0.0
RL + UN	0.799	573.7	171.0	29.8
RL + Ov	0.842	921.5	0.0	0.0
RL + UP-UN-Ov	**0.880**	900.9	1.4	0.2
RL + UN-UP-Ov	0.802	621.4	187.4	30.2
RL + BestStrategy	0.877	663.7	122.2	18.4
RL + BestPosRule	0.875	701.3	0.0	0.0
RL + BestRule	0.879	655.4	154.6	23.6

Table 7.22: Micro-averaged F_1-measure and the average number of rules for the classification of the 20NG-A dataset using χ^2 and IG for feature selection and the mixed representation

Techniques	χ^2	Rank	IG	Rank
KP-2				
RL	0.920	1	0.874	2
SMO	0.905	2	0.878	1
NB	0.704	6	0.646	6
JRip	0.785	5	0.716	5
OlexGreedy	0.862	4	0.820	4
OlexGA	0.871	3	0.823	3
Mixed				
RL	0.888	2	0.880	2
SMO	0.891	1	0.885	1
NB	0.728	6	0.725	6
JRip	0.827	5	0.817	5
OlexGreedy	0.882	4	0.875	4
OlexGA	0.886	3	0.877	3

Table 7.23: Micro-averaged F_1-measure for the classification of the 20NG-A dataset using χ^2 and IG for feature selection for the best RL strategy in comparison with the other machine learning techniques using the KP-2 and mixed representation

Strategies	F_1	Average # of rules	Average # of rules with negation	% of rules with negation
		χ^2		
RL + UP	0.933	1908.8	0.0	0.0
RL + UN	0.911	1544.8	158.1	10.2
RL + Ov	0.908	1803.6	0.0	0.0
RL + UP-UN-Ov	**0.939**	1908.8	4.3	0.2
RL + UN-UP-Ov	0.914	1642.1	176.6	10.8
RL + BestStrategy	0.920	1565.5	107.8	6.9
RL + BestPosRule	0.919	1589.9	0.0	0.0
RL + BestRule	0.919	1565.9	153.9	9.8
		IG		
RL + UP	**0.896**	2159.2	0.0	0.0
RL + UN	0.870	1550.8	412.3	26.6
RL + Ov	0.854	2042.7	0.0	0.0
RL + UP-UN-Ov	**0.896**	2159.1	7.0	0.3
RL + UN-UP-Ov	0.871	1559.0	414.0	26.6
RL + BestStrategy	0.889	1567.2	238.7	15.2
RL + BestPosRule	0.878	1604.8	0.0	0.0
RL + BestRule	0.884	1567.4	408.9	26.1

Table 7.24: Micro-averaged F_1-measure and the average number of rules for the classification of the 20NG-B dataset using χ^2 and IG for feature selection and the KP-2 representation

Table 7.25 shows the results of using the mixed representation for the 20NG-B dataset. The best strategies were RL + UP and RL + UP-UN-Ov, which had the largest ruleset while the worst was RL + UN, which had the smallest ruleset, regardless of the feature selection technique used. When χ^2 was used, RL + BestStrategy, which generated rules with negation, was the second best strategy, achieving its results with 177.0 less rules. When IG was used, RL + BestStrategy and RL + BestPosRule were equally second best, but RL + BestStrategy had a smaller ruleset, 306.7 less rules than the best strategies. Again, the use of IG enabled the RL strategies to produce rulesets with higher percentage of rules with negation. However, using χ^2, higher F_1-measure values were achieved by the RL strategies compared to using IG.

Compared to the other machine learning techniques, the best RL strategy was the best technique when χ^2 was used, regardless of the representation used, as shown in Table 7.26. When IG was used, the best RL strategy was second behind SMO using the KP-2 representation and came third behind SMO and OlexGA using the mixed representation. NB was the worst strategy in all cases. When comparing only the rule-based techniques (not including SMO and NB), the best RL strategy was the best technique when χ^2 was used. It was also the best when IG was used with the KP-2 representation and second best when IG was used with the mixed representation.

Strategies	F_1	Average # of rules	Average # of rules with negation	% of rules with negation
χ^2				
RL + UP	**0.922**	908.1	0.0	0.0
RL + UN	0.805	515.3	116.2	22.5
RL + Ov	0.864	904.7	0.0	0.0
RL + UP-UN-Ov	**0.922**	908.1	1.0	0.1
RL + UN-UP-Ov	0.840	657.0	161.8	24.6
RL + BestStrategy	0.910	731.1	93.9	12.8
RL + BestPosRule	0.909	760.7	0.0	0.0
RL + BestRule	0.909	734.8	121.1	16.5
IG				
RL + UP	**0.903**	1012.8	0.0	0.0
RL + UN	0.823	597.4	175.9	29.4
RL + Ov	0.849	970.3	0.0	0.0
RL + UP-UN-Ov	**0.903**	1012.8	2.6	0.3
RL + UN-UP-Ov	0.830	635.0	196.1	30.9
RL + BestStrategy	0.896	706.1	133.8	18.9
RL + BestPosRule	0.896	759.6	0.0	0.0
RL + BestRule	0.895	712.4	171.6	24.1

Table 7.25: Micro-averaged F_1-measure and the average number of rules for the classification of the 20NG-B dataset using χ^2 and IG for feature selection and the mixed representation

Techniques	χ^2	Rank	IG	Rank
KP-2				
RL	0.939	1	0.896	2
SMO	0.895	3	0.901	1
NB	0.734	6	0.666	6
JRip	0.844	5	0.805	5
OlexGreedy	0.890	4	0.850	4
OlexGA	0.897	2	0.856	3
Mixed				
RL	0.922	1	0.903	3
SMO	0.920	2	0.909	1
NB	0.767	6	0.747	6
JRip	0.865	5	0.841	5
OlexGreedy	0.901	4	0.895	4
OlexGA	0.918	3	0.907	2

Table 7.26: Micro-averaged F_1-measure for the classification of the 20NG-A dataset using χ^2 and IG for feature selection for the best RL strategy in comparison with the other machine learning techniques using the KP-2 and mixed representation

Overall, with respect to the binary classification of the 20 Newsgroups dataset, the strategies RL + UP and RL + UP-UN-Ov were shown to be better than the other strategies when using the keyphrase representation, albeit with much bigger rulesets. These two strategies did not differ much, suggesting that the UP sub-space was rarely empty and therefore, both strategies generated almost the same ruleset. The use of χ^2 enabled the RL strategies to obtain higher F_1-measure values compared to the use of IG. Compared to the other machine learning techniques, the best RL strategy was among the top performers; outperforming the other techniques and competitive with SMO.

7.2.2 Experiments Using the Reuters-8 Dataset

This sub-section discusses the evaluation of the eight strategies in the proposed IRL mechanism using the Reuters-8 dataset. Table 7.27 shows the number of keyphrases (KP-2, KP-3 and mixed) used for the representation of each class in the Reuters-8 dataset, selected using χ^2 and IG.

Class name	KP-2		KP-3		Mixed	
	χ^2	IG	χ^2	IG	χ^2	IG
acq	5,153	3,637	12,579	9,675	830	776
crude	1,694	1,832	3,825	4,090	242	248
earn	3,298	3,379	8,074	8,224	563	511
grain	257	371	485	697	37	39
interest	688	804	1,542	1,774	97	104
money-fx	1,494	1,632	3,440	3,704	201	207
ship	448	561	851	1,060	49	56
trade	1,780	1,959	4,210	4,585	252	259
total	14,812	14,175	35,006	33,809	2,271	2,200

Table 7.27: The number of keyphrase features used for the representation of each class in the Reuters-8 dataset

For the binary classification task, the micro-averaged F_1-measure and the average number of rules generated by the eight strategies in the IRL mechanism using χ^2 and IG for the Reuters-8 dataset are shown in Table 7.28 for the KP-2 representation, and in Table 7.29 for the mixed representation. The results obtained from the other machine learning techniques were then compared with the best RL strategy. These results are shown in Table 7.30.

From Table 7.28, it was noted that the best RL strategies when χ^2 was used were RL + UP and RL + UP-UN-Ov, which did not generate any rules with negation but generated the largest rulesets. Both strategies achieved identical results and number of rules, signifying that the UP sub-space was never empty. RL + BestPosRule and

RL + BestRule were the second best strategies, with RL + BestRule, which generated rules with negation, having a smaller ruleset, 452.1 fewer rules compared to the best strategies. Although the results for all the strategies were fairly close, the worst strategy was RL + Ov. When IG was used, the best RL strategy was RL + BestStrategy, which generated rules with negation. RL + BestPosRule and RL + BestRule were equally second best. The worst strategy was again RL + Ov while the strategies which generated the largest ruleset were RL + UP and RL + UP-UN-Ov. When using IG, the RL strategies generated a higher percentage of rules with negation. However, the use of χ^2 enabled the RL strategies to obtain higher F_1-measure values.

Strategies	F_1	Average # of rules	Average # of rules with negation	% of rules with negation
χ^2				
RL + UP	**0.929**	957.2	0.0	0.0
RL + UN	0.908	454.5	181.4	39.9
RL + Ov	0.898	774.7	0.0	0.0
RL + UP-UN-Ov	**0.929**	957.2	0.0	0.0
RL + UN-UP-Ov	0.908	472.9	184.8	39.1
RL + BestStrategy	0.922	510.7	123.1	24.1
RL + BestPosRule	0.923	568.7	0.0	0.0
RL + BestRule	0.923	505.1	206.1	40.8
IG				
RL + UP	0.895	968.9	0.0	0.0
RL + UN	0.893	436.5	220.0	50.4
RL + Ov	0.887	742.5	0.0	0.0
RL + UP-UN-Ov	0.895	968.9	0.2	0.0
RL + UN-UP-Ov	0.892	452.0	223.6	49.5
RL + BestStrategy	**0.913**	475.4	124.6	26.2
RL + BestPosRule	0.909	547.6	0.0	0.0
RL + BestRule	0.909	475.0	239.6	50.4

Table 7.28: Micro-averaged F_1-measure and the average number of rules for the classification of the Reuters-8 dataset using χ^2 and IG for feature selection and the KP-2 representation

Table 7.29 shows that the best RL strategy for the mixed representation when χ^2 was used was RL + BestRule, while RL + BestStrategy and RL + BestRule were equally the best when IG was used. All these strategies generated rules with negation. Regardless of the feature selection technique used, the worst strategy was RL + UN, although it had the smallest ruleset. RL + UP and RL + UP-UN-Ov recorded the same results and the same average number of rules, with only a very small percentage of rules with negation generated by RL + UP-UN-Ov, suggesting that the UP subspace was very rarely empty. These two strategies also generated the largest ruleset.

The use of IG resulted in the RL strategies generating a higher average number of rules (except for RL + BestStrategy and RL + BestRule) and a higher percentage of rules with negation. Generally, when χ^2 was used, higher F_1-measure values were achieved by the RL strategies (except for RL + UN, RL + Ov and RL + UN-UP-Ov).

Strategies	F_1	Average # of rules	Average # of rules with negation	% of rules with negation
χ^2				
RL + UP	0.956	400.6	0.0	0.0
RL + UN	0.754	163.8	58.8	35.9
RL + Ov	0.907	292.9	0.0	0.0
RL + UP-UN-Ov	0.956	400.6	0.1	0.0
RL + UN-UP-Ov	0.835	202.0	75.3	37.3
RL + BestStrategy	0.956	262.4	56.6	21.6
RL + BestPosRule	0.949	290.4	0.0	0.0
RL + BestRule	**0.957**	267.5	72.2	27.0
IG				
RL + UP	0.929	471.3	0.0	0.0
RL + UN	0.861	201.0	102.1	50.8
RL + Ov	0.918	362.2	0.0	0.0
RL + UP-UN-Ov	0.929	471.3	0.7	0.1
RL + UN-UP-Ov	0.862	221.5	110.7	50.0
RL + BestStrategy	**0.939**	257.1	87.0	33.8
RL + BestPosRule	0.934	299.6	0.0	0.0
RL + BestRule	**0.939**	260.6	108.3	41.6

Table 7.29: Micro-averaged F_1-measure and the average number of rules for the classification of the Reuters-8 dataset using χ^2 and IG for feature selection and the mixed representation

When compared to the other machine learning techniques, as shown in Table 7.30, the best RL strategy was second best behind SMO when χ^2 was used, regardless of the text representation used. When IG was used, the best RL strategy was third behind SMO and OlexGA using the KP-2 representation, and was second behind SMO using the mixed representation. In all the cases, NB was the worst technique. When comparing only the rule-based techniques (not including SMO and NB), the best RL strategy was the best technique overall, except when IG was used with the KP-2 representation, whereby it came second behind OlexGA.

Overall, the RL strategies that generated rules with negation performed best except when χ^2 was used with the KP-2 representation. Although the use of χ^2 resulted in the RL strategies generating a lower percentage of rules with negation, it enabled higher F_1-measure values to be produced. Compared to the other machine learning techniques,

the best RL strategy generally outperformed all the techniques and was competitive with SMO and OlexGA.

Techniques	χ^2	Rank	IG	Rank
KP-2				
RL	0.929	2	0.913	3
SMO	0.953	1	0.940	1
NB	0.843	6	0.800	6
JRip	0.907	5	0.900	5
OlexGreedy	0.915	4	0.912	4
OlexGA	0.921	3	0.917	2
Mixed				
RL	0.957	2	0.939	2
SMO	0.970	1	0.956	1
NB	0.901	6	0.878	6
JRip	0.949	5	0.931	5
OlexGreedy	0.952	4	0.934	4
OlexGA	0.954	3	0.937	3

Table 7.30: Micro-averaged F_1-measure for the classification of the Reuters-8 dataset using χ^2 and IG for feature selection for the best RL strategy in comparison with the other machine learning techniques using the KP-2 and mixed representation

7.2.3 Experiments Using the SAVSNET Dataset

The evaluation of the eight strategies in the proposed IRL mechanism using the SAVSNET dataset is discussed in this sub-section. Table 7.31 shows the number of keyphrases (KP-2, KP-3, mixed) used for the representation of each class in the SAVSNET dataset, selected using χ^2 and IG.

Class name	KP-2		KP-3		Mixed	
	χ^2	IG	χ^2	IG	χ^2	IG
aggression	18	32	27	48	1	1
diarrhoea	421	402	792	767	38	28
pruritus	532	563	963	1,025	38	20
vomit	262	308	456	530	23	26
total	1,233	1,305	2,238	2,370	100	75

Table 7.31: The number of keyphrase features used for the representation of each class in the SAVSNET dataset

For the binary classification task, the micro-averaged F_1-measure and the average number of rules generated by the eight strategies in the IRL mechanism using χ^2 and

IG for the SAVSNET dataset are shown in Tables 7.32 and 7.33 for the KP-2 and mixed representation respectively. Table 7.34 shows the best RL strategy in comparison with the other machine learning techniques.

From Table 7.32, when the KP-2 representation was used, it was noted that the best RL strategies were RL + UP and RL + UP-UN-Ov (when χ^2 was used) and RL + BestPosRule (when IG was used). All these strategies did not generate rules with negation, except for RL + UP-UN-Ov which generated a very small percentage ($<$ 1%) of rules with negation. Regardless of the feature selection technique used, the worst strategy was RL + UN. RL + UP and RL + UP-UN-Ov produced the largest ruleset while RL + UN produced the smallest. It was noted that when IG was used, the average number of rules generated by the RL strategies was higher. In addition, the percentage of rules with negation was also higher (except for RL + UP-UN-Ov). However, when χ^2 was used, the F_1-measure values generated by the RL strategies were higher compared to when IG was used.

Strategies	F_1	Average # of rules	Average # of rules with negation	% of rules with negation
χ^2				
RL + UP	**0.901**	183.5	0.0	0.0
RL + UN	0.886	145.0	24.4	16.8
RL + Ov	0.888	176.1	0.0	0.0
RL + UP-UN-Ov	**0.901**	183.5	0.8	0.4
RL + UN-UP-Ov	0.888	146.8	25.5	17.4
RL + BestStrategy	0.896	151.7	14.9	9.8
RL + BestPosRule	0.894	164.7	0.0	0.0
RL + BestRule	0.896	150.4	24.7	16.4
IG				
RL + UP	0.877	219.8	0.0	0.0
RL + UN	0.845	151.4	36.9	24.4
RL + Ov	0.846	202.3	0.0	0.0
RL + UP-UN-Ov	0.877	219.8	0.0	0.0
RL + UN-UP-Ov	0.847	153.6	38.0	24.7
RL + BestStrategy	0.875	172.4	23.9	13.9
RL + BestPosRule	**0.883**	189.5	0.0	0.0
RL + BestRule	0.873	173.0	38.7	22.4

Table 7.32: Micro-averaged F_1-measure and the average number of rules for the classification of the SAVSNET dataset using χ^2 and IG for feature selection and the KP-2 representation

Table 7.33 shows the results for the classification of the SAVSNET dataset using the mixed representation, whereby the best strategy was RL + BestStrategy (when χ^2 was used), which generated rules with negation and RL + Ov (when IG was used),

which did not generate rules with negation. The equal worst strategies, regardless of the feature selection technique used, were RL + UP and RL + UP-UN-Ov. These two strategies also generated the same average number of rules. RL + UP-UN-Ov did not generate any rules with negation, suggesting that the UP sub-space was never empty. RL + Ov generated the largest ruleset, while RL + UN had the smallest ruleset in this case. It was observed that the average number of rules and the percentage of rules with negation generated by the RL strategies were lower when IG was used compared to when χ^2 was used. However, when using χ^2, the RL strategies again generated higher F_1-measure values.

Strategies	F_1	Average # of rules	Average # of rules with negation	% of rules with negation
χ^2				
RL + UP	0.741	51.7	0.0	0.0
RL + UN	0.787	39.8	18.7	47.0
RL + Ov	0.816	53.5	0.0	0.0
RL + UP-UN-Ov	0.741	51.7	0.0	0.0
RL + UN-UP-Ov	0.784	43.2	18.4	42.6
RL + BestStrategy	**0.844**	45.4	11.6	25.6
RL + BestPosRule	0.830	47.4	0.0	0.0
RL + BestRule	0.828	45.1	11.7	25.9
IG				
RL + UP	0.537	31.0	0.0	0.0
RL + UN	0.665	26.0	10.7	41.2
RL + Ov	**0.688**	33.1	0.0	0.0
RL + UP-UN-Ov	0.537	31.0	0.0	0.0
RL + UN-UP-Ov	0.639	27.5	11.1	40.4
RL + BestStrategy	0.680	30.0	4.6	15.3
RL + BestPosRule	0.672	30.4	0.0	0.0
RL + BestRule	0.677	30.2	5.2	17.2

Table 7.33: Micro-averaged F_1-measure and the average number of rules for the classification of the SAVSNET dataset using χ^2 and IG for feature selection and the mixed representation

When comparing the best RL strategy to the other machine learning techniques using the KP-2 representation, the RL strategy was found to be the best technique when χ^2 was used and equal best with SMO when IG was used. NB and JRip were the worst techniques respectively when χ^2 and IG were used. When using the mixed representation, the best RL strategy did not perform as well, coming in third behind SMO and OlexGA when χ^2 was used, and fourth behind SMO, NB and JRip when IG was used. When comparing only the rule-based techniques (not including SMO and NB), the best RL strategy produced the best performance using the KP-2 representation and second best when using the mixed representation.

Techniques	χ^2	Rank	IG	Rank
KP-2				
RL	0.901	1	0.883	1
SMO	0.891	3	0.883	1
NB	0.820	6	0.810	5
JRip	0.840	5	0.807	6
OlexGreedy	0.855	4	0.835	4
OlexGA	0.892	2	0.877	3
Mixed				
RL	0.844	3	0.688	4
SMO	0.855	1	0.808	1
NB	0.831	6	0.806	2
JRip	0.833	5	0.792	3
OlexGreedy	0.843	4	0.675	6
OlexGA	0.845	2	0.677	5

Table 7.34: Micro-averaged F_1-measure for the classification of the SAVSNET dataset using χ^2 and IG for feature selection for the best RL strategy in comparison with the other machine learning techniques using the KP-2 and mixed representation

Overall, with respect to the binary classification of the SAVSNET dataset, the best RL strategy was one that did not generate rules with negation, except for when χ^2 was used in the mixed representation. When using χ^2, higher F_1-measure values were obtained by the RL strategies compared to when using IG. Comparison with the other machine learning techniques showed that the best RL strategy produced the best performance when the KP-2 representation was used and only had an average performance when the mixed representation was used.

7.2.4 Summary of Evaluation Using the Keyphrase Representation

This sub-section summarizes the evaluation of the eight strategies in the proposed IRL mechanism using both binary and multi-class classification with respect to the 20 Newsgroups, Reuters-8 and SAVSNET datasets using the keyphrase representation. The eight RL strategies were first compared against one another to identify the best RL strategy. In the binary classification task, the best RL strategy in each case was then compared to the other machine learning techniques, namely SMO, JRip, NB, OlexGreedy and OlexGA. The discussion on multi-class classification can be found in Appendix C. In the multi-class classification task, the best RL strategy in the mixed representation was compared to the TFPC algorithm with its four phrase selection strategies. Two feature selection techniques, χ^2 and IG, were used for the IRL mechanism in the experiments conducted. It was observed that the use of χ^2 generally enabled the RL strategies to produce higher F_1-measure values (in binary classification) and higher accuracies (in multi-class classification) compared to the use of IG,

regardless of the text representation used. In general, the RL strategies that generated rules with negation (RL + UN, RL + UN-UP-Ov, RL + BestStrategy, RL + BestRule) produced smaller rulesets compared to the RL strategies which did not generate rules with negation (RL + UP, RL + Ov, RL + UP-UN-Ov, RL + BestPosRule). Although RL + UP-UN-Ov generated a very small percentage of rules with negation ($< 1\%$) in some cases, this RL strategy was not considered to be a strategy which was expected to generate a significant number of rules with negation.

Tables 7.35, 7.36, 7.37 and 7.38 present a summary of the conducted experiments. The tables indicate whether the RL strategies which generated rules with or without negation were best with respect to each dataset in the context of binary and multi-class classification for both the KP-2 and mixed representations (a tick in a column indicates the best performance). With respect to binary classification using the KP-2 representation, as shown in Table 7.35, the RL strategies which generated rules without negation were found to produce better results. There was only one case where the RL strategy that generated rules with negation was better, which was in the case of the Reuters-8 dataset when IG was used. When using the mixed representation, as shown in Table 7.36, it was noted that the RL strategies which generated rules with negation were better with respect to the Reuters-8 and SAVSNET datasets when χ^2 was used and only in the Reuters-8 dataset when IG was used. In the multi-class classification setting, the RL strategies that generated rules without negation were found to produce better results with respect to all the datasets, regardless of the feature selection technique or representation used, as shown in Tables 7.37 and 7.38.

Overall, with respect to the use of keyphrase features in the text representation, it was found that the RL strategies that generated rules without negation performed better more often than the RL strategies that generated rules with negation. In the following section, the evaluation on the use of fuzzy phrases as the text representation in the context of the eight strategies in the proposed IRL mechanism is discussed.

Datasets	χ^2		IG	
	Without Neg	With Neg	Without Neg	With Neg
20NG-A	√		√	
20NG-B	√		√	
Reuters-8	√			√
SAVSNET	√		√	

Table 7.35: Summary of the best RL strategy for binary classification using the KP-2 representation with respect to all the datasets

Datasets	χ^2		IG	
	Without Neg	With Neg	Without Neg	With Neg
20NG-A	√		√	
20NG-B	√		√	
Reuters-8		√		√
SAVSNET		√	√	

Table 7.36: Summary of the best RL strategy for binary classification using the mixed representation with respect to all the datasets

Datasets	χ^2		IG	
	Without Neg	With Neg	Without Neg	With Neg
20NG-A	√		√	
20NG-B	√		√	
Reuters-8	√		√	
SAVSNET	√		√	

Table 7.37: Summary of the best RL strategy for multi-class classification using the KP-2 representation with respect to all the datasets

Datasets	χ^2		IG	
	Without Neg	With Neg	Without Neg	With Neg
20NG-A	√		√	
20NG-B	√		√	
Reuters-8	√		√	
SAVSNET	√		√	

Table 7.38: Summary of the best RL strategy for multi-class classification using the mixed representation with respect to all the datasets

7.3 Evaluation Using the Fuzzy Phrase Representation

The experimental setup, results and evaluation using fuzzy phrases as the text representation in the proposed Inductive Rule Learning (IRL) mechanism are described in this section. Three different fuzzy phrase representations were used in the evaluation:

1. FuzzyPhrase-1 (FP-1)
2. FuzzyPhrase-2 (FP-2)
3. Mixed (Keyword, FuzzyPhrase-1, FuzzyPhrase-2)

Similar to case in n-gram phrases and keyphrases, the use of the FP-2 representation was expected to be less effective than the FP-1 representation. The results obtained from the experiments conducted demonstrated this to be true. Therefore, only the results for the FP-1 and mixed representations will be discussed in this section. If the reader is interested in the results for the use of the FP-2 representation, these have been included in Appendix D.

Recall that the extraction of fuzzy phrases was based upon the keywords selected using χ^2 and IG. Therefore, different subsets of keywords were selected by each technique, resulting in different numbers of fuzzy phrases being extracted using χ^2 and IG. A pre-determined percentage of 10% (90% reduction) of the total number of features in each class for the FP-1, FP-2 and mixed representations was again used with χ^2 and IG to select a subset of fuzzy phrases. The pre-determined percentage of 1% (99% reduction) was not used for the mixed representation (as in the case of the evaluations described for n-gram phrases and keyphrases) as the number of fuzzy phrases extracted was manageable in terms of computational power.

7.3.1 Experiments Using the 20 Newsgroups Dataset

This sub-section discusses the evaluation of the eight strategies in the proposed IRL mechanism using the 20 Newsgroups dataset. Table 7.39 shows the number of fuzzy phrase features (FP-1, FP-2, mixed) used to represent each class in the 20 Newsgroups dataset, selected using χ^2 and IG.

For the binary classification task, the micro-averaged F_1-measure and the average number of rules generated by the eight strategies in the IRL mechanism using χ^2 and IG for the 20NG-A dataset is shown in Table 7.40 for the FP-1 representation and Table 7.41 for the mixed representation. Tables 7.43 and 7.44 show the same results but for the 20NG-B dataset. The micro-averaged F_1-measure for the other machine learning techniques in comparison with the best RL strategy for classification of the 20NG-A and 20NG-B datasets are shown in Tables 7.42 and 7.45 respectively.

From Table 7.40, when the FP-1 representation was used with χ^2, it was observed that all the results recorded were almost the same, with RL + UP, RL + Ov, RL + UP-UN-Ov, RL + BestStrategy, RL + BestPosRule and RL + BestRule all equally sharing the best results. Out of these, RL + BestStrategy, which generated a very small percentage of rules with negation, could be considered slightly better than the rest because it had a slightly smaller ruleset compared to the other strategies with equal results. Almost the same trend could be observed when IG was used, again with RL + BestStrategy considered the best strategy. The use of χ^2 enabled the RL strategies to generate smaller rulesets and obtained higher F_1-measure values compared to the use of IG. Regardless of the feature selection technique used, the percentage of rules with negation generated was very low. The low percentage of rules with negation,

particularly in RL + UN, indicated that the rules learnt using FP-1 phrases either covered very few or did not cover any negative documents, thus resulting in either lowly populated or empty UN sub-space.

Class name	FP-1		FP-2		Mixed	
	χ^2	IG	χ^2	IG	χ^2	IG
20NG-A						
alt.atheism	507	619	598	746	1,186	1,435
comp.sys.ibm.pc.hardware	425	480	487	562	924	1,051
comp.windows.x	822	1,119	920	1,288	1,766	2,399
misc.forsale	357	377	378	406	784	828
rec.motorcycles	199	204	228	241	526	543
rec.sport.baseball	462	549	507	615	1,019	1,203
sci.electronics	186	271	223	311	512	677
sci.med	417	700	470	784	1,034	1,606
talk.politics.mideast	1,132	1,364	1,321	1,612	2,510	3,016
talk.religion.misc	457	615	533	723	1,098	1,438
total	4,964	6,298	5,665	7,288	11,359	14,196
20NG-B						
comp.graphics	474	447	514	491	1,052	1,006
comp.os.ms-windows.misc	1,236	1,380	1,301	1,491	2,527	2,858
comp.sys.mac.hardware	282	341	328	398	669	792
rec.autos	195	303	230	357	529	751
rec.sport.hockey	1,014	1,040	1,142	1,158	2,055	2,091
sci.crypt	542	772	665	920	1,276	1,742
sci.space	472	774	529	879	1,101	1,727
soc.religion.christian	825	951	958	1,098	1,835	2,094
talk.politics.guns	491	758	590	894	1,190	1,740
talk.politics.misc	563	969	679	1,151	1,353	2,204
total	6,094	7,735	6,936	8,837	13,587	17,005

Table 7.39: The number of fuzzy phrase features used for the representation of each class in the 20 Newsgroups dataset

Table 7.41 shows the results obtained when using the mixed representation for the classification of the 20NG-A dataset. Regardless of the feature selection technique used, the best RL strategies were RL + UP and RL + UP-UN-Ov. RL + UP did not generate rules with negation, but RL + UP-UN-Ov generated an average of 0.3 rules with negation when IG was used. Although these two strategies recorded the best results, they had the largest rulesets. The second best strategy was RL + BestStrategy, which generated rules with negation. It had an average of 88.2 fewer rules when χ^2 was used, and 389.3 fewer rules when IG was used, compared to the best RL strategies. The worst strategy was RL + UN, although it had the smallest ruleset. When χ^2 was used, higher F_1-measure values were obtained compared to when IG was used.

Strategies	F_1	Average # of rules	Average # of rules with negation	% of rules with negation
χ^2				
RL + UP	**0.748**	1228.1	0.0	0.0
RL + UN	0.747	1207.0	12.6	1.0
RL + Ov	**0.748**	1214.9	0.0	0.0
RL + UP-UN-Ov	**0.748**	1228.0	2.3	0.2
RL + UN-UP-Ov	0.747	1218.5	12.6	1.0
RL + BestStrategy	**0.748**	1212.5	10.8	0.9
RL + BestPosRule	**0.748**	1213.6	0.0	0.0
RL + BestRule	**0.748**	1212.6	12.5	1.0
IG				
RL + UP	**0.729**	1457.7	0.0	0.0
RL + UN	0.727	1397.4	48.5	3.5
RL + Ov	0.728	1421.9	0.0	0.0
RL + UP-UN-Ov	**0.729**	1457.2	7.4	0.5
RL + UN-UP-Ov	0.726	1417.8	48.4	3.4
RL + BestStrategy	**0.729**	1401.3	40.5	2.9
RL + BestPosRule	**0.729**	1405.5	0.0	0.0
RL + BestRule	**0.729**	1401.7	47.9	3.4

Table 7.40: Micro-averaged F_1-measure and the average number of rules for the classification of the 20NG-A dataset using χ^2 and IG for feature selection and the FP-1 representation

Table 7.42 shows the comparison of the best RL strategy with the other machine learning techniques. The best RL strategy was the best technique in all the cases except when IG was used with the mixed representation, where it came second behind SMO. The worst technique in all cases was NB. When comparing only the rule-based techniques (not including SMO and NB), the best RL strategy was the best technique in all cases.

Table 7.43 shows the results obtained when the FP-1 representation was used for the 20NG-B dataset. When χ^2 was used, very similar results were recorded, with five strategies sharing the equal best results. Out of these five, RL + UN, RL + BestStrategy and RL + BestRule recorded the smallest rulesets. All these three strategies generated rules with negation, although this was only a small proportion of the total number of rules. When IG was used, again the results were very close, but RL + BestPosRule was slightly better. Similar to the case of the 20NG-A dataset, the use of χ^2 produced smaller rulesets and higher F_1-measure values compared to the use of IG. Regardless of the feature selection technique used, the percentage of rules with negation generated was again very low, due to the same reason as stated in the case of the 20NG-A dataset.

Strategies	F_1	Average # of rules	Average # of rules with negation	% of rules with negation
χ^2				
RL + UP	**0.971**	1017.1	0.0	0.0
RL + UN	0.930	810.0	34.4	4.2
RL + Ov	0.952	987.9	0.0	0.0
RL + UP-UN-Ov	**0.971**	1017.1	0.0	0.0
RL + UN-UP-Ov	0.945	933.4	41.9	4.5
RL + BestStrategy	0.962	928.9	32.6	3.5
RL + BestPosRule	0.961	916.0	0.0	0.0
RL + BestRule	0.961	905.8	34.3	3.8
IG				
RL + UP	**0.930**	1352.9	0.0	0.0
RL + UN	0.873	880.0	235.5	26.8
RL + Ov	0.886	1295.5	0.0	0.0
RL + UP-UN-Ov	**0.930**	1352.9	0.3	0.0
RL + UN-UP-Ov	0.873	881.4	236.1	26.8
RL + BestStrategy	0.918	963.6	173.7	18.0
RL + BestPosRule	0.914	1107.8	0.0	0.0
RL + BestRule	0.918	979.6	244.6	25.0

Table 7.41: Micro-averaged F_1-measure and the average number of rules for the classification of the 20NG-A dataset using χ^2 and IG for feature selection and the mixed representation

Techniques	χ^2	Rank	IG	Rank
FP-1				
RL	0.748	1	0.729	1
SMO	0.719	2	0.706	2
NB	0.325	6	0.335	6
JRip	0.620	5	0.576	5
OlexGreedy	0.683	4	0.677	4
OlexGA	0.692	3	0.690	3
Mixed				
RL	0.971	1	0.930	2
SMO	0.958	3	0.936	1
NB	0.782	6	0.717	6
JRip	0.922	5	0.852	5
OlexGreedy	0.951	4	0.894	4
OlexGA	0.962	2	0.903	3

Table 7.42: Micro-averaged F_1-measure for the classification of the 20NG-A dataset using χ^2 and IG for feature selection for the best RL strategy in comparison with the other machine learning techniques using the FP-1 and mixed representation

Strategies	F_1	Average # of rules	Average # of rules with negation	% of rules with negation
		χ^2		
RL + UP	0.788	1467.7	0.0	0.0
RL + UN	**0.789**	1452.5	7.9	0.5
RL + Ov	**0.789**	1457.5	0.0	0.0
RL + UP-UN-Ov	0.788	1467.7	3.2	0.2
RL + UN-UP-Ov	0.787	1466.1	9.0	0.6
RL + BestStrategy	**0.789**	1452.5	6.4	0.4
RL + BestPosRule	**0.789**	1452.7	0.0	0.0
RL + BestRule	**0.789**	1452.5	7.9	0.5
		IG		
RL + UP	0.764	1649.2	0.0	0.0
RL + UN	0.763	1576.5	52.9	3.4
RL + Ov	0.762	1624.2	0.0	0.0
RL + UP-UN-Ov	0.764	1649.0	7.5	0.5
RL + UN-UP-Ov	0.764	1615.5	55.4	3.4
RL + BestStrategy	0.765	1581.9	36.1	2.3
RL + BestPosRule	**0.766**	1586.5	0.0	0.0
RL + BestRule	0.765	1582.0	51.1	3.2

Table 7.43: Micro-averaged F_1-measure and the average number of rules for the classification of the 20NG-B dataset using χ^2 and IG for feature selection and the FP-1 representation

The best RL strategies recorded when the mixed representation was used for the 20NG-B dataset, regardless of the feature selection technique used, were RL + UP and RL + UP-UN-Ov, as shown in Table 7.44. These two strategies did not generate any rules with negation. Although they were the best RL strategies, they had the largest rulesets. The worst strategy was RL + UN, although it generated the smallest ruleset.

Table 7.45 shows the results for the classification of the 20NG-B dataset when comparing the best RL strategy with the other machine learning techniques. The ranking order of the techniques compared was almost identical to that when the 20NG-A dataset was used. The best RL strategy was the top performer for all the cases except when IG was used with the mixed representation, where it came second behind SMO. The worst results was again recorded using NB. When comparing only the rule-based techniques (not including SMO and NB), the best RL strategy achieved the best results.

Overall, with respect to the binary classification of the 20 Newsgroups dataset, all the strategies recorded almost similar results when the FP-1 representation was used. In this case, RL + BestStrategy, which generated rules with negation, was regarded as slightly better as it achieved the results obtained with a slightly smaller ruleset. However, it was observed that not many rules with negation were generated, just a

Strategies	F_1	Average # of rules	Average # of rules with negation	% of rules with negation
		χ^2		
RL + UP	**0.972**	966.8	0.0	0.0
RL + UN	0.929	714.7	44.2	6.2
RL + Ov	0.946	927.2	0.0	0.0
RL + UP-UN-Ov	**0.972**	966.8	0.0	0.0
RL + UN-UP-Ov	0.939	796.4	44.2	5.5
RL + BestStrategy	0.962	847.4	33.1	3.9
RL + BestPosRule	0.964	850.9	0.0	0.0
RL + BestRule	0.963	831.6	40.7	4.9
		IG		
RL + UP	**0.939**	1399.5	0.0	0.0
RL + UN	0.872	801.7	216.8	27.0
RL + Ov	0.883	1319.8	0.0	0.0
RL + UP-UN-Ov	**0.939**	1399.5	0.0	0.0
RL + UN-UP-Ov	0.872	802.1	217.1	27.1
RL + BestStrategy	0.919	975.4	183.4	18.8
RL + BestPosRule	0.923	1089.9	0.0	0.0
RL + BestRule	0.918	978.5	257.2	26.3

Table 7.44: Micro-averaged F_1-measure and the average number of rules for the classification of the 20NG-B dataset using χ^2 and IG for feature selection and the mixed representation

very small percentage for each of the strategies that generated rules with negation. When the mixed representation was used, the strategies RL + UP and RL + UP-UN-Ov, which did not generate rules with negation were shown to be better than the other RL strategies, albeit with bigger rulesets. These two strategies produced identical results, suggesting that the UP sub-space was never empty and therefore, both strategies generated the same ruleset, except in the case of the 20NG-A dataset when IG was used, where a very small proportion of rules with negation was generated by RL + UP-UN-OV. It was also observed that the percentage of rules with negation generated by the RL strategies was higher using the mixed representation than when using the FP-1 representation. This was due to the presence of keywords in the mixed representation, whereby keywords could co-occur more frequently in many documents compared to when fuzzy phrases were used, which led to a tendency for rules to cover more negative documents and therefore populate (to a greater degree) the UN sub-space with features to be used for rule refinement. Compared to the other machine learning techniques, the best RL strategy was the top performer, except in the case when the mixed representation was used with IG, where it came second behind SMO.

Techniques	χ^2	Rank	IG	Rank
FP-1				
RL	0.789	1	0.766	1
SMO	0.764	2	0.744	2
NB	0.347	6	0.355	6
JRip	0.685	5	0.639	5
OlexGreedy	0.715	4	0.702	4
OlexGA	0.725	3	0.712	3
Mixed				
RL	0.972	1	0.939	2
SMO	0.962	3	0.949	1
NB	0.781	6	0.727	6
JRip	0.930	5	0.879	5
OlexGreedy	0.958	4	0.907	4
OlexGA	0.971	2	0.917	3

Table 7.45: Micro-averaged F_1-measure for the classification of the 20NG-B dataset using χ^2 and IG for feature selection for the best RL strategy in comparison with the other machine learning techniques using the FP-1 and mixed representation

7.3.2 Experiments Using the Reuters-8 Dataset

This sub-section discusses the evaluation of the eight strategies in the proposed IRL mechanism using the Reuters-8 dataset. Table 7.46 shows the number of fuzzy phrases (FP-1, FP-2 and mixed) used for the representation of each class in the Reuters-8 dataset, selected using χ^2 and IG.

Class name	FP-1		FP-2		Mixed	
	χ^2	IG	χ^2	IG	χ^2	IG
acq	1,425	621	1,722	756	2,895	1,302
crude	365	426	420	489	764	883
earn	799	829	922	964	1,573	1,634
grain	23	55	25	61	72	135
interest	130	175	148	202	281	372
money-fx	352	417	431	502	758	885
ship	59	96	70	108	158	228
trade	413	509	472	569	855	1,033
total	3,566	3,128	4,210	3,651	7,356	6,472

Table 7.46: The number of fuzzy phrase features used for the representation of each class in the Reuters-8 dataset

For the binary classification task, the micro-averaged F_1-measure and the average number of rules generated by the eight strategies in the IRL mechanism using χ^2 and IG for the Reuters-8 dataset are shown in Table 7.47 for the FP-1 representation, and in Table 7.48 for the mixed representation. The results obtained from the other machine

learning techniques were then compared with the best RL strategy. These results are shown in Table 7.49.

Strategies	F_1	Average # of rules	Average # of rules with negation	% of rules with negation
		χ^2		
RL + UP	0.894	1076.5	0.0	0.0
RL + UN	0.887	725.2	92.7	12.8
RL + Ov	0.895	968.8	0.0	0.0
RL + UP-UN-Ov	0.893	1076.4	2.0	0.2
RL + UN-UP-Ov	0.886	784.8	96.9	12.3
RL + BestStrategy	0.897	768.5	79.5	10.3
RL + BestPosRule	0.897	793.9	0.0	0.0
RL + BestRule	**0.899**	772.0	101.9	13.2
		IG		
RL + UP	0.799	918.2	0.0	0.0
RL + UN	0.846	616.6	130.7	21.2
RL + Ov	0.847	839.0	0.0	0.0
RL + UP-UN-Ov	0.799	917.9	9.1	1.0
RL + UN-UP-Ov	0.842	668.6	134.6	20.1
RL + BestStrategy	**0.852**	656.4	96.6	14.7
RL + BestPosRule	0.847	692.5	0.0	0.0
RL + BestRule	0.850	659.7	140.7	21.3

Table 7.47: Micro-averaged F_1-measure and the average number of rules for the classification of the Reuters-8 dataset using χ^2 and IG for feature selection and the FP-1 representation

From Table 7.47, it was observed that when χ^2 was used, the best RL strategy was RL + BestRule, which generated rules with negation. The worst RL strategy was RL + UN-UP-Ov. RL + UN had the smallest ruleset while RL + UP had the largest. When IG was used, the best RL strategy was RL + BestStrategy, also a strategy which generated rules with negation. The worst RL strategies were RL + UP and RL + UP-UN-OV. Again, RL + UN had the smallest ruleset while RL + UP had the largest ruleset. The use of χ^2 resulted in higher F_1-measure values and larger rulesets for the RL strategies than when IG was used. However, when IG was used, the percentage of rules with negation generated by the RL strategies was higher than when χ^2 was used.

Table 7.48 shows that the best RL strategies for the mixed representation when χ^2 was used, were RL + UP and RL + UP-UN-OV, although they generated the largest ruleset. Both strategies produced identical F_1-measure values and the same number of rules, indicating that the UP sub-space was never empty in the case of RL + UP-UN-OV. RL + BestRule performed slightly worse and came second, although it was observed that it achieved its results with an average of 234.8 less rules than the best RL

strategies. RL + UN was the worst RL strategy, although it had the smallest ruleset. When IG was used, the best RL strategy was RL + BestRule which generated rules with negation. Again, RL + UN was the worst RL strategy although it had the smallest ruleset. The F_1-measure values generated by the RL strategies were slightly higher when χ^2 was used compared to when IG was used, except for RL + BestStrategy, RL + BestPosRule and RL + BestRule where they were equal. In addition, the average number of rules generated was slightly higher for all the RL strategies with the use of χ^2. However, when IG was used, a higher percentage of rules with negation was generated by the RL strategies.

Strategies	F_1	Average # of rules	Average # of rules with negation	% of rules with negation
χ^2				
RL + UP	**0.959**	535.2	0.0	0.0
RL + UN	0.912	190.3	54.1	28.4
RL + Ov	0.926	330.6	0.0	0.0
RL + UP-UN-Ov	**0.959**	535.2	0.0	0.0
RL + UN-UP-Ov	0.927	258.3	75.4	29.2
RL + BestStrategy	0.954	312.4	55.7	17.8
RL + BestPosRule	0.949	377.7	0.0	0.0
RL + BestRule	0.958	300.4	95.4	31.8
IG				
RL + UP	0.955	493.8	0.0	0.0
RL + UN	0.890	173.2	71.1	41.1
RL + Ov	0.922	315.4	0.0	0.0
RL + UP-UN-Ov	0.955	493.8	0.0	0.0
RL + UN-UP-Ov	0.897	197.3	84.1	42.6
RL + BestStrategy	0.954	269.4	66.5	24.7
RL + BestPosRule	0.949	355.1	0.0	0.0
RL + BestRule	**0.958**	260.1	102.0	39.2

Table 7.48: Micro-averaged F_1-measure and the average number of rules for the classification of the Reuters-8 dataset using χ^2 and IG for feature selection and the mixed representation

Table 7.49 shows the performance of the best RL strategy in comparison with the other machine learning techniques. When χ^2 was used, the best RL strategy was second behind SMO regardless of the text representation used. The worst technique in this case was NB. When IG was used with the FP-1 representation, the best RL strategy was fourth behind OlexGA, SMO and OlexGreedy, only outperforming JRip and NB. The worst strategy was again NB. The use of IG with the mixed representation saw the best RL strategy performing better, coming second behind SMO and outperforming the other techniques. NB was again the worst technique. When comparing only the

rule-based techniques (not including SMO and NB), the best RL strategy was the top performer in all cases except when IG was used with the FP-1 representation.

Techniques	χ^2	Rank	IG	Rank
FP-1				
RL	0.899	2	0.852	4
SMO	0.903	1	0.864	2
NB	0.761	6	0.726	6
JRip	0.841	5	0.818	5
OlexGreedy	0.878	4	0.856	3
OlexGA	0.887	3	0.865	1
Mixed				
RL	0.959	2	0.958	2
SMO	0.979	1	0.976	1
NB	0.881	6	0.827	6
JRip	0.955	3	0.951	3
OlexGreedy	0.943	5	0.949	4
OlexGA	0.947	4	0.948	5

Table 7.49: Micro-averaged F_1-measure for the classification of the Reuters-8 dataset using χ^2 and IG for feature selection for the best RL strategy in comparison with the other machine learning techniques using the FP-1 and mixed representation

Overall, with respect to the classification of the Reuters-8 dataset, the RL strategies that generated rules with negation performed better when the FP-1 representation was used and when the mixed representation was used with IG. When the mixed representation was used, the RL strategies generated a higher percentage of rules with negation compared to when the FP-1 representation was used. The use of χ^2 generally enabled the RL strategies to generate higher F_1-measure values although using IG resulted in the RL strategies generating smaller rulesets. Compared to the other machine learning techniques, the best RL strategy was among the top performers in most cases, only coming second behind SMO while outperforming the other machine learning techniques.

7.3.3 Experiments Using the SAVSNET Dataset

This sub-section discusses the evaluation of the eight strategies in the proposed IRL mechanism using the SAVSNET dataset. Table 7.50 shows the number of fuzzy phrases (FP-1, FP-2, mixed) used for representation of each class in the SAVSNET dataset, selected using χ^2 and IG.

Class name	FP-1		FP-2		Mixed	
	χ^2	IG	χ^2	IG	χ^2	IG
aggression	1	2	1	3	9	12
diarrhoea	76	68	85	74	169	150
pruritus	120	128	129	139	255	269
vomit	44	60	46	62	101	131
total	241	258	261	278	534	562

Table 7.50: The number of fuzzy phrase features used for the representation of each class in the SAVSNET dataset

For the binary classification task, the micro-averaged F_1-measure and the average number of rules generated by the eight strategies in the IRL mechanism using χ^2 and IG for the SAVSNET dataset are shown in Tables 7.51 and 7.52 for the FP-1 and mixed representation respectively. Table 7.53 shows the best RL strategy in comparison with the other machine learning techniques.

Strategies	F_1	Average # of rules	Average # of rules with negation	% of rules with negation
χ^2				
RL + UP	0.784	129.2	0.0	0.0
RL + UN	0.793	123.2	4.1	3.3
RL + Ov	0.791	126.7	0.0	0.0
RL + UP-UN-Ov	0.784	129.2	0.0	0.0
RL + UN-UP-Ov	0.786	128.9	5.0	3.9
RL + BestStrategy	**0.798**	124.3	2.3	1.9
RL + BestPosRule	0.795	124.2	0.0	0.0
RL + BestRule	0.795	124.0	4.1	3.3
IG				
RL + UP	0.770	137.2	0.0	0.0
RL + UN	0.793	125.1	7.7	6.2
RL + Ov	**0.796**	131.5	0.0	0.0
RL + UP-UN-Ov	0.770	137.2	0.9	0.7
RL + UN-UP-Ov	0.767	134.1	8.6	6.4
RL + BestStrategy	0.793	125.9	2.9	2.3
RL + BestPosRule	0.794	126.1	0.0	0.0
RL + BestRule	0.789	125.7	5.9	4.7

Table 7.51: Micro-averaged F_1-measure and the average number of rules for the classification of the SAVSNET dataset using χ^2 and IG for feature selection and the FP-1 representation

Table 7.51 shows that the best RL strategy when χ^2 was used with the FP-1 representation was RL + BestStrategy, which generated rules with negation. The worst RL strategies were RL + UP and RL + UP-UN-OV. These two RL strategies had

identical results, indicating that the UP sub-space was never empty in each round of the rule refinement process. The average number of rules generated by all eight RL strategies was almost the same across these eight strategies, with RL + UP having slightly more than the others, while RL + UN had slightly less. When IG was used, the best RL strategy was RL + Ov while the worst were again RL + UP and RL + UP-UN-Ov. Both the worst RL strategies produced the same F_1-measure values and the same average number of rules, although RL + UP-UN-Ov had a small percentage of rules with negation; suggesting that in some rounds of the rule refinement process, the UP sub-space was empty and therefore, the features from the UN sub-space had to be used. When χ^2 was used, the F_1-measure values were slightly higher than when IG was used, except for RL + UN, which had an equal value and RL + Ov, which had a slightly lower value. The use of χ^2 also enabled the RL strategies to generate slightly smaller rulesets.

Strategies	F_1	Average # of rules	Average # of rules with negation	% of rules with negation
χ^2				
RL + UP	0.944	91.5	0.0	0.0
RL + UN	0.928	64.5	18.7	29.0
RL + Ov	0.933	77.9	0.0	0.0
RL + UP-UN-Ov	0.944	91.5	0.0	0.0
RL + UN-UP-Ov	0.928	65.2	18.7	28.7
RL + BestStrategy	0.944	67.8	12.8	18.9
RL + BestPosRule	0.944	77.9	0.0	0.0
RL + BestRule	**0.945**	68.4	19.3	28.2
IG				
RL + UP	**0.942**	85.9	0.0	0.0
RL + UN	0.811	57.7	16.7	28.9
RL + Ov	0.920	87.1	0.0	0.0
RL + UP-UN-Ov	**0.942**	85.9	0.0	0.0
RL + UN-UP-Ov	0.811	58.8	16.7	28.4
RL + BestStrategy	0.940	69.8	10.0	14.3
RL + BestPosRule	0.938	82.9	0.0	0.0
RL + BestRule	0.941	69.7	19.3	27.7

Table 7.52: Micro-averaged F_1-measure and the average number of rules for the classification of the SAVSNET dataset using χ^2 and IG for feature selection and the mixed representation

When the mixed representation was used with χ^2, the best RL strategy was RL + BestRule, which generated rules with negation, as shown in Table 7.52. RL + UN and RL + UN-UP-Ov were the worst RL strategies. Both RL + UP and RL + UP-UN-OV, which produced the largest rulesets, generated identical results, indicating that

the UP sub-space was never empty in each round of the rule refinement process. RL + UN had the smallest average number of rules. When IG was used, the best RL strategies were RL + UP and RL + UP-UN-Ov. Both of these RL strategies also had the largest ruleset. Identical results obtained by these two RL strategies signified that the UP sub-space was never empty during each round of the rule refinement process. The worst RL strategies were RL + UN and RL + UN-UP-Ov. The use of χ^2 enabled the RL strategies to generate slightly higher F_1-measure values compared to when IG was used.

Table 7.53 shows that when the FP-1 representation was used, regardless of the feature selection technique used, the best RL strategy came second behind OlexGA while JRip was the worst technique. When the mixed representation was used, the best RL strategy only had an average performance, ranking fourth when χ^2 was used and third when IG was used. Comparing only the rule-based techniques (not including SMO and NB), the best RL strategy was second behind OlexGA when χ^2 was used and was ranked first when IG was used. Regardless of the feature selection technique used, the worst technique using the mixed representation was OlexGreedy.

Techniques	χ^2	Rank	IG	Rank
FP-1				
RL	0.798	2	0.796	2
SMO	0.767	4	0.788	4
NB	0.671	5	0.699	5
JRip	0.633	6	0.651	6
OlexGreedy	0.781	3	0.795	3
OlexGA	0.801	1	0.815	1
Mixed				
RL	0.945	4	0.942	3
SMO	0.955	2	0.953	1
NB	0.958	1	0.948	2
JRip	0.921	5	0.913	5
OlexGreedy	0.902	6	0.906	6
OlexGA	0.948	3	0.938	4

Table 7.53: Micro-averaged F_1-measure for the classification of the SAVSNET dataset using χ^2 and IG for feature selection for the best RL strategy in comparison with the other machine learning techniques using the FP-1 and mixed representation

Overall, with respect to the binary classification of the SAVSNET dataset, RL strategies that generated rules with and without negation were each better in different cases. When χ^2 was used, the RL strategies which generated rules with negation performed better, while the RL strategies that did not generate any rules with negation were better when IG was used. It was observed that the use of χ^2 with respect to

both the FP-1 and mixed representation enabled the RL strategies to generate higher F_1-measure values compared to when IG was used. When being compared to the other machine learning techniques, the best RL strategy was among the top performers when the FP-1 representation was used, and performed only averagely when the mixed representation was used.

7.3.4 Summary of Evaluation Using the Fuzzy Phrase Representation

This sub-section summarizes the evaluation of the eight strategies in the proposed IRL mechanism, using both binary and multi-class classification with respect to the 20 Newsgroups, Reuters-8 and SAVSNET datasets using the fuzzy phrase representation. Firstly, the eight RL strategies were compared among one another to identify the best RL strategy. The best RL strategy was then compared to the other machine learning techniques, namely SMO, JRip, NB, OlexGreedy and OlexGA in the binary classification task. The discussion on multi-class classification can be found in Appendix D. In the multi-class classification task, the best RL strategy in the mixed representation was compared with the TFPC algorithm with its four phrase selection strategies. In all the experiments, χ^2 and IG were used as the feature selection techniques for the IRL mechanism. The use of χ^2 generally produced higher F_1-measure values (in binary classification) and higher accuracies (in multi-class classification) compared to the use of IG, regardless of the text representation used. In general, the RL strategies which generated rules with negation (RL + UN, RL + UN-UP-Ov, RL + BestStrategy, RL + BestRule) produced smaller rulesets compared to the strategies which did not generate rules with negation (RL + UP, RL + Ov, RL + UP-UN-Ov, RL + BestPosRule). Although RL + UP-UN-Ov generated a very small percentage of rules with negation ($<$ 1%) in some cases, it should be recalled that this RL strategy was not considered to be a strategy that was expected to generate a significant number of rules with negation.

Tables 7.54, 7.55, 7.56 and 7.57 present a summary of the conducted experiments. The tables indicate whether the RL strategies that generated rules with or without negation were best with respect to each dataset in the context of binary and multi-class classification for both the FP-1 and mixed representations (a tick in a column indicates the best performance). With respect to binary classification using the FP-1 representation, as shown in Table 7.54, the RL strategies that generated rules with negation were found to produce better results in most cases. The RL strategies that generated rules without negation were only better when IG was used in the 20NG-B and SAVSNET datasets. When using the mixed representation, as shown in Table 7.55, the RL strategies that generated rules without negation were better, with only two exceptions: when IG was used in the Reuters-8 dataset and when χ^2 was used in the SAVSNET dataset. In the multi-class classification setting, the RL strategies that generated rules without negation were found to produce better results, regardless of

the feature selection technique or text representation used in all the datasets except for the SAVSNET dataset, as shown in Tables 7.56 and 7.57.

Overall, with respect to the use of fuzzy phrase features in the text representation, it was found that there was a mix of both RL strategies that generated rules with and without negation that performed the best. In the case of binary classification, the RL strategies that generated rules with negation performed better when the FP-1 representation was used than when the mixed representation was used. In the case of multi-class classification, the RL strategies which generated rules without negation performed better.

Datasets	χ^2		IG	
	Without Neg	With Neg	Without Neg	With Neg
20NG-A	√	√	√	√
20NG-B	√	√	√	
Reuters-8		√		√
SAVSNET		√	√	

Table 7.54: Summary of the best RL strategy for binary classification using the FP-1 representation with respect to all the datasets

Datasets	χ^2		IG	
	Without Neg	With Neg	Without Neg	With Neg
20NG-A	√		√	
20NG-B	√		√	
Reuters-8	√			√
SAVSNET		√	√	

Table 7.55: Summary of the best RL strategy for binary classification using the mixed representation with respect to all the datasets

Datasets	χ^2		IG	
	Without Neg	With Neg	Without Neg	With Neg
20NG-A	√		√	
20NG-B	√		√	
Reuters-8	√		√	
SAVSNET		√	√	√

Table 7.56: Summary of the best RL strategy for multi-class classification using the FP-1 representation with respect to all the datasets

Datasets	χ^2		IG	
	Without Neg	With Neg	Without Neg	With Neg
20NG-A	√		√	
20NG-B	√		√	
Reuters-8	√		√	
SAVSNET	√		√	

Table 7.57: Summary of the best RL strategy for multi-class classification using the mixed representation with respect to all the datasets

7.4 Summary

This chapter has presented the experimental setup, results and evaluation of the proposed IRL mechanism using phrases for the text representation. Three different types of phrases were considered: (i) n-gram phrases; (ii) keyphrases and (iii) fuzzy phrases. For each type of phrase used, both binary and multi-class classification were conducted on three text datasets: 20 Newsgroups, Reuters-8 and SAVSNET. The results for the binary classification task are discussed in this chapter while the results for the multi-class classification task are discussed and presented in Appendix B, C and D for the n-gram phrase, keyphrase and fuzzy phrase representations respectively. The next chapter, Chapter 8, which is the concluding chapter of this thesis, summarizes the thesis, presents the main findings and contributions, and discusses some avenues for future research directions.

Chapter 8

Conclusion and Future Directions

This chapter concludes this thesis, commencing with a review of the proposed Inductive Rule Learning (IRL) mechanism and the associated text classification framework in Section 8.1. The main findings and contributions of this thesis are then discussed in Section 8.2. Finally, some suggestions for future research directions are given in Section 8.3.

8.1 Summary

The research work conducted with respect to this thesis has been directed at an investigation into the use of negation in IRL for text classification. The thesis has reported on experimental work that has been carried out using an IRL-based text classification framework to investigate the use of negation, in terms of both keywords and phrases as the text representation, using text datasets (20 Newsgroups (20NG-A and 20NG-B), Reuters-8 and SAVSNET) and five tabular datasets from the UCI Machine Learning Repository. Three main processes were incorporated into the text classification framework: (i) document preprocessing; (ii) the proposed IRL mechanism with negation and (iii) the classification process.

In document preprocessing, a number of sub-processes were considered. These include: data cleaning, keyword extraction, phrase extraction, feature selection and text representation. In data cleaning, the documents in the datasets were stripped off unwanted content. From these documents, keywords and phrases (n-grams, keyphrases, fuzzy phrases) were extracted. Since a substantial number of keywords and phrases were found in the datasets used, feature selection was used to select a subset of the extracted keywords and phrases to be used in the text representation. Two feature selection techniques, Chi-Square (χ^2) and Information Gain (IG), were used in the experiments. The selected keywords and phrases were then used as features, using the bag-of-words and bag-of-phrases as the text representations. A classifier was then learnt using the proposed IRL mechanism with a training set of documents.

The proposed IRL mechanism, which was the focus of this thesis, comprised eight

rule refinement strategies, where strategies that generated rules with and without negation were included. These rule refinement strategies were devised based on the division of the feature space into three sub-spaces containing different types of features according to the documents covered by the "rule so far"; namely Unique Positive (UP), Unique Negative (UN) and Overlap (Ov) feature sub-spaces. UP features are features that are found only in the positive documents covered and not in any negative documents covered. UN features are features that are found only in the negative documents covered and not in any positive documents covered. Ov features are features that are found in at least one positive and one negative document covered. The three different types of features were then used in the rule refinement strategies to generate rules with and without negation. The eight rule refinement strategies were:

1. RL + UP
2. RL + UN
3. RL + Ov
4. RL + UP-UN-Ov
5. RL + UN-UP-Ov
6. RL + BestStrategy
7. RL + BestPosRule
8. RL + BestRule

Strategies 1, 3 and 7 only generated rules without negation while strategies 2, 4, 5, 6 and 8 could generate a mixture of rules with and without negation. A rule with negation would be generated if the UN sub-space was not empty. Therefore, strategy 4 would either generate a very small percentage of rules with negation or no rules at all with negation. Strategy 8 would generate rules with negation if adding a UN feature to the current rule improved its accuracy.

Two text classification tasks, namely binary classification and multi-class classification, were considered within the framework to test the effectiveness of the proposed IRL mechanism. Comparison with the Sequential Minimal Optimization (SMO) algorithm, Naive Bayes (NB), JRip, OlexGreedy and OlexGA was done with respect to the binary classification task; while in the multi-class classification task, comparison was done with the Total From Partial Classification (TFPC) algorithm.

In the experiments using keywords as features for the text representation, two types of datasets were used: text datasets and tabular datasets. Regardless of whether χ^2 or IG was used for feature selection, the results obtained showed that the RL strategies

that generated rules with negation achieved the best results in the binary classification task when classifying the text datasets. However, when classifying the tabular datasets, RL strategies that did not generate rules with negation were generally better. In the case of the multi-class classification task, the RL strategies that did not generate rules with negation achieved the best results more often than the RL strategies which generated rules with negation. Therefore, the use of negation in IRL was more prevalent and effective in binary classification when keywords were used as features for the text representation.

In the experiments using phrases, three types of phrases, namely n-gram phrases, keyphrases and fuzzy phrases, were experimented with. In the context of binary classification, when n-gram phrases were used as the text representation, there was a mix of both RL strategies that generated rules with and without negation achieving the best results. When the 2-gram representation was used, the RL strategies that generated rules without negation achieved the best results more often than the RL strategies which generated rules with negation. This outcome was reversed when the mixed n-gram phrase representation was used. The use of keyphrase features as the text representation also saw a mix of both RL strategies with and without negation achieving the best results. However, for both the KP-2 and mixed representation of keyphrases, the RL strategies that did not generate rules with negation achieved the best results more often than the RL strategies that generated rules with negation. When fuzzy phrase features were used as the text representation, the RL strategies that generated rules with negation achieved the best results more often than the RL strategies that did not generate rules with negation using the FP-1 representation, while the outcome was reversed when the mixed representation of fuzzy phrases was used. In the context of multi-class classification for all the representations used, the RL strategies that did not generate rules with negation achieved the best results in most cases.

Therefore, it could be summarized that the use of negation in the proposed IRL mechanism was more effective when keywords were used as the text representation in binary classification. There was a mix of both RL strategies that generated both rules with and without negation achieving the best results when phrases were used as the text representation. However, the use of negation in the proposed IRL mechanism did not seem advantageous in the context of multi-class classification, regardless of the text representation used. With respect to the feature selection techniques used, the use of χ^2 in general enabled the RL strategies to achieve higher F_1-measure and accuracy values.

In comparison to the other machine learning techniques that were compared, the RL strategies in the proposed IRL mechanism were shown to outperform all the compared techniques in most cases, and were competitive with SMO in the context of the binary classification task. With respect to the multi-class classification task, the performance

of the RL strategies in the proposed IRL mechanism was found to be more effective than the TFPC algorithm.

With respect to the efficiency of the RL strategies used in comparison to the other machine learning techniques, only a sample of the run-times for the IRL process were recorded for the purpose of comparing the efficiency of the RL strategies and that of the other machine learning techniques due to the considerable number of experiments undertaken. The sample taken was for the binary classification of the SAVSNET dataset using χ^2 and the keyword representation. Table 8.1 shows the run-times recorded for the IRL process for the eight strategies in the proposed IRL mechanism and the other machine learning techniques for the SAVSNET dataset using χ^2 and the keyword representation. From Table 8.1, it can be observed that the RL strategy that recorded the fastest time was RL + UN-UP-Ov while RL + BestRule recorded the slowest time among the eight RL strategies. With respect to the other machine learning techniques, NB was the fastest while OlexGA was the slowest. It was also observed that the RL strategies were more efficient than the other machine learning techniques. Recall that the results for the SAVSNET dataset using χ^2 and the keyword representation, presented in Sub-section 6.3.1 in Chapter 6, showed that RL + UN was the best strategy among the eight RL strategies. In comparison with the other machine learning techniques, this strategy came second behind SMO. This showed that the RL strategies were efficient and effective in their use when classifying the SAVSNET dataset using χ^2 and the keyword representation.

Techniques	Time (seconds)
RL + UP	0.29
RL + UN	0.24
RL + Ov	0.27
RL + UP-U-Ov	0.30
RL + UN-UP-Ov	0.21
RL + BestStrategy	0.57
RL + BestPosRule	0.30
RL + BestRule	0.96
SMO	4.83
JRip	16.64
NB	2.36
OlexGreedy	9.23
OlexGA	95.78

Table 8.1: The run-times for the IRL process for the eight strategies in the proposed IRL mechanism and the other machine learning techniques for the SAVSNET dataset using χ^2 and the keyword representation.

8.2 Main Findings and Contributions

This section discusses the main findings of the research work that was carried out and described in this thesis, with respect to the use of the proposed IRL mechanism for text classification. The contributions of this thesis are also summarized. The aim of this thesis was to investigate the use of negation in IRL to establish its effectiveness for the text classification task.

Recall that in Section 1.2 in Chapter 1, two issues were identified with respect to the aim of this thesis. The first issue, directed at the generation of rules with negation, was the identification of the negated feature to be included in a rule. In the proposed IRL mechanism, this was done through the process of feature space division. The feature space, which comprised features from documents covered by an initial rule before refinement, were divided into three sub-spaces according to their occurrences in the documents: the UP, UN and Ov feature sub-spaces. Based on this division, the features grouped under the UN sub-space were considered features which could be advantageously negated. The second issue identified was the strategies to be used to refine a rule. In rule refinement, when a rule can be refined with both positive and negated features, which would be a better option? Refining a rule with a positive feature would mean that the rule generated did not contain negation, while refining a rule with a negated feature would produce a rule which contained negation. This issue was addressed with the implementation of eight rule refinement strategies for incorporation into the proposed IRL mechanism. Each strategy refined a rule differently using either one or more features from the three sub-spaces identified earlier. The following describes how the strategies play a role in determining if adding a positive or negated feature is a better option:

1. RL + UP, RL + UN and RL + Ov each used features from only one sub-space. While RL + UN generated rules with negation, RL + UP and RL + Ov did not. Direct comparison of these three strategies showed whether adding a positive or negated feature was a better option. However, each of these strategies used only features from one same sub-space for rule refinement, therefore it was conjectured that they could be further improved by the inclusion of features from more than one sub-space in the rule refinement process. In addition, these three strategies also suffered from the problem of empty sub-spaces, which led to the rule refinement process stopping prematurely.

2. RL + UP-UN-Ov and RL + UN-UP-Ov were devised to tackle the empty sub-space problem which might occur in RL + UP, RL + UN and RL + Ov. If a sub-space was empty, then the initial rule to be refined with RL + UP, RL + UN or RL + Ov would not be refined, resulting in rule refinement stopping prematurely. Both RL + UP-UN-Ov and RL + UN-UP-Ov were able to generate

rules with negation. RL + UP-UN-Ov generated rules with negation only if the UP sub-space was empty and the UN sub-space was not empty. RL + UN-UP-Ov generated rules with negation only if the UN sub-space was not empty. In the experiments conducted, it was observed that the percentage of rules with negation generated by RL + UN-UP-Ov was higher than that generated by RL + UP-UN-OV, which usually generated either a very small percentage of rules with negation or none at all.

3. For every rule that needed to be refined, RL + BestStrategy selected the best rule generated from RL + UP, RL + UN, RL + Ov, RL + UP-UN-Ov and RL + UN-UP-Ov. Out of these five strategies, only three of the strategies could generate rules with negation. In the rule learning process, for every rule generated, RL + BestStrategy selected the best rule out of the five generated. If the best rule selected was a rule with negation, it signified that, with respect to the learning of the current rule, the use of negation was more advantageous.

4. RL + BestPosRule and RL + BestRule were the most comprehensive strategies, in that they used more than one sub-space in each round of rule refinement. RL + BestPosRule used two sub-spaces (UP and Ov) and did not generate any rules with negation because features from the UN sub-space were not considered to be included. On the other hand, it was possible for RL + BestRule to generate rules with negation, as it included features from all the three sub-spaces, including the UN sub-space. It was noted however, that rules with negation would only be generated if the use of a negated feature resulted in the generation of better rules. Therefore, the fact that the final ruleset produced by RL + BestRule comprised rules with negation, signified that the use of negated feature during rule refinement was beneficial.

For the binary classification task, when keywords were used as the text representation, it was noted that for the classification of all the text datasets (20NG-A, 20NG-B, Reuters-8 and SAVSNET), the use of negation in IRL was effective. However, when phrases were used as the text representation, the effect of using negation in IRL was more varied. The following shows which type of phrase and feature selection techniques benefited the RL strategies which generated rules with negation for the binary classification of each dataset.

For the binary classification of the 20NG-A dataset, the RL strategies which generated rules with negation were best when used with:

- Mixed n-gram representation with χ^2 and IG;
- FP-1 representation with χ^2 and IG.

For the binary classification of the 20NG-B dataset, the RL strategies which generated rules with negation were best when used with:

- 2-gram representation with IG;
- Mixed n-gram representation with IG;
- FP-1 representation with χ^2.

For the binary classification of the Reuters-8 dataset, the RL strategies which generated rules with negation were best when used with:

- 2-gram representation with χ^2;
- Mixed n-gram representation with χ^2 and IG;
- KP-2 representation with IG;
- Mixed keyphrase representation with χ^2 and IG;
- FP-1 representation with χ^2 and IG;
- Mixed fuzzy phrase representation with IG.

For the binary classification of the SAVSNET dataset, the RL strategies which generated rules with negation were best when used with:

- 2-gram representation with IG;
- Mixed n-gram representation with χ^2 and IG;
- Mixed keyphrase representation with χ^2;
- FP-1 representation with χ^2;
- Mixed fuzzy phrase representation with χ^2.

From the above, it was observed that when phrases were used for the text representation, the use of negation in IRL was less effective for the binary classification of both the 20NG-A and 20NG-B datasets and more effective for the Reuters-8 and SAVSNET datasets.

In the case of multi-class classification, a general observation was that the RL strategies that generated rules with negation were not as effective as the RL strategies that generated rules without negation for all the text datasets (20NG-A, 20NG-B, Reuters-8 and SAVSNET), regardless of the text representation or feature selection technique used.

The research contributions of this thesis may therefore be summarized as follows:

1. An approach to dynamically identify features that can be advantageously negated within the context of an IRL process. This approach offers the advantage of not including all possible negations as part of the input feature set. The significance of this is that text datasets often contain many thousands of features and thus negation of all these features is impractical.

2. Eight rule refinement strategies for generating rules both with and without negation as part of the IRL process. These strategies were used to demonstrate that using negation in the IRL process for text classification is beneficial.

3. An IRL mechanism that comprises the dynamic identification of features which can be advantageously negated, and the eight rule refinement strategies, to be used as part of a text classification framework to conduct both the binary and multi-class classification of text datasets.

4. Three different types of phrase representations to support the evaluation of the proposed IRL mechanism. In addition to the use of keywords with the bag-of-words representation, n-gram phrases, keyphrases and fuzzy phrases were used with the bag-of-phrases representation.

5. A comprehensive evaluation of the proposed IRL mechanism, in comparison with the existing machine learning techniques in the context of text classification for both binary and multi-class classification. In the context of binary classification, the proposed IRL mechanism was shown to perform better than the compared machine learning techniques and was competitive with SMO, a support vector machine (SVM) algorithm, considered to be one of the best performing techniques for text classification. In the context of multi-class classification, the proposed IRL mechanism performs distinctly better than the TFPC algorithm.

8.3 Future Directions

With respect to the experiments carried out in this thesis using the proposed IRL mechanism, a number of observations were noted that could be improved on. The application of the proposed IRL mechanism with negation with respect to other application domains might also serve to identify additional benefits.

With regards to the work described in earlier chapters, it was observed that the RL strategies that generated rules with negation achieved the best results less often than the RL strategies that generated rules without negation when phrases (n-gram phrases, keyphrases, fuzzy phrases) were used as the text representation, compared to when keywords were used in the context of binary classification. One of the reasons for this may be that unsuitable phrases were extracted. Therefore, one suggested avenue

for future work is the investigation of alternative techniques for extracting phrases which would benefit the use of negation in IRL.

The proposed IRL mechanism was evaluated using standard benchmark datasets (20 Newsgroups and Reuters-8) and a questionnaire dataset (SAVSNET). In the context of the SAVSNET dataset, the RL strategies that generated rules with negation showed an acceptable performance in that, in all the text representations considered, there is at least one case whereby they achieved the most effective classification results. Consequently, the use of the proposed IRL mechanism may be particularly appropriate to questionnaire-style datasets with respect to applications such as opinion and brand reputation mining. These domains refer to the discovery of whether feedback from end users is positive, negative or neutral with respect to brands, products or services. This can be data from questionnaires about films, blogs, hotels, books and so on. In the context of opinion and brand reputation mining, the notion of IRL with negation may be suited to garnering opinions from text data. It is therefore suggested that it may be of interest to direct future research work of the use of IRL with negation towards opinion and brand reputation mining. Similarly, it might also be useful to investigate the application of IRL with negation with respect to recommender systems. Recommender systems are systems used to predict the preferences of users about a product or service. With respect to IRL with negation, rules such as "People who like X, also like Y but not like Z" could be generated by incorporating the proposed IRL mechanism into the framework for recommender systems.

Appendix A

Stop Words List

a
able
about
above
abroad
according
accordingly
across
actually
adj
after
afterwards
again
against
ago
ahead
ain't
all
allow
allows
almost
alone
along
alongside
already
also
although
always
am
amid
amidst
among
amongst
an
and
another
any
anybody
anyhow
anyone
anything
anyway
anyways
anywhere
apart
appear
appreciate
appropriate
are
aren't
around
as
a's
aside
ask
asking
associated
at
available
away
awfully
b
back
backward
backwards
be
became
because
become
becomes
becoming
been
before
beforehand
begin
behind
being
believe
below
beside
besides
best
better
between
beyond
both
brief
but
by
c
came
can
cannot
cant
can't
caption
cause
causes
certain
certainly
changes
clearly
c'mon
co
co.
com
come
comes
concerning
consequently
consider
considering

contain
containing
contains
corresponding
could
couldn't
course
c's
currently
d
dare
daren't
definitely
described
despite
did
didn't
different
directly
do
does
doesn't
doing
done
don't
down
downwards
during
e
each
edu
eg
eight
eighty
either
else
elsewhere
end
ending
enough
entirely
especially
et
etc
even
ever
evermore
every
everybody
everyone
everything
everywhere
ex
exactly
example
except
f
fairly
far
farther
few
fewer
fifth
first
five
followed
following
follows
for
forever
former
formerly
forth
forward
found
four
from
further
furthermore
g
get
gets
getting
given
gives
go
goes
going
gone
got
gotten
greetings
h
had
hadn't
half
happens
hardly
has
hasn't
have
haven't
having
he
he'd
he'll
hello
help
hence
her
here
hereafter
hereby
herein
here's
hereupon
hers
herself
he's
hi
him
himself
his
hither
hopefully
how
howbeit
however
hundred
i
i'd
ie
if
ignored
i'll
i'm
immediate
in
inasmuch
inc
inc.
indeed
indicate
indicated
indicates
inner
inside
insofar
instead
into
inward
is
isn't
it
it'd
it'll

its
it's
itself
i've
j
just
k
keep
keeps
kept
know
known
knows
l
last
lately
later
latter
latterly
least
less
lest
let
let's
like
liked
likely
likewise
little
look
looking
looks
low
lower
ltd
m
made
mainly
make
makes
many
may
maybe
mayn't
me
mean
meantime
meanwhile
merely
might
mightn't
mine
minus
miss
more
moreover
most
mostly
mr
mrs
much
must
mustn't
my
myself
n
name
namely
nd
near
nearly
necessary
need
needn't
needs
neither
never
neverf
neverless
nevertheless
new
next
nine
ninety
no
nobody
non
none
nonetheless
noone
no-one
nor
normally
not
nothing
notwithstanding
novel
now
nowhere
o
obviously
of
off
often
oh
ok
okay
old
on
once
one
ones
one's
only
onto
opposite
or
other
others
otherwise
ought
oughtn't
our
ours
ourselves
out
outside
over
overall
own
p
particular
particularly
past
per
perhaps
placed
please
plus
possible
presumably
probably
provided
provides
q
que
quite
qv
r
rather
rd
re
really
reasonably
recent
recently

regarding
regardless
regards
relatively
respectively
right
round
s
said
same
saw
say
saying
says
second
secondly
see
seeing
seem
seemed
seeming
seems
seen
self
selves
sensible
sent
serious
seriously
seven
several
shall
shan't
she
she'd
she'll
she's
should
shouldn't
since
six
so
some
somebody
someday
somehow
someone
something
sometime
sometimes
somewhat
somewhere
soon
sorry
specified
specify
specifying
still
sub
such
sup
sure
t
take
taken
taking
tell
tends
th
than
thank
thanks
thanx
that
that'll
thats
that's
that've
the
their
theirs
them
themselves
then
thence
there
thereafter
thereby
there'd
therefore
therein
there'll
there're
theres
there's
thereupon
there've
these
they
they'd
they'll
they're
they've
thing
things
think
third
thirty
this
thorough
thoroughly
those
though
three
through
throughout
thru
thus
till
to
together
too
took
toward
towards
tried
tries
truly
try
trying
t's
twice
two
u
un
under
underneath
undoing
unfortunately
unless
unlike
unlikely
until
unto
up
upon
upwards
us
use
used
useful
uses
using
usually
v
value

various
versus
very
via
viz
vs
w
want
wants
was
wasn't
way
we
we'd
welcome
well
we'll
went
were
we're
weren't
we've
what
whatever
what'll
what's
what've
when
whence
whenever
where
whereafter
whereas
whereby
wherein
where's
whereupon
wherever
whether
which
whichever
while
whilst
whither
who
who'd
whoever
whole
who'll
whom
whomever
who's
whose
why
will
willing
wish
with
within
without
wonder
won't
would
wouldn't
x
y
yes
yet
you
you'd
you'll
your
you're
yours
yourself
yourselves
you've
z
zero

Appendix B

Additional Results and Analysis for Using the N-gram Phrase Representation

B.1 Experiments Using the 20 Newsgroups Dataset

B.1.1 Results and Analysis for Multi-Class Classification

In this sub-section, the results from evaluating the proposed strategies in the IRL mechanism in the context of multi-class classification are presented and discussed. Comparison was made between the proposed strategies and the TFPC algorithm that used its own four phrase selection strategies. Table B.1 shows the average accuracy obtained and the average number of rules generated with respect to the 20NG-A dataset for the eight strategies in the proposed IRL mechanism using χ^2 and IG and the 2-gram representation. Table B.2 shows the same results for the eight strategies in the proposed IRL mechanism but using the mixed representation in comparison to the TFPC algorithm with its four phrase selection strategies. Tables B.3 and B.4 show the same results but for the classification of the 20NG-B dataset.

For the multi-class classification of the 20NG-A dataset, using the 2-gram representation, the results in Table B.1 shows that the best RL strategy regardless of the feature selection technique used, was RL + Ov, which did not generate any rules with negation. The worst RL strategy was RL + UP-UN-Ov. The difference in accuracies between the best and worst RL strategies was 3.1% (when χ^2 was used) and 3.8% (when IG was used). These translated to an additional average of 310.0 and 380.0 (out of 10,000) documents correctly classified by the best RL strategy. The use of χ^2 enabled the RL strategies to produce better average accuracies compared to the use of IG.

Strategies	Average Acc (%)	Average # of rules	Average # of rules with negation	% of rules with negation
χ^2				
RL + UP	77.6	2918.3	0.0	0.0
RL + UN	78.7	2267.1	257.5	11.4
RL + Ov	**80.6**	2517.4	0.0	0.0
RL + UP-UN-Ov	77.5	2922.8	20.5	0.7
RL + UN-UP-Ov	78.2	2381.5	267.7	11.2
RL + BestStrategy	78.3	2276.5	173.9	7.6
RL + BestPosRule	79.3	2293.9	0.0	0.0
RL + BestRule	78.3	2283.7	263.2	11.5
IG				
RL + UP	74.4	3070.4	0.0	0.0
RL + UN	75.0	2300.2	425.3	18.5
RL + Ov	**78.0**	2634.3	0.0	0.0
RL + UP-UN-Ov	74.2	3078.7	19.4	0.6
RL + UN-UP-Ov	74.6	2398.6	437.6	18.2
RL + BestStrategy	75.8	2319.0	244.8	10.6
RL + BestPosRule	77.3	2344.1	0.0	0.0
RL + BestRule	75.7	2324.3	423.3	18.2

Table B.1: Average accuracy and average number of rules for the classification of the 20NG-A dataset using χ^2 and IG for feature selection and the 2-gram representation

Table B.2 shows the same trend, using the mixed representation, whereby the best RL strategy was also found to be RL + Ov while the worst was RL + UP-UN-Ov, regardless of the feature selection technique used. The difference in accuracies between the best and worst RL strategies was 7.2% (when χ^2 was used) and 7.8% (when IG was used); indicating that the best RL strategy correctly classified an additional average of 720.0 and 780.0 (out of 10,000) documents. When χ^2 was used, better average accuracies were achieved by the RL strategies compared to when IG was used, with the exception of RL + UN-UP-Ov and RL + BestRule. RL + Ov was better than the TFPC algorithm. In fact, all the RL strategies were better than the best TFPC phrase selection strategy except for RL + UP and RL + UP-UN-Ov. The best RL strategy was 5.4% higher than the best TFPC phrase selection strategy. This signified that an additional average of 540.0 (out of 10,000) documents were correctly classified by the best RL strategy. It should also be noted that this was achieved using 111.9 fewer rules.

The results for the multi-class classification of the 20NG-B dataset also found RL + Ov to be the best strategy regardless of the feature selection techniques used, using both the 2-gram and mixed representations as shown respectively in Tables B.3 and B.4. The worst RL strategy was found to be RL + UN-UP-Ov when the 2-gram representation was used, regardless of the feature selection technique used, while the identified worst RL strategies in the mixed representation were RL + UN-UP-Ov (when χ^2 was used) and RL + UP and RL + UP-UN-Ov (when IG was used). In the 2-gram representation,

Strategies	Average Acc (%)	Average # of rules	Average # of rules with negation	% of rules with negation
		χ^2		
RL + UP	75.1	1533.7	0.0	0.0
RL + UN	79.5	820.8	293.2	35.7
RL + Ov	**82.1**	1305.6	0.0	0.0
RL + UP-UN-Ov	74.9	1535.8	7.7	0.5
RL + UN-UP-Ov	78.6	867.8	310.3	35.8
RL + BestStrategy	79.2	904.5	209.7	23.2
RL + BestPosRule	80.6	1027.6	0.0	0.0
RL + BestRule	78.7	954.6	350.5	36.7
		IG		
RL + UP	73.2	1400.7	0.0	0.0
RL + UN	78.8	741.3	342.3	46.2
RL + Ov	**80.9**	1172.8	0.0	0.0
RL + UP-UN-Ov	73.1	1403.0	4.8	0.3
RL + UN-UP-Ov	78.8	751.0	342.4	45.6
RL + BestStrategy	79.0	764.5	226.7	29.7
RL + BestPosRule	79.8	938.3	0.0	0.0
RL + BestRule	79.2	789.3	366.7	46.5
TFPC-DelSN_contGO	76.7	1417.5	0.0	0.0
TFPC-DelSN_contGW	75.8	1767.0	0.0	0.0
TFPC-DelSO_contGN	0.0	0.0	0.0	0.0
TFPC-DelSO_contGW	50.2	1022.7	0.0	0.0

Table B.2: Average accuracy and average number of rules for the classification of the 20NG-A dataset using χ^2 and IG for feature selection and the mixed representation

the difference between the best and worst RL strategies was 3.3% (when χ^2 was used) and 3.5% (when IG was used), indicating that the best RL strategy correctly classified an additional average of 329.9 and 349.9 (out of 9,997) documents.

When the mixed representation was used, the best and worst RL strategies differed by 4.9% (when χ^2 was used) and 5.1% (when IG was used); translating to an additional average of 489.9 and 509.8 (out of 9,997) documents correctly classified by the best RL strategy. In comparison to the TFPC algorithm, when χ^2 was used, RL + Ov achieved an accuracy which was 5.4% higher, which translated to an additional average of 539.8 (out of 9,997) documents correctly classified compared to the best TFPC phrase selection strategy but with an average of 264.4 more rules. However, when IG was used, RL + Ov was 4.3% more accurate, which translated to an additional average of 429.9 (out of 9,997) documents correctly classified by RL + Ov, with a ruleset that had on average, 237.4 less rules. All the RL strategies performed better than the TFPC algorithm except RL + UP, RL + UN, RL + UP-UN-Ov and RL + UN-UP-Ov which were very slightly worse when IG was used. It was observed that the use of χ^2 for both representations produced higher average accuracies for the RL strategies compared to the use of IG.

Strategies	Average Acc (%)	Average # of rules	Average # of rules with negation	% of rules with negation
χ^2				
RL + UP	81.4	2751.0	0.0	0.0
RL + UN	80.8	2129.2	258.0	12.1
RL + Ov	**83.4**	2407.8	0.0	0.0
RL + UP-UN-Ov	81.3	2753.9	9.4	0.3
RL + UN-UP-Ov	80.1	2223.3	265.6	11.9
RL + BestStrategy	80.9	2138.1	168.2	7.9
RL + BestPosRule	82.2	2159.9	0.0	0.0
RL + BestRule	81.2	2138.7	265.1	12.4
IG				
RL + UP	78.7	2826.2	0.0	0.0
RL + UN	78.2	2121.8	441.6	20.8
RL + Ov	**81.4**	2460.4	0.0	0.0
RL + UP-UN-Ov	78.7	2828.5	11.5	0.4
RL + UN-UP-Ov	77.9	2183.3	452.2	20.7
RL + BestStrategy	79.2	2131.9	230.3	10.8
RL + BestPosRule	80.4	2161.1	0.0	0.0
RL + BestRule	79.3	2130.6	443.6	20.8

Table B.3: Average accuracy and average number of rules for the classification of the 20NG-B dataset using χ^2 and IG for feature selection and the 2-gram representation

Strategies	Average Acc (%)	Average # of rules	Average # of rules with negation	% of rules with negation
χ^2				
RL + UP	84.5	2155.4	0.0	0.0
RL + UN	81.4	994.8	309.1	31.1
RL + Ov	**86.2**	1740.1	0.0	0.0
RL + UP-UN-Ov	84.5	2157.7	2.1	0.1
RL + UN-UP-Ov	81.3	1001.1	310.0	31.0
RL + BestStrategy	84.3	1246.8	221.3	17.7
RL + BestPosRule	86.0	1427.5	0.0	0.0
RL + BestRule	85.2	1249.1	406.2	32.5
IG				
RL + UP	80.0	1436.6	0.0	0.0
RL + UN	80.4	662.0	294.0	44.4
RL + Ov	**85.1**	1202.3	0.0	0.0
RL + UP-UN-Ov	80.0	1438.0	2.4	0.2
RL + UN-UP-Ov	80.4	663.4	294.1	44.3
RL + BestStrategy	83.1	815.3	224.2	27.5
RL + BestPosRule	84.4	979.2	0.0	0.0
RL + BestRule	83.6	805.1	368.6	45.8
TFPC-DelSN_contGO	80.8	1439.7	0.0	0.0
TFPC-DelSN_contGW	78.2	1875.6	0.0	0.0
TFPC-DelSO_contGN	0.0	0.0	0.0	0.0
TFPC-DelSO_contGW	45.2	1035.3	0.0	0.0

Table B.4: Average accuracy and average number of rules for the classification of the 20NG-B dataset using χ^2 and IG for feature selection and the mixed representation

Overall, the strategy that did not generate rules with negation, RL + Ov, was the best strategy in the multi-class classification setting for the 20 Newsgroups dataset. In general, the RL strategies also performed better than the TFPC algorithm.

B.1.2 Results for Using the 3-gram Representation

Strategies	F_1	Average # of rules	Average # of rules with negation	% of rules with negation
		χ^2		
RL + UP	0.786	2595.7	0.0	0.0
RL + UN	0.793	2305.6	45.7	2.0
RL + Ov	**0.796**	2359.7	0.0	0.0
RL + UP-UN-Ov	0.783	2597.7	12.1	0.5
RL + UN-UP-Ov	0.788	2447.5	56.9	2.3
RL + BestStrategy	0.791	2310.3	30.9	1.3
RL + BestPosRule	0.794	2322.7	0.0	0.0
RL + BestRule	0.789	2321.3	48.7	2.1
		IG		
RL + UP	0.753	2953.1	0.0	0.0
RL + UN	0.758	2584.5	79.4	3.1
RL + Ov	**0.763**	2656.5	0.0	0.0
RL + UP-UN-Ov	0.750	2956.3	14.5	0.5
RL + UN-UP-Ov	0.757	2738.6	90.6	3.3
RL + BestStrategy	0.757	2593.4	50.6	2.0
RL + BestPosRule	**0.763**	2616.4	0.0	0.0
RL + BestRule	0.754	2613.7	75.5	2.9

Table B.5: Micro-averaged F_1-measure and the average number of rules for the classification of the 20NG-A dataset using χ^2 and IG for feature selection and the 3-gram representation

Techniques	χ^2	Rank	IG	Rank
	3-gram			
RL	0.796	1	0.763	1
SMO	0.759	2	0.751	2
NB	0.480	6	0.449	6
JRip	0.612	3	0.563	5
OlexGreedy	0.580	5	0.579	4
OlexGA	0.583	4	0.583	3

Table B.6: Micro-averaged F_1-measure for the classification of the 20NG-A dataset using χ^2 and IG for feature selection for the best RL strategy in comparison with the other machine learning techniques using the 3-gram representation

Strategies	F_1	Average # of rules	Average # of rules with negation	% of rules with negation
χ^2				
RL + UP	0.827	2595.7	0.0	0.0
RL + UN	0.830	2309.2	57.9	2.5
RL + Ov	**0.831**	2363.1	0.0	0.0
RL + UP-UN-Ov	0.826	2596.9	11.1	0.4
RL + UN-UP-Ov	0.828	2449.0	76.5	3.1
RL + BestStrategy	0.829	2313.3	39.8	1.7
RL + BestPosRule	0.829	2320.6	0.0	0.0
RL + BestRule	0.830	2312.4	53.7	2.3
IG				
RL + UP	0.789	3028.5	0.0	0.0
RL + UN	0.790	2674.3	154.2	5.8
RL + Ov	0.790	2736.0	0.0	0.0
RL + UP-UN-Ov	0.788	3029.7	20.6	0.7
RL + UN-UP-Ov	0.791	2820.5	163.0	5.8
RL + BestStrategy	0.793	2681.0	81.2	3.0
RL + BestPosRule	**0.794**	2688.1	0.0	0.0
RL + BestRule	0.793	2680.5	145.2	5.4

Table B.7: Micro-averaged F_1-measure and the average number of rules for the classification of the 20NG-B dataset using χ^2 and IG for feature selection and the 3-gram representation

Techniques	χ^2	Rank	IG	Rank
3-gram				
RL	0.831	1	0.794	1
SMO	0.800	2	0.790	2
NB	0.540	6	0.509	6
JRip	0.694	3	0.666	3
OlexGreedy	0.619	5	0.620	5
OlexGA	0.622	4	0.624	4

Table B.8: Micro-averaged F_1-measure for the classification of the 20NG-B dataset using χ^2 and IG for feature selection for the best RL strategy in comparison with the other machine learning techniques using the 3-gram representation

Strategies	Average Acc (%)	Average # of rules	Average # of rules with negation	% of rules with negation
χ^2				
RL + UP	71.5	2595.7	0.0	0.0
RL + UN	72.4	2305.6	45.7	2.0
RL + Ov	**72.7**	2359.7	0.0	0.0
RL + UP-UN-Ov	71.3	2597.7	12.1	0.5
RL + UN-UP-Ov	71.8	2447.5	56.9	2.3
RL + BestStrategy	72.2	2310.3	30.9	1.3
RL + BestPosRule	72.4	2322.7	0.0	0.0
RL + BestRule	71.9	2321.3	48.7	2.1
IG				
RL + UP	68.0	2953.1	0.0	0.0
RL + UN	68.7	2584.5	79.4	3.1
RL + Ov	68.9	2656.5	0.0	0.0
RL + UP-UN-Ov	67.9	2956.3	14.5	0.5
RL + UN-UP-Ov	68.5	2738.6	90.6	3.3
RL + BestStrategy	68.5	2593.4	50.6	2.0
RL + BestPosRule	**69.0**	2616.4	0.0	0.0
RL + BestRule	68.4	2613.7	75.5	2.9

Table B.9: Average accuracy and the average number of rules for the classification of the 20NG-A dataset using χ^2 and IG for feature selection and the 3-gram representation

Strategies	Average Acc (%)	Average # of rules	Average # of rules with negation	% of rules with negation
χ^2				
RL + UP	73.4	2595.7	0.0	0.0
RL + UN	73.6	2309.2	57.9	2.5
RL + Ov	**74.0**	2363.1	0.0	0.0
RL + UP-UN-Ov	73.3	2596.9	11.1	0.4
RL + UN-UP-Ov	73.3	2449.0	76.5	3.1
RL + BestStrategy	73.4	2313.3	39.8	1.7
RL + BestPosRule	73.5	2320.6	0.0	0.0
RL + BestRule	73.5	2312.4	53.7	2.3
IG				
RL + UP	69.1	3028.5	0.0	0.0
RL + UN	69.2	2674.3	154.2	5.8
RL + Ov	**69.6**	2736.0	0.0	0.0
RL + UP-UN-Ov	69.1	3029.7	20.6	0.7
RL + UN-UP-Ov	69.2	2820.5	163.0	5.8
RL + BestStrategy	69.2	2681.0	81.2	3.0
RL + BestPosRule	**69.6**	2688.1	0.0	0.0
RL + BestRule	69.4	2680.5	145.2	5.4

Table B.10: Average accuracy and the average number of rules for the classification of the 20NG-B dataset using χ^2 and IG for feature selection and the 3-gram representation

B.2 Experiments Using the Reuters-8 Dataset

B.2.1 Results and Analysis for Multi-Class Classification

This sub-section discusses the evaluation results produced using the eight strategies in the proposed IRL mechanism for the multi-class classification of the Reuters-8 dataset. The RL strategies were compared to the TFPC algorithm with its four phrase selection strategies. Table B.11 shows the average accuracy obtained and the average number of rules generated with respect to the Reuters-8 dataset for the eight strategies in the proposed IRL mechanism using χ^2 and IG, and the 2-gram representation. Table B.12 shows the same results for the eight strategies in the proposed IRL mechanism but using the mixed representation in comparison to the TFPC algorithm with its four phrase selection strategies.

Strategies	Average Acc (%)	Average # of rules	Average # of rules with negation	% of rules with negation
χ^2				
RL + UP	85.2	1532.8	0.0	0.0
RL + UN	86.6	742.5	297.5	40.1
RL + Ov	**89.2**	1193.3	0.0	0.0
RL + UP-UN-Ov	85.2	1532.8	2.6	0.2
RL + UN-UP-Ov	86.4	754.6	297.7	39.5
RL + BestStrategy	86.0	785.3	189.5	24.1
RL + BestPosRule	88.1	869.9	0.0	0.0
RL + BestRule	87.1	819.0	336.2	41.1
IG				
RL + UP	77.1	1843.0	0.0	0.0
RL + UN	83.5	796.2	321.4	40.4
RL + Ov	**86.7**	1372.3	0.0	0.0
RL + UP-UN-Ov	77.1	1843.0	3.5	0.2
RL + UN-UP-Ov	83.1	817.9	324.3	39.7
RL + BestStrategy	83.6	855.7	168.6	19.7
RL + BestPosRule	85.4	934.1	0.0	0.0
RL + BestRule	84.2	859.8	358.3	41.7

Table B.11: Average accuracy and average number of rules for the classification of the Reuters-8 dataset using χ^2 and IG for feature selection and the 2-gram representation

In the evaluation using the 2-gram representation, the best RL strategy was RL + Ov, which generated rules without negation, regardless of the feature selection techniques used, as shown in Table B.11. The worst RL strategies were RL + UP and RL + UP-UN-Ov. Their difference in accuracies were recorded as 4.0% (when χ^2 was used) and 9.6% (when IG was used), translating to an additional average of 265.7 and 637.7 (out of 6,643) documents correctly classified by RL + Ov.

However, when using the mixed representation, the best RL strategy was RL + BestRule, which generated rules with negation regardless of the feature selection techniques used, as shown in Table B.12. The worst RL strategies were RL + UN and RL + UN-UP-Ov (when χ^2 was used), and RL + UP and RL + UP-UN-Ov (when IG was used). The difference between the best and worst RL strategies was 4.5% (when χ^2 was used) and 6.8% (when IG was used), indicating that an additional average of 298.9 and 451.7 (out of 6,643) documents were correctly classified by RL + BestRule.

Strategies	Average Acc (%)	Average # of rules	Average # of rules with negation	% of rules with negation
		χ^2		
RL + UP	86.6	818.1	0.0	0.0
RL + UN	85.5	208.0	114.0	54.8
RL + Ov	88.3	371.1	0.0	0.0
RL + UP-UN-Ov	86.6	818.1	1.4	0.2
RL + UN-UP-Ov	85.5	266.6	144.4	54.2
RL + BestStrategy	89.0	335.0	91.5	27.3
RL + BestPosRule	89.4	416.5	0.0	0.0
RL + BestRule	**90.0**	345.0	191.7	55.6
		IG		
RL + UP	82.6	654.3	0.0	0.0
RL + UN	84.0	162.0	101.6	62.7
RL + Ov	87.6	306.0	0.0	0.0
RL + UP-UN-Ov	82.6	654.3	0.9	0.1
RL + UN-UP-Ov	84.1	172.4	107.6	62.4
RL + BestStrategy	88.3	246.5	92.1	37.4
RL + BestPosRule	88.7	333.1	0.0	0.0
RL + BestRule	**89.4**	251.7	165.3	65.7
TFPC-DelSN_contGO	76.7	1084.8	0.0	0.0
TFPC-DelSN_contGW	74.8	1399.7	0.0	0.0
TFPC-DelSO_contGN	0.0	0.0	0.0	0.0
TFPC-DelSO_contGW	72.6	996.0	0.0	0.0

Table B.12: Average accuracy and average number of rules for the classification of the Reuters-8 dataset using χ^2 and IG for feature selection and the mixed representation

It was observed that the average accuracies of the RL strategies were higher when χ^2 was used compared to when IG was used, for both the 2-gram and mixed representations. The best RL strategy outperformed the best TFPC phrase selection strategy by a substantial average margin of 13.3%, which translated to an additional average of 883.5 (out of 6,643) documents correctly classified by the best RL strategy. This was also achieved with an average of 739.8 fewer rules than the best TFPC phrase selection strategy. In fact, all the RL strategies, regardless of the feature selection technique used, performed better than the TFPC algorithm.

Overall, for the multi-class classification of the Reuters-8 dataset, RL + Ov, which did not generate rules with negation was the best RL strategy when using the 2-gram representation. A similar observation was made with respect to the 20 Newsgroups datasets. However, when the mixed representation was used, the best RL strategy was RL + BestRule, which generated rules with negation. Compared to the TFPC algorithm, all the RL strategies were undoubtably better.

B.2.2 Results for Using the 3-gram Representation

Strategies	F_1	Average # of rules	Average # of rules with negation	% of rules with negation
		χ^2		
RL + UP	0.851	2027.9	0.0	0.0
RL + UN	0.862	1453.2	225.0	15.5
RL + Ov	**0.867**	1696.8	0.0	0.0
RL + UP-UN-Ov	0.851	2027.8	1.9	0.1
RL + UN-UP-Ov	0.857	1518.8	237.1	15.6
RL + BestStrategy	0.861	1461.7	159.9	10.9
RL + BestPosRule	0.859	1485.3	0.0	0.0
RL + BestRule	0.863	1461.7	241.5	16.5
		IG		
RL + UP	0.691	2828.8	0.0	0.0
RL + UN	**0.768**	2008.5	356.7	17.8
RL + Ov	0.765	2392.7	0.0	0.0
RL + UP-UN-Ov	0.691	2828.9	6.2	0.2
RL + UN-UP-Ov	0.758	2083.8	368.1	17.7
RL + BestStrategy	0.761	2053.6	149.5	7.3
RL + BestPosRule	0.763	2096.5	0.0	0.0
RL + BestRule	0.763	2056.9	390.1	19.0

Table B.13: Micro-averaged F_1-measure and the average number of rules for the classification of the Reuters-8 dataset using χ^2 and IG for feature selection and the 3-gram representation

Techniques	χ^2	Rank	IG	Rank
	3-gram			
RL	0.867	1	0.768	2
SMO	0.840	2	0.752	4
NB	0.698	6	0.647	6
JRip	0.735	5	0.719	5
OlexGreedy	0.787	4	0.765	3
OlexGA	0.795	3	0.772	1

Table B.14: Micro-averaged F_1-measure for the classification of the Reuters-8 dataset using χ^2 and IG for feature selection for the best RL strategy in comparison with the other machine learning techniques using the 3-gram representation

Strategies	Average Acc (%)	Average # of rules	Average # of rules with negation	% of rules with negation
χ^2				
RL + UP	79.6	2027.9	0.0	0.0
RL + UN	81.9	1453.2	225.0	15.5
RL + Ov	**83.7**	1696.8	0.0	0.0
RL + UP-UN-Ov	79.6	2027.8	1.9	0.0
RL + UN-UP-Ov	81.2	1518.8	237.1	15.6
RL + BestStrategy	81.0	1461.7	159.9	10.9
RL + BestPosRule	81.8	1485.3	0.0	0.0
RL + BestRule	81.4	1461.7	241.5	16.5
IG				
RL + UP	65.4	2828.8	0.0	0.0
RL + UN	71.5	2008.5	356.7	17.8
RL + Ov	**72.5**	2392.7	0.0	0.0
RL + UP-UN-Ov	65.4	2828.9	6.2	0.2
RL + UN-UP-Ov	70.1	2083.8	368.1	17.7
RL + BestStrategy	70.7	2053.6	149.5	7.3
RL + BestPosRule	71.4	2096.5	0.0	0.0
RL + BestRule	70.6	2056.9	390.1	19.0

Table B.15: Average accuracy and the average number of rules for the classification of the Reuters-8 dataset using χ^2 and IG for feature selection and the 3-gram representation

B.3 Experiments Using the SAVSNET Dataset

B.3.1 Results and Analysis for Multi-Class Classification

The evaluation of the eight strategies in the proposed IRL mechanism for the multi-class classification of the SAVSNET dataset is discussed in this sub-section. Table B.16 shows the average accuracy obtained and the average number of rules generated with respect to the SAVSNET dataset for the eight strategies in the proposed IRL mechanism using χ^2 and IG, and the 2-gram representation. Table B.17 shows the same data for the eight strategies in the proposed IRL mechanism using the mixed representation in comparison to the TFPC algorithm with its four phrase selection strategies.

Strategies	Average Acc (%)	Average # of rules	Average # of rules with negation	% of rules with negation
χ^2				
RL + UP	77.3	334.1	0.0	0.0
RL + UN	79.8	232.4	62.7	27.0
RL + Ov	**81.4**	279.7	0.0	0.0
RL + UP-UN-Ov	77.3	334.1	0.8	0.2
RL + UN-UP-Ov	79.2	237.6	62.9	26.5
RL + BestStrategy	77.7	243.0	34.2	14.1
RL + BestPosRule	79.2	258.0	0.0	0.0
RL + BestRule	78.1	243.1	64.1	26.4
IG				
RL + UP	72.5	344.6	0.0	0.0
RL + UN	77.8	214.7	68.4	31.9
RL + Ov	**78.5**	264.6	0.0	0.0
RL + UP-UN-Ov	72.5	344.6	0.8	0.2
RL + UN-UP-Ov	77.7	216.5	68.4	31.6
RL + BestStrategy	77.4	221.4	40.4	18.2
RL + BestPosRule	77.2	251.1	0.0	0.0
RL + BestRule	78.3	221.4	72.9	32.9

Table B.16: Average accuracy and average number of rules for the classification of the SAVSNET dataset using χ^2 and IG for feature selection and the 2-gram representation

The evaluation of the multi-class classification of the SAVSNET dataset produced consistent results with respect to the best and worst identified RL strategies regardless of the feature selection techniques or the representation used. In each case, the best strategy was RL + Ov, which did not generate rules with negation, and the worst strategies were RL + UP and RL + UP-UN-Ov. When the 2-gram representation was used, the best and worst RL strategies differed by 4.1% (when χ^2 was used) and 6.0% (when IG was used), translating to an additional average of 33.9 and 49.7 (out of 828) documents correctly classified by RL + Ov. In the case of the mixed representation,

the difference in accuracies between the best and worst RL strategies was 6.6% (when χ^2 was used) and 6.2% (when IG was used), indicating an additional average of 54.6 and 51.3 (out of 828) documents correctly classified by RL + Ov.

Strategies	Average Acc (%)	Average # of rules	Average # of rules with negation	% of rules with negation
		χ^2		
RL + UP	76.3	159.9	0.0	0.0
RL + UN	81.5	84.5	40.0	47.3
RL + Ov	**82.7**	108.9	0.0	0.0
RL + UP-UN-Ov	76.1	159.9	0.1	0.1
RL + UN-UP-Ov	80.8	91.6	42.9	46.8
RL + BestStrategy	80.1	97.0	31.7	32.7
RL + BestPosRule	80.1	119.0	0.0	0.0
RL + BestRule	79.4	96.8	48.9	50.5
		IG		
RL + UP	75.6	145.5	0.0	0.0
RL + UN	80.9	78.1	43.6	55.8
RL + Ov	**81.8**	97.8	0.0	0.0
RL + UP-UN-Ov	75.7	145.5	1.9	1.3
RL + UN-UP-Ov	80.1	80.5	43.8	54.4
RL + BestStrategy	79.7	83.4	34.8	41.7
RL + BestPosRule	77.9	106.1	0.0	0.0
RL + BestRule	79.2	83.1	50.3	60.5
TFPC-DelSN_contGO	38.4	1614.0	0.0	0.0
TFPC-DelSN_contGW	41.7	1541.1	0.0	0.0
TFPC-DelSO_contGN	39.5	1742.2	0.0	0.0
TFPC-DelSO_contGW	43.0	1587.6	0.0	0.0

Table B.17: Average accuracy and average number of rules for the classification of the SAVSNET dataset using χ^2 and IG for feature selection and the mixed representation

When χ^2 was used, the average accuracy for the RL strategies were higher than when IG was used, with the exception of RL + BestRule when using the 2-gram representation. In comparison with the TFPC algorithm, the results of the RL strategies far surpassed that of the TFPC algorithm. The best TFPC phrase selection strategy only managed an average accuracy of 43.0%, as opposed to the best RL strategy, which recorded an average accuracy of 83.7%. This translated to an additional average of 337.0 (out of 828) documents being correctly classified using the best RL strategy. The poor performance of the TFPC algorithm could be attributed to the content of the SAVSNET dataset, which featured poor grammar and punctuation.

Overall, with respect to the multi-class classification of the SAVSNET dataset, the best RL strategy was RL + Ov, which did not generate rules with negation. When com-

pared to the results for the TFPC algorithm, the results achieved by the RL strategies were undoubtably much better.

B.3.2 Results for Using the 3-gram Representation

Strategies	F_1	Average # of rules	Average # of rules with negation	% of rules with negation
χ^2				
RL + UP	0.729	386.8	0.0	0.0
RL + UN	0.778	281.5	40.7	14.5
RL + Ov	**0.787**	320.3	0.0	0.0
RL + UP-UN-Ov	0.730	386.8	1.8	0.5
RL + UN-UP-Ov	0.768	297.3	45.0	15.1
RL + BestStrategy	0.769	290.1	25.6	8.8
RL + BestPosRule	0.766	299.0	0.0	0.0
RL + BestRule	0.779	288.0	38.9	13.5
IG				
RL + UP	0.642	429.9	0.0	0.0
RL + UN	**0.732**	299.9	63.4	21.1
RL + Ov	0.731	347.5	0.0	0.0
RL + UP-UN-Ov	0.642	429.9	4.5	1.0
RL + UN-UP-Ov	0.729	314.1	65.3	20.8
RL + BestStrategy	0.719	309.2	39.3	12.7
RL + BestPosRule	0.722	321.3	0.0	0.0
RL + BestRule	0.729	308.7	64.8	21.0

Table B.18: Micro-averaged F_1-measure and the average number of rules for the classification of the SAVSNET dataset using χ^2 and IG for feature selection and the 3-gram representation

Techniques	χ^2	Rank	IG	Rank
3-gram				
RL	0.787	1	0.732	1
SMO	0.694	3	0.645	4
NB	0.649	5	0.618	5
JRip	0.604	6	0.610	6
OlexGreedy	0.682	4	0.664	3
OlexGA	0.700	2	0.668	2

Table B.19: Micro-averaged F_1-measure for the classification of the SAVSNET dataset using χ^2 and IG for feature selection for the best RL strategy in comparison with the other machine learning techniques using the 3-gram representation

Strategies	Average Acc (%)	Average # of rules	Average # of rules with negation	% of rules with negation
χ^2				
RL + UP	71.6	386.8	0.0	0.0
RL + UN	76.6	281.5	40.7	14.5
RL + Ov	**78.5**	320.3	0.0	0.0
RL + UP-UN-Ov	71.5	386.8	1.8	0.5
RL + UN-UP-Ov	75.5	297.3	45.0	15.1
RL + BestStrategy	75.0	290.1	25.6	8.8
RL + BestPosRule	74.8	299.0	0.0	0.0
RL + BestRule	76.3	288.0	38.9	13.5
IG				
RL + UP	63.6	429.9	0.0	0.0
RL + UN	71.6	299.9	63.4	21.1
RL + Ov	**71.8**	347.5	0.0	0.0
RL + UP-UN-Ov	63.7	429.9	4.5	1.0
RL + UN-UP-Ov	71.4	314.1	65.3	20.8
RL + BestStrategy	71.1	309.2	39.3	12.7
RL + BestPosRule	71.0	321.3	0.0	0.0
RL + BestRule	71.5	308.7	64.8	21.0

Table B.20: Average accuracy and the average number of rules for the classification of the SAVSNET dataset using χ^2 and IG for feature selection and the 3-grams representation

Appendix C

Additional Results and Analysis for Using the Keyphrase Representation

C.1 Experiments Using the 20 Newsgroups Dataset

C.1.1 Results and Analysis for Multi-Class Classification

This sub-section discusses the evaluation of the eight strategies in the proposed IRL mechanism with respect to the multi-class classification task in comparison with the TFPC algorithm with its four phrase selection strategies. Table C.1 shows the average accuracy obtained and the average number of rules generated with respect to the 20NG-A dataset for the eight strategies in the proposed IRL mechanism using χ^2 and IG, and the KP-2 representation. Table C.2 shows the same results but using the mixed representation in comparison to the TFPC algorithm, which had four phrase selection strategies. Tables C.3 and C.4 show the same results but for the classification of the 20NG-B dataset.

In Table C.1, the best RL strategies when χ^2 was used with the KP-2 representation were RL + UP and RL + UP-UN-Ov. RL + Ov and RL + BestPosRule was close behind, achieving the results with much smaller rulesets. When IG was used, the best RL strategy was RL + BestPosRule. While RL + UP-UN-Ov generated a very small percentage of rules with negation, the other strategies mentioned did not generate any rules with negation. The worst RL strategy was RL + UN-UP-Ov and RL + UN. The difference in accuracy between the best and worst strategies was 2.8% (when χ^2 was used) and 3.9% (when IG was used), translating into an additional average of 280.0 and 390.0 (out of 10,000) documents being correctly classified by the best RL strategy. In general, the use of χ^2 resulted in the RL strategies generating higher average accuracies as compared to the use of IG.

Strategies	Average Acc (%)	Average # of rules	Average # of rules with negation	% of rules with negation
χ^2				
RL + UP	**87.8**	2000.5	0.0	0.0
RL + UN	85.1	1626.2	162.9	10.0
RL + Ov	87.7	1870.3	0.0	0.0
RL + UP-UN-Ov	**87.8**	2000.5	7.9	0.4
RL + UN-UP-Ov	85.0	1708.4	172.0	10.1
RL + BestStrategy	86.8	1677.3	111.3	6.6
RL + BestPosRule	87.7	1677.9	0.0	0.0
RL + BestRule	86.8	1673.1	157.1	9.4
IG				
RL + UP	83.0	2233.0	0.0	0.0
RL + UN	79.9	1631.0	444.6	27.3
RL + Ov	82.9	2101.2	0.0	0.0
RL + UP-UN-Ov	82.9	2233.0	6.1	0.3
RL + UN-UP-Ov	79.9	1636.1	445.8	27.2
RL + BestStrategy	82.0	1686.9	246.3	14.6
RL + BestPosRule	**83.8**	1756.1	0.0	0.0
RL + BestRule	82.1	1692.3	429.8	25.4

Table C.1: Average accuracy and the average number of rules for the classification of the 20NG-A dataset using χ^2 and IG for feature selection and the KP-2 representation

Using the mixed representation for the 20NG-A dataset, the best RL strategy was RL + BestPosRule, regardless of the feature selection technique used, as shown in Table C.2. The worst RL strategy was RL + UN (when χ^2 was used) and RL + UN-UP-Ov (when IG was used). The difference in accuracies between the best and worst RL strategies was 9.2% (when χ^2 was used) and 10.1% (when IG was used), indicating that an additional average of 920.0 and 1,010.0 (out of 10,000) documents were correctly classified by the best RL strategy. Again, the use of χ^2 enabled the RL strategies to obtain better average accuracies compared to the use of IG. In comparison with the TFPC algorithm, the best RL strategy was 8.0% more accurate than the best TFPC phrase selection strategy. This translated to an additional average of 800.0 (out of 10,000) documents being correctly classified by the best RL strategy. In fact, all the RL strategies, except for RL + UN (when χ^2 was used) and RL + UN and RL + UN-UP-Ov (when IG was used), were better than the best TFPC phrase selection strategy.

Table C.3 shows the classification results when using the KP-2 representation for the 20NG-B dataset. The best strategies were RL + UP and RL + UP-UN-Ov (when χ^2 was used) and RL + Ov (when IG was used). RL + UP-UN-Ov generated a very small percentage of rules with negation, while the other two strategies did not generate any rules with negation. The worst strategy was RL + UN (when χ^2 was used) and RL + UN-UP-Ov (when IG was used). The difference in accuracies between the best

and worst strategies was 2.3% (when χ^2 was used) and 2.2% (when IG was used), translating to an additional average of 229.9 and 219.9 (out of 9,997) documents being correctly classified by the best RL strategies. Again, the use of χ^2 resulted in the RL strategies achieving higher average accuracies when classifying the 20NG-B dataset, as compared to when IG was used.

Strategies	Average Acc (%)	Average # of rules	Average # of rules with negation	% of rules with negation
		χ^2		
RL + UP	82.1	825.9	0.0	0.0
RL + UN	75.5	536.4	156.8	29.2
RL + Ov	82.7	866.6	0.0	0.0
RL + UP-UN-Ov	82.1	825.9	0.1	0.0
RL + UN-UP-Ov	77.6	624.7	173.6	27.8
RL + BestStrategy	82.2	645.3	101.9	15.8
RL + BestPosRule	**84.7**	673.7	0.0	0.0
RL + BestRule	82.5	650.1	127.9	19.7
		IG		
RL + UP	81.1	900.9	0.0	0.0
RL + UN	74.1	573.7	171.0	29.8
RL + Ov	83.4	921.5	0.0	0.0
RL + UP-UN-Ov	81.1	900.9	1.4	0.2
RL + UN-UP-Ov	73.8	621.4	187.4	30.2
RL + BestStrategy	81.6	663.7	122.2	18.4
RL + BestPosRule	**83.9**	701.3	0.0	0.0
RL + BestRule	82.0	655.4	154.6	23.6
TFPC-DelSN_contGO	76.7	1417.5	0.0	0.0
TFPC-DelSN_contGW	75.8	1767.0	0.0	0.0
TFPC-DelSO_contGN	0.0	0.0	0.0	0.0
TFPC-DelSO_contGW	50.2	1022.7	0.0	0.0

Table C.2: Average accuracy and the average number of rules for the classification of the 20NG-A dataset using χ^2 and IG for feature selection and the mixed representation

Table C.4 shows that when using the mixed representation, regardless of the feature selection technique used, the best RL strategy for classifying the 20NG-B dataset was RL + BestPosRule, which did not generate rules with negation. The worst RL strategy was RL + UN. The difference in accuracies between the best and worst RL strategies was 9.6% (when χ^2 was used) and 8.6% (when IG was used), indicating that an additional average of 959.7 and 859.7 (out of 9,997) documents were correctly classified by the best RL strategy. Similar to the case of the 20NG-A dataset, the use of χ^2 here also enabled the RL strategies to obtain better average accuracies than when IG was used. Compared to the best TFPC phrase selection strategy, the best RL strategy was

8.1% more accurate. This translated to an additional average of 809.8 (out of 9,997) documents being correctly classified by the best RL strategy.

Strategies	Average Acc (%)	Average # of rules	Average # of rules with negation	% of rules with negation
χ^2				
RL + UP	**89.7**	1908.8	0.0	0.0
RL + UN	87.4	1544.8	158.1	10.2
RL + Ov	89.5	1803.6	0.0	0.0
RL + UP-UN-Ov	**89.7**	1908.8	4.3	0.2
RL + UN-UP-Ov	87.6	1642.1	176.6	10.8
RL + BestStrategy	88.1	1565.5	107.8	6.9
RL + BestPosRule	89.1	1589.9	0.0	0.0
RL + BestRule	88.0	1565.9	153.9	9.8
IG				
RL + UP	85.6	2159.2	0.0	0.0
RL + UN	84.3	1550.8	412.3	26.6
RL + Ov	**86.4**	2042.7	0.0	0.0
RL + UP-UN-Ov	85.6	2159.1	7.0	0.3
RL + UN-UP-Ov	84.2	1559.0	414.0	26.6
RL + BestStrategy	84.4	1567.2	238.7	15.2
RL + BestPosRule	85.8	1604.8	0.0	0.0
RL + BestRule	84.7	1567.4	408.9	26.1

Table C.3: Average accuracy and the average number of rules for the classification of the 20NG-B dataset using χ^2 and IG for feature selection and the KP-2 representation

Strategies	Average Acc (%)	Average # of rules	Average # of rules with negation	% of rules with negation
		χ^2		
RL + UP	88.3	908.1	0.0	0.0
RL + UN	79.3	515.3	116.2	22.5
RL + Ov	87.3	904.7	0.0	0.0
RL + UP-UN-Ov	88.3	908.1	1.0	0.1
RL + UN-UP-Ov	80.6	657.0	161.8	24.6
RL + BestStrategy	87.0	731.1	93.9	12.8
RL + BestPosRule	**88.9**	760.7	0.0	0.0
RL + BestRule	87.2	734.8	121.1	16.5
		IG		
RL + UP	85.7	1012.8	0.0	0.0
RL + UN	79.3	597.4	175.9	29.4
RL + Ov	85.6	970.3	0.0	0.0
RL + UP-UN-Ov	85.7	1012.8	2.6	0.3
RL + UN-UP-Ov	79.1	635.0	196.1	30.9
RL + BestStrategy	85.6	706.1	133.8	18.9
RL + BestPosRule	**87.7**	759.6	0.0	0.0
RL + BestRule	85.6	712.4	171.6	24.1
TFPC-DelSN_contGO	80.8	1439.7	0.0	0.0
TFPC-DelSN_contGW	78.2	1875.6	0.0	0.0
TFPC-DelSO_contGN	0.0	0.0	0.0	0.0
TFPC-DelSO_contGW	45.2	1035.3	0.0	0.0

Table C.4: Average accuracy and the average number of rules for the classification of the 20NG-B dataset using χ^2 and IG for feature selection and the mixed representation

Overall, the strategies that did not generate rules with negation had the best results in the multi-class classification task for the 20 Newsgroups dataset using keyphrases for the text representation. This was consistent with the evaluation of the use of keywords and n-gram phrases as the text representation in the context of multi-class classification for the 20 Newsgroups dataset. The use of χ^2 enabled the RL strategies to achieve higher average accuracies than the use of IG. When compared to the TFPC algorithm, the RL strategies performed better.

C.1.2 Results for Using the KP-3 Representation

Strategies	F_1	Average # of rules	Average # of rules with negation	% of rules with negation
χ^2				
RL + UP	**0.873**	2562.5	0.0	0.0
RL + UN	0.859	2364.9	126.1	5.3
RL + Ov	0.859	2445.1	0.0	0.0
RL + UP-UN-Ov	**0.873**	2562.4	6.6	0.3
RL + UN-UP-Ov	0.862	2413.9	126.9	5.3
RL + BestStrategy	0.864	2397.3	69.9	2.9
RL + BestPosRule	0.866	2417.3	0.0	0.0
RL + BestRule	0.864	2402.9	124.2	5.2
IG				
RL + UP	0.828	2926.3	0.0	0.0
RL + UN	0.799	2478.2	409.7	16.5
RL + Ov	0.796	2702.7	0.0	0.0
RL + UP-UN-Ov	**0.829**	2926.9	13.1	0.4
RL + UN-UP-Ov	0.801	2532.3	420.1	16.6
RL + BestStrategy	0.813	2544.9	203.5	8.0
RL + BestPosRule	0.817	2582.3	0.0	0.0
RL + BestRule	0.810	2548.4	398.5	15.6

Table C.5: Micro-averaged F_1-measure and the average number of rules for the classification of the 20NG-A dataset using χ^2 and IG for feature selection and the KP-3 representation

Techniques	χ^2	Rank	IG	Rank
KP-3				
RL	0.873	1	0.829	2
SMO	0.853	2	0.832	1
NB	0.587	6	0.552	6
JRip	0.665	5	0.628	5
OlexGreedy	0.714	4	0.688	4
OlexGA	0.719	3	0.694	3

Table C.6: Micro-averaged F_1-measure for the classification of the 20NG-A dataset using χ^2 and IG for feature selection for the best RL strategy in comparison with the other machine learning techniques using the KP-3 representation

Strategies	F_1	Average # of rules	Average # of rules with negation	% of rules with negation
χ^2				
RL + UP	**0.891**	2432.1	0.0	0.0
RL + UN	0.881	2274.1	114.7	5.0
RL + Ov	0.880	2367.2	0.0	0.0
RL + UP-UN-Ov	**0.891**	2432.9	9.7	0.4
RL + UN-UP-Ov	0.882	2329.8	125.7	5.4
RL + BestStrategy	0.884	2280.0	67.5	3.0
RL + BestPosRule	0.882	2287.3	0.0	0.0
RL + BestRule	0.883	2278.9	112.9	5.0
IG				
RL + UP	**0.844**	2735.7	0.0	0.0
RL + UN	0.825	2392.8	384.5	16.1
RL + Ov	0.822	2590.7	0.0	0.0
RL + UP-UN-Ov	**0.844**	2735.8	12.3	0.4
RL + UN-UP-Ov	0.826	2446.7	401.4	16.4
RL + BestStrategy	0.836	2397.0	182.2	7.6
RL + BestPosRule	0.838	2428.9	0.0	0.0
RL + BestRule	0.832	2398.1	370.3	15.4

Table C.7: Micro-averaged F_1-measure and the average number of rules for the classification of the 20NG-B dataset using χ^2 and IG for feature selection and the KP-3 representation

Techniques	χ^2	Rank	IG	Rank
KP-3				
RL	0.891	1	0.844	2
SMO	0.873	2	0.845	1
NB	0.604	6	0.587	6
JRip	0.746	5	0.713	5
OlexGreedy	0.758	4	0.725	4
OlexGA	0.763	3	0.730	3

Table C.8: Micro-averaged F_1-measure for the classification of the 20NG-A dataset using χ^2 and IG for feature selection for the best RL strategy in comparison with the other machine learning techniques using the KP-3 representation

Strategies	Average Acc (%)	Average # of rules	Average # of rules with negation	% of rules with negation
χ^2				
RL + UP	**81.8**	2562.5	0.0	0.0
RL + UN	80.2	2364.9	126.1	5.3
RL + Ov	81.1	2445.1	0.0	0.0
RL + UP-UN-Ov	**81.8**	2562.4	6.6	0.3
RL + UN-UP-Ov	80.3	2413.9	126.9	5.3
RL + BestStrategy	80.5	2397.3	69.9	2.9
RL + BestPosRule	81.4	2417.3	0.0	0.0
RL + BestRule	80.5	2402.9	124.2	5.2
IG				
RL + UP	**77.3**	2926.3	0.0	0.0
RL + UN	73.7	2478.2	409.7	16.5
RL + Ov	76.3	2702.7	0.0	0.0
RL + UP-UN-Ov	**77.3**	2926.9	13.1	0.4
RL + UN-UP-Ov	73.6	2532.3	420.1	16.6
RL + BestStrategy	75.3	2544.9	203.5	8.0
RL + BestPosRule	76.9	2582.3	0.0	0.0
RL + BestRule	75.1	2548.4	398.5	15.6

Table C.9: Average accuracy and the average number of rules for the classification of the 20NG-A dataset using χ^2 and IG for feature selection and the KP-3 representation

Strategies	Average Acc (%)	Average # of rules	Average # of rules with negation	% of rules with negation
χ^2				
RL + UP	83.0	2432.1	0.0	0.0
RL + UN	81.8	2274.1	114.7	5.0
RL + Ov	**83.3**	2367.2	0.0	0.0
RL + UP-UN-Ov	82.9	2432.9	9.7	0.4
RL + UN-UP-Ov	81.6	2329.8	125.7	5.4
RL + BestStrategy	82.1	2280.0	67.5	3.0
RL + BestPosRule	82.4	2287.3	0.0	0.0
RL + BestRule	82.0	2278.9	112.9	5.0
IG				
RL + UP	78.0	2735.7	0.0	0.0
RL + UN	76.3	2392.8	384.5	16.1
RL + Ov	**78.7**	2590.7	0.0	0.0
RL + UP-UN-Ov	78.0	2735.8	12.3	0.4
RL + UN-UP-Ov	75.8	2446.7	401.4	16.4
RL + BestStrategy	76.8	2397.0	182.2	7.6
RL + BestPosRule	78.2	2428.9	0.0	0.0
RL + BestRule	76.8	2398.1	370.3	15.4

Table C.10: Average accuracy and the average number of rules for the classification of the 20NG-B dataset using χ^2 and IG for feature selection and the KP-3 representation

C.2 Experiments Using the Reuters-8 Dataset

C.2.1 Results and Analysis for Multi-Class Classification

The evaluation results produced using the eight strategies in the proposed IRL mechanism for the multi-class classification of the Reuters-8 dataset is discussed in this subsection. The TFPC algorithm with its four phrase selection strategies were compared to the RL strategies. Table C.11 shows the average accuracy obtained and the average number of rules generated with respect to the Reuters-8 dataset for the eight strategies in the proposed IRL mechanism using χ^2 and IG, and the KP-2 representation. Table C.12 shows the same results but using the mixed representation in comparison to the TFPC algorithm with its four phrase selection strategies.

As can be seen from Table C.11, using the KP-2 representation, the best RL strategy was RL + BestPosRule (when χ^2 was used) and RL + Ov (when IG was used). Both these strategies did not generate any rules with negation. The worst strategy was RL + BestStrategy when χ^2 was used, while RL + UP and RL + UP-UN-OV were equally the worst when IG was used. The difference between the best and worst strategies was 1.5% (when χ^2 was used) and 4.2% (when IG was used). These figures translated to an additional average of 99.6 and 279.0 (out of 6,643) documents being correctly classified by the best RL strategies. The use of χ^2 resulted in the RL strategies generating higher average accuracies compared to when IG was used.

Strategies	Average Acc (%)	Average # of rules	Average # of rules with negation	% of rules with negation
χ^2				
RL + UP	91.2	957.2	0.0	0.0
RL + UN	91.1	454.5	181.4	39.9
RL + Ov	92.1	774.7	0.0	0.0
RL + UP-UN-Ov	91.2	957.2	0.0	0.0
RL + UN-UP-Ov	91.0	472.9	184.8	39.1
RL + BestStrategy	90.9	510.7	123.1	24.1
RL + BestPosRule	**92.4**	568.7	0.0	0.0
RL + BestRule	91.2	505.1	206.1	40.8
IG				
RL + UP	87.7	968.9	0.0	0.0
RL + UN	90.1	436.5	220.0	50.4
RL + Ov	**91.9**	742.5	0.0	0.0
RL + UP-UN-Ov	87.7	968.9	0.2	0.0
RL + UN-UP-Ov	90.1	452.0	223.6	49.5
RL + BestStrategy	90.1	475.4	124.6	26.2
RL + BestPosRule	91.0	547.6	0.0	0.0
RL + BestRule	90.4	475.0	239.6	50.4

Table C.11: Average accuracy and the average number of rules for the classification of the Reuters-8 dataset using χ^2 and IG for feature selection and the KP-2 representation

In the case of using the mixed representation, the best RL strategy was RL + BestPosRule, regardless of the feature selection technique used, as shown in Table C.12. The worst strategies were RL + UN and RL + UN-UP-Ov respectively when χ^2 and IG were used. The difference in accuracies between the best and worst RL strategies was 20.0% (when χ^2 was used) and 11.8% (when IG was used), indicating that an additional average of 1328.6 and 783.9 (out of 6,643) documents were correctly classified by RL + BestPosRule. The use of χ^2 however, did not show a definite trend of higher average accuracy values being obtained by the RL strategies, as it was in the case of the KP-2 representation. When the RL strategies were compared to the TFPC algorithm, it was observed that the best RL strategy was 17.9% better than the best TFPC phrase selection strategy. This translated to an additional average of 1189.1 (out of 6,643) documents being correctly classified by the best RL strategy. In fact, all the RL strategies were better than the best phrase selection strategy in the TFPC algorithm, except for RL + UN and RL + UN-UP-Ov.

Strategies	Average Acc (%)	Average # of rules	Average # of rules with negation	% of rules with negation
		χ^2		
RL + UP	94.0	400.6	0.0	0.0
RL + UN	74.6	163.8	58.8	35.9
RL + Ov	90.6	292.9	0.0	0.0
RL + UP-UN-Ov	94.0	400.6	0.1	0.0
RL + UN-UP-Ov	75.5	202.0	75.3	37.3
RL + BestStrategy	94.5	262.4	56.6	21.6
RL + BestPosRule	**94.6**	290.4	0.0	0.0
RL + BestRule	94.4	267.5	72.2	27.0
		IG		
RL + UP	93.0	471.3	0.0	0.0
RL + UN	82.4	201.0	102.1	50.8
RL + Ov	93.8	362.2	0.0	0.0
RL + UP-UN-Ov	93.0	471.3	0.7	0.1
RL + UN-UP-Ov	82.3	221.5	110.7	50.0
RL + BestStrategy	93.9	257.1	87.0	33.8
RL + BestPosRule	**94.1**	299.6	0.0	0.0
RL + BestRule	93.7	260.6	108.3	41.6
TFPC-DelSN_contGO	76.7	1084.8	0.0	0.0
TFPC-DelSN_contGW	74.8	1399.7	0.0	0.0
TFPC-DelSO_contGN	0.0	0.0	0.0	0.0
TFPC-DelSO_contGW	72.6	996.0	0.0	0.0

Table C.12: Average accuracy and the average number of rules for the classification of the Reuters-8 dataset using χ^2 and IG for feature selection and the mixed representation

Overall, the strategies that did not generate rules with negation performed better with respect to the multi-class classification of the Reuters-8 dataset, regardless of the feature selection technique or text representation used. In comparison to the TFPC algorithm, the best RL strategy produced much better results.

C.2.2 Results for Using the KP-3 Representation

Strategies	F_1	Average # of rules	Average # of rules with negation	% of rules with negation
χ^2				
RL + UP	**0.908**	1357.9	0.0	0.0
RL + UN	0.887	794.5	225.9	28.4
RL + Ov	0.890	1106.6	0.0	0.0
RL + UP-UN-Ov	**0.908**	1357.9	0.8	0.1
RL + UN-UP-Ov	0.887	812.7	229.0	28.2
RL + BestStrategy	0.906	813.8	147.2	18.1
RL + BestPosRule	0.904	872.8	0.0	0.0
RL + BestRule	0.907	824.2	246.2	29.9
IG				
RL + UP	0.842	1613.8	0.0	0.0
RL + UN	0.855	852.7	370.6	43.5
RL + Ov	0.854	1261.4	0.0	0.0
RL + UP-UN-Ov	0.842	1613.8	2.5	0.2
RL + UN-UP-Ov	0.855	870.5	372.9	42.8
RL + BestStrategy	0.872	896.8	157.9	17.6
RL + BestPosRule	**0.877**	947.2	0.0	0.0
RL + BestRule	0.865	897.8	392.5	43.7

Table C.13: Micro-averaged F_1-measure and the average number of rules for the classification of the Reuters-8 dataset using χ^2 and IG for feature selection and the KP-3 representation

Techniques	χ^2	Rank	IG	Rank
KP-3				
RL	0.908	2	0.877	2
SMO	0.916	1	0.888	1
NB	0.799	6	0.792	6
JRip	0.854	5	0.852	5
OlexGreedy	0.875	4	0.874	4
OlexGA	0.881	3	0.877	2

Table C.14: Micro-averaged F_1-measure for the classification of the Reuters-8 dataset using χ^2 and IG for feature selection for the best RL strategy in comparison with the other machine learning techniques using the KP-3 representation

Strategies	Average Acc (%)	Average # of rules	Average # of rules with negation	% of rules with negation
χ^2				
RL + UP	88.4	1357.9	0.0	0.0
RL + UN	85.6	794.5	225.9	28.4
RL + Ov	**89.4**	1106.6	0.0	0.0
RL + UP-UN-Ov	88.4	1357.9	0.8	0.1
RL + UN-UP-Ov	85.4	812.7	229.0	28.2
RL + BestStrategy	87.9	813.8	147.2	18.1
RL + BestPosRule	**89.4**	872.8	0.0	0.0
RL + BestRule	88.5	824.2	246.2	29.9
IG				
RL + UP	83.1	1613.8	0.0	0.0
RL + UN	84.6	852.7	370.6	43.5
RL + Ov	**88.0**	1261.4	0.0	0.0
RL + UP-UN-Ov	83.1	1613.8	2.5	0.2
RL + UN-UP-Ov	86.8	870.5	372.9	42.8
RL + BestStrategy	85.6	896.8	157.9	17.6
RL + BestPosRule	**88.0**	947.2	0.0	0.0
RL + BestRule	85.7	897.8	392.5	43.7

Table C.15: Average accuracy and the average number of rules for the classification of the Reuters-8 dataset using χ^2 and IG for feature selection and the KP-3 representation

C.3 Experiments Using the SAVSNET Dataset

C.3.1 Results and Analysis for Multi-Class Classification

This sub-section discusses the evaluation of the eight strategies in the proposed IRL mechanism for the multi-class classification of the SAVSNET dataset. Table C.16 shows the average accuracy obtained and the average number of rules generated with respect to the SAVSNET dataset for the eight strategies in the proposed IRL mechanism using χ^2 and IG, and the KP-2 representation. Table C.17 shows the same data for the eight strategies in the proposed IRL mechanism using the mixed representation in comparison to the TFPC algorithm with its four phrase selection strategies.

Strategies	Average Acc (%)	Average # of rules	Average # of rules with negation	% of rules with negation
χ^2				
RL + UP	88.9	183.5	0.0	0.0
RL + UN	86.7	145.0	24.4	16.8
RL + Ov	**89.2**	176.1	0.0	0.0
RL + UP-UN-Ov	88.9	183.5	0.8	0.4
RL + UN-UP-Ov	87.6	146.8	25.5	17.4
RL + BestStrategy	88.0	151.7	14.9	9.8
RL + BestPosRule	88.6	164.7	0.0	0.0
RL + BestRule	88.0	150.4	24.7	16.4
IG				
RL + UP	**87.9**	219.8	0.0	0.0
RL + UN	79.7	151.4	36.9	24.4
RL + Ov	84.4	202.3	0.0	0.0
RL + UP-UN-Ov	**87.9**	219.8	0.0	0.0
RL + UN-UP-Ov	80.0	153.6	38.0	24.7
RL + BestStrategy	85.3	172.4	23.9	13.9
RL + BestPosRule	87.4	189.5	0.0	0.0
RL + BestRule	85.3	173.0	38.7	22.4

Table C.16: Average accuracy and the average number of rules for the classification of the SAVSNET dataset using χ^2 and IG for feature selection and the KP-2 representation

When the KP-2 representation was used for the SAVSNET dataset, the strategies that did not generate rules with negation performed better. When χ^2 was used, the best RL strategy was RL + Ov while RL + UP and RL + UP-UN-Ov were the best strategies when IG was used. The worst strategy was RL + UN, regardless of the feature selection technique used. The difference in average accuracies between the best and worst strategies was 2.5% (when χ^2 was used) and 8.2% (when IG was used). These translated to an additional average of 20.7 and 67.9 (out of 828) documents

being correctly classified by the best RL strategy. Again, the use of χ^2 resulted in the RL strategies generating higher average accuracies compared to the use of IG.

Strategies	Average Acc (%)	Average # of rules	Average # of rules with negation	% of rules with negation
		χ^2		
RL + UP	79.8	51.7	0.0	0.0
RL + UN	72.7	39.8	18.7	47.0
RL + Ov	81.5	53.5	0.0	0.0
RL + UP-UN-Ov	79.8	51.7	0.0	0.0
RL + UN-UP-Ov	71.7	43.2	18.4	42.6
RL + BestStrategy	79.6	45.4	11.6	25.6
RL + BestPosRule	**81.6**	47.4	0.0	0.0
RL + BestRule	80.2	45.1	11.7	25.9
		IG		
RL + UP	72.5	31.0	0.0	0.0
RL + UN	72.1	26.0	10.7	41.2
RL + Ov	**81.2**	33.1	0.0	0.0
RL + UP-UN-Ov	72.5	31.0	0.0	0.0
RL + UN-UP-Ov	71.8	27.5	11.1	40.4
RL + BestStrategy	77.1	30.0	4.6	15.3
RL + BestPosRule	77.4	30.4	0.0	0.0
RL + BestRule	76.9	30.2	5.2	17.2
TFPC-DelSN_contGO	38.4	1614.0	0.0	0.0
TFPC-DelSN_contGW	41.7	1541.1	0.0	0.0
TFPC-DelSO_contGN	39.5	1742.2	0.0	0.0
TFPC-DelSO_contGW	43.0	1587.6	0.0	0.0

Table C.17: Average accuracy and the average number of rules for the classification of the SAVSNET dataset using χ^2 and IG for feature selection and the mixed representation

In the mixed representation of the SAVSNET dataset, the best RL strategies were again those that generated rules without negation. When χ^2 was used, RL + BestPosRule was the best RL strategy while RL + UN-UP-Ov was the worst. Their average accuracy difference was 9.9%, which equated to an additional average of 82.0 (out of 828) documents being correctly classified by RL + BestPosRule. When IG was used, the best RL strategy was RL + Ov, while the worst was again RL + UN-UP-Ov, producing an average accuracy difference of 9.4%. This translated to an additional average of 77.8 (out of 828) documents being correctly classified by RL + Ov. The use of χ^2 again enabled the RL strategies to produce higher average accuracies compared to the use of IG (except for RL + UN-UP-Ov). In comparison to the TFPC algorithm, all the RL strategies were undoubtably better. The best RL strategy outperformed the best TFPC phrase selection strategy by a substantial 38.6%, which translated to an

additional average of 319.6 (out of 828) documents being correctly classified by the best RL strategy.

Overall, with respect to the multi-class classification of the SAVSNET dataset, the strategies that did not generate rules with negation were better than those which did. It was also generally noted that the use of χ^2 enabled the RL strategies to produce higher average accuracies compared to the use of IG. When compared to the TFPC algorithm, the RL strategies were substantially better.

C.3.2 Results for Using the KP-3 Representation

Strategies	F_1	Average # of rules	Average # of rules with negation	% of rules with negation
χ^2				
RL + UP	0.824	287.3	0.0	0.0
RL + UN	**0.835**	226.2	30.4	13.4
RL + Ov	0.831	263.9	0.0	0.0
RL + UP-UN-Ov	0.824	287.3	0.0	0.0
RL + UN-UP-Ov	0.830	237.9	31.7	13.3
RL + BestStrategy	0.831	235.4	17.1	7.3
RL + BestPosRule	0.831	243.6	0.0	0.0
RL + BestRule	0.834	232.1	29.9	12.9
IG				
RL + UP	0.764	322.2	0.0	0.0
RL + UN	0.795	243.9	51.5	21.1
RL + Ov	0.795	283.3	0.0	0.0
RL + UP-UN-Ov	0.763	322.2	1.8	0.6
RL + UN-UP-Ov	0.791	251.8	53.0	21.0
RL + BestStrategy	0.787	250.1	29.5	11.8
RL + BestPosRule	0.780	260.9	0.0	0.0
RL + BestRule	**0.796**	249.5	49.9	20.0

Table C.18: Micro-averaged F_1-measure and the average number of rules for the classification of the SAVSNET dataset using χ^2 and IG for feature selection and the KP-3 representation

Techniques	χ^2	Rank	IG	Rank
KP-3				
RL	0.835	1	0.796	1
SMO	0.796	2	0.759	2
NB	0.730	5	0.699	5
JRip	0.699	6	0.695	6
OlexGreedy	0.747	4	0.731	4
OlexGA	0.770	3	0.745	3

Table C.19: Micro-averaged F_1-measure for the classification of the SAVSNET dataset using χ^2 and IG for feature selection for the best RL strategy in comparison with the other machine learning techniques using the KP-3 representation

Strategies	Average Acc (%)	Average # of rules	Average # of rules with negation	% of rules with negation
χ^2				
RL + UP	81.8	287.3	0.0	0.0
RL + UN	80.9	226.2	30.4	13.4
RL + Ov	**82.4**	263.9	0.0	0.0
RL + UP-UN-Ov	81.8	287.3	0.0	0.0
RL + UN-UP-Ov	80.3	237.9	31.7	13.3
RL + BestStrategy	81.2	235.4	17.1	7.3
RL + BestPosRule	81.7	243.6	0.0	0.0
RL + BestRule	81.4	232.1	29.9	12.9
IG				
RL + UP	74.8	322.2	0.0	0.0
RL + UN	76.6	243.9	51.5	21.1
RL + Ov	**77.9**	283.3	0.0	0.0
RL + UP-UN-Ov	74.7	322.2	1.8	0.6
RL + UN-UP-Ov	76.1	251.8	53.0	21.0
RL + BestStrategy	77.0	250.1	29.5	11.8
RL + BestPosRule	76.3	260.9	0.0	0.0
RL + BestRule	76.9	249.5	49.9	20.0

Table C.20: Average accuracy and the average number of rules for the classification of the SAVSNET dataset using χ^2 and IG for feature selection and the KP-3 representation

Appendix D

Additional Results and Analysis for Using the Fuzzy Phrase Representation

D.1 Experiments Using the 20 Newsgroups Dataset

D.1.1 Results and Analysis for Multi-Class Classification

The evaluation of the eight strategies in the proposed IRL mechanism with respect to the multi-class classification task in comparison with the TFPC algorithm with its four phrase selection strategies is discussed in this sub-section. Table D.1 shows the average accuracy obtained and the average number of rules generated with respect to the 20NG-A dataset for the eight strategies in the proposed IRL mechanism using χ^2 and IG and the FP-1 representation. Table D.2 shows the same results for the eight strategies in the proposed IRL mechanism but using the mixed representation in comparison to the TFPC algorithm and its associated four phrase selection strategies. Tables D.3 and D.4 show the same results but for the classification of the 20NG-B dataset.

Although the results recorded in Table D.1 for the 20NG-A dataset were very close for all the RL strategies, some of the RL strategies performed slightly better than others. The best RL strategies when χ^2 was used with the FP-1 representation were RL + Ov and RL + BestPosRule, both of which did not generate any rules with negation. When IG was used, RL + BestPosRule was the best RL strategy. The worst strategies were RL + UN and RL + UN-UP-OV when χ^2 was used and RL + UN-UP-Ov when IG was used. The difference between the best and worst results was 0.6% (when χ^2 was used) and 1.1% (when IG was used), which translated to an additional average of 60 and 110 (out of 10,000) documents being correctly classified by the best RL strategies. The use of χ^2 enabled the strategies to achieve higher average accuracies compared to the use of IG.

Strategies	Average Acc (%)	Average # of rules	Average # of rules with negation	% of rules with negation
χ^2				
RL + UP	64.8	1228.1	0.0	0.0
RL + UN	64.4	1207.0	12.6	1.0
RL + Ov	**65.0**	1214.9	0.0	0.0
RL + UP-UN-Ov	64.7	1228.0	2.3	0.2
RL + UN-UP-Ov	64.4	1218.5	12.6	1.0
RL + BestStrategy	64.7	1212.5	10.8	0.9
RL + BestPosRule	**65.0**	1213.6	0.0	0.0
RL + BestRule	64.7	1212.6	12.5	1.0
IG				
RL + UP	63.1	1457.7	0.0	0.0
RL + UN	62.8	1397.4	48.5	3.5
RL + Ov	63.5	1421.9	0.0	0.0
RL + UP-UN-Ov	63.1	1457.2	7.4	0.5
RL + UN-UP-Ov	62.5	1417.8	48.4	3.4
RL + BestStrategy	63.1	1401.3	40.5	2.9
RL + BestPosRule	**63.6**	1405.5	0.0	0.0
RL + BestRule	63.2	1401.7	47.9	3.4

Table D.1: Average accuracy and the average number of rules for the classification of the 20NG-A dataset using χ^2 and IG for feature selection and the FP-1 representation

When the mixed representation was used for the 20NG-A dataset, RL + UP and RL + UP-UN-OV recorded the best results using χ^2, as shown in Table D.2. These two RL strategies did not generate any rules with negation. The worst RL strategy was RL + UN, which was 5.2% less accurate, thus signifying that an average of 520 (out of 10,000) fewer documents were classified correctly. When IG was used, three RL strategies recorded an equal best result. The worst RL strategies were RL + UN and RL + UN-UP-OV. The difference in accuracy between the best and worst strategy was 4.8%, which translated to an additional average of 480.0 (out of 10,000) documents being correctly classified by the best RL strategies. Again, when χ^2 was used, the RL strategies achieved higher average accuracies compared to when IG was used. In comparison with the TFPC algorithm, the best RL strategy was 18.5% more accurate than the best phrase selection strategy in the TFPC algorithm. This translated to an additional average of 1,850.0 (out of 10,000) documents being correctly classified by the best RL strategy. In fact, all the RL strategies were better than the TFPC algorithm.

Table D.3 presents the classification results for the 20NG-B dataset when using the FP-1 representation. When χ^2 was used, all the RL strategies recorded very close results. RL + OV, which did not generate any rules with negation, was the best RL strategy while the worst strategy was RL + UN-UP-OV. The difference in accuracy was 0.4% which translated to an additional average of 40.0 (out of 9,997) documents being correctly classified by RL + Ov. When IG was used, the best RL strategy was RL +

BestPosRule, which also did not generate any rules with negation. RL + UN-UP-OV was again the worst strategy by 0.8%, whereby an average of 80.0 (out of 9,997) fewer documents were being correctly classified. The use of χ^2 enabled the RL strategies to achieve higher average accuracy compared to when IG was used.

Strategies	Average Acc (%)	Average # of rules	Average # of rules with negation	% of rules with negation
		χ^2		
RL + UP	**95.2**	1017.1	0.0	0.0
RL + UN	90.0	810.0	34.4	4.2
RL + Ov	94.2	987.9	0.0	0.0
RL + UP-UN-Ov	**95.2**	1017.1	0.0	0.0
RL + UN-UP-Ov	91.2	933.4	41.9	4.5
RL + BestStrategy	94.1	928.9	32.6	3.5
RL + BestPosRule	94.6	916.0	0.0	0.0
RL + BestRule	94.1	905.8	34.3	3.8
		IG		
RL + UP	**89.6**	1352.9	0.0	0.0
RL + UN	84.8	880.0	235.5	26.8
RL + Ov	89.2	1295.5	0.0	0.0
RL + UP-UN-Ov	**89.6**	1352.9	0.3	0.0
RL + UN-UP-Ov	84.8	881.4	236.1	26.8
RL + BestStrategy	88.3	963.6	173.7	18.0
RL + BestPosRule	**89.6**	1107.8	0.0	0.0
RL + BestRule	88.7	979.6	244.6	25.0
TFPC-DelSN_contGO	76.7	1417.5	0.0	0.0
TFPC-DelSN_contGW	75.8	1767.0	0.0	0.0
TFPC-DelSO_contGN	0.0	0.0	0.0	0.0
TFPC-DelSO_contGW	50.2	1022.7	0.0	0.0

Table D.2: Average accuracy and the average number of rules for the classification of the 20NG-A dataset using χ^2 and IG for feature selection and the mixed representation

When the mixed representation was used with χ^2 for the 20NG-B dataset, RL + UP and RL + UP-UN-Ov were the best RL strategies while RL + UN-UP-OV was the worst, as shown in Table D.4. The difference in accuracy between the best and worst RL strategies was 4.7%, which translated to an additional average of 469.9 (out of 9,997) documents being correctly classified by the best RL strategies. When IG was used, RL + BestPosRule was the best RL strategy, achieving 5.9% higher accuracy than the worst RL strategy, which signified an additional average of 589.8 (out of 9,997) documents being correctly classified. Again, when χ^2 was used, higher average accuracies were obtained by the RL strategies, as compared to when IG was used. When compared to the best phrase selection strategy for the TFPC algorithm, the best RL strategy was 14.6% more accurate, which means an additional average of 1,459.6 (out of 9,997)

documents were being correctly classified. Again, as observed in the classification of the 20NG-A dataset, all the RL strategies performed better than the TFPC algorithm.

Strategies	Average Acc (%)	Average # of rules	Average # of rules with negation	% of rules with negation
χ^2				
RL + UP	67.3	1467.7	0.0	0.0
RL + UN	67.5	1452.5	7.9	0.5
RL + Ov	**67.6**	1457.5	0.0	0.0
RL + UP-UN-Ov	67.3	1467.7	3.2	0.2
RL + UN-UP-Ov	67.2	1466.1	9.0	0.6
RL + BestStrategy	67.5	1452.5	6.4	0.4
RL + BestPosRule	67.4	1452.7	0.0	0.0
RL + BestRule	67.5	1452.5	7.9	0.5
IG				
RL + UP	64.9	1649.2	0.0	0.0
RL + UN	64.7	1576.5	52.9	3.4
RL + Ov	65.1	1624.2	0.0	0.0
RL + UP-UN-Ov	64.9	1649.0	7.5	0.5
RL + UN-UP-Ov	64.6	1615.5	55.4	3.4
RL + BestStrategy	65.0	1581.9	36.1	2.3
RL + BestPosRule	**65.4**	1586.5	0.0	0.0
RL + BestRule	65.0	1582.0	51.1	3.2

Table D.3: Average accuracy and the average number of rules for the classification of the 20NG-B dataset using χ^2 and IG for feature selection and the FP-1 representation

Strategies	Average Acc (%)	Average # of rules	Average # of rules with negation	% of rules with negation
		χ^2		
RL + UP	**95.4**	966.8	0.0	0.0
RL + UN	90.9	714.7	44.2	6.2
RL + Ov	94.0	927.2	0.0	0.0
RL + UP-UN-Ov	**95.4**	966.8	0.0	0.0
RL + UN-UP-Ov	90.7	796.4	44.2	5.5
RL + BestStrategy	94.3	847.4	33.1	3.9
RL + BestPosRule	95.0	850.9	0.0	0.0
RL + BestRule	94.3	831.6	40.7	4.9
		IG		
RL + UP	91.1	1399.5	0.0	0.0
RL + UN	85.7	801.7	216.8	27.0
RL + Ov	89.7	1319.8	0.0	0.0
RL + UP-UN-Ov	91.1	1399.5	0.0	0.0
RL + UN-UP-Ov	85.7	802.1	217.1	27.1
RL + BestStrategy	89.4	975.4	183.4	18.8
RL + BestPosRule	**91.6**	1089.9	0.0	0.0
RL + BestRule	89.4	978.5	257.2	26.3
TFPC-DelSN_contGO	80.8	1439.7	0.0	0.0
TFPC-DelSN_contGW	78.2	1875.6	0.0	0.0
TFPC-DelSO_contGN	0.0	0.0	0.0	0.0
TFPC-DelSO_contGW	45.2	1035.3	0.0	0.0

Table D.4: Average accuracy and the average number of rules for the classification of the 20NG-B dataset using χ^2 and IG for feature selection and the mixed representation

Overall, the RL strategies that did not generate any rules with negation were shown to be better, with respect to the multi-class classification of the 20 Newsgroups dataset when fuzzy phrases were used as the text representation. This was consistent with the evaluation of the use of keywords, n-gram phrases and keyphrases as the text representation for the multi-class classification of the 20 Newsgroups dataset. Higher average accuracies were produced by the RL strategies when χ^2 was used, compared to when IG was used. In comparison with the TFPC algorithm, the RL strategies performed better.

D.1.2 Results for Using the FP-2 Representation

Strategies	F_1	Average # of rules	Average # of rules with negation	% of rules with negation
χ^2				
RL + UP	**0.703**	1352.3	0.0	0.0
RL + UN	0.701	1333.6	10.9	0.8
RL + Ov	0.702	1339.5	0.0	0.0
RL + UP-UN-Ov	**0.703**	1352.3	2.7	0.2
RL + UN-UP-Ov	0.700	1342.1	10.9	0.8
RL + BestStrategy	**0.703**	1338.6	9.9	0.7
RL + BestPosRule	**0.703**	1339.5	0.0	0.0
RL + BestRule	**0.703**	1338.8	10.2	0.8
IG				
RL + UP	**0.699**	1715.4	0.0	0.0
RL + UN	0.692	1661.5	37.1	2.2
RL + Ov	0.693	1684.5	0.0	0.0
RL + UP-UN-Ov	**0.699**	1715.2	5.2	0.3
RL + UN-UP-Ov	0.696	1685.3	38.2	2.3
RL + BestStrategy	0.696	1673.3	29.5	1.8
RL + BestPosRule	0.697	1673.6	0.0	0.0
RL + BestRule	0.697	1673.4	36.3	2.2

Table D.5: Micro-averaged F_1-measure and the average number of rules for the classification of the 20NG-A dataset using χ^2 and IG for feature selection and the FP-2 representation

Techniques	χ^2	Rank	IG	Rank
FP-2				
RL	0.703	1	0.699	1
SMO	0.680	2	0.670	2
NB	0.268	6	0.280	6
JRip	0.584	5	0.526	5
OlexGreedy	0.617	4	0.625	4
OlexGA	0.630	3	0.637	3

Table D.6: Micro-averaged F_1-measure for the classification of the 20NG-A dataset using χ^2 and IG for feature selection for the best RL strategy in comparison with the other machine learning techniques using the FP-2 representation

Strategies	F_1	Average # of rules	Average # of rules with negation	% of rules with negation
χ^2				
RL + UP	0.755	1563.8	0.0	0.0
RL + UN	0.755	1544.5	7.4	0.5
RL + Ov	0.755	1544.7	0.0	0.0
RL + UP-UN-Ov	0.755	1563.8	3.5	0.2
RL + UN-UP-Ov	0.755	1560.8	7.4	0.5
RL + BestStrategy	**0.757**	1547.3	6.4	0.4
RL + BestPosRule	**0.757**	1547.3	0.0	0.0
RL + BestRule	**0.757**	1547.3	7.4	0.5
IG				
RL + UP	0.734	1832.7	0.0	0.0
RL + UN	0.733	1780.9	41.3	2.3
RL + Ov	0.734	1799.6	0.0	0.0
RL + UP-UN-Ov	0.734	1832.5	7.9	0.4
RL + UN-UP-Ov	0.735	1815.7	42.6	2.3
RL + BestStrategy	0.735	1783.2	21.7	1.2
RL + BestPosRule	0.735	1782.1	0.0	0.0
RL + BestRule	**0.736**	1782.0	39.3	2.2

Table D.7: Micro-averaged F_1-measure and the average number of rules for the classification of the 20NG-B dataset using χ^2 and IG for feature selection and the FP-2 representation

Techniques	χ^2	Rank	IG	Rank
FP-2				
RL	0.757	1	0.736	1
SMO	0.732	2	0.712	2
NB	0.283	6	0.294	6
JRip	0.688	3	0.646	5
OlexGreedy	0.658	5	0.649	4
OlexGA	0.668	4	0.659	3

Table D.8: Micro-averaged F_1-measure for the classification of the 20NG-B dataset using χ^2 and IG for feature selection for the best RL strategy in comparison with the other machine learning techniques using the FP-2 representation

Strategies	Average Acc (%)	Average # of rules	Average # of rules with negation	% of rules with negation
χ^2				
RL + UP	**59.7**	1352.3	0.0	0.0
RL + UN	59.2	1333.6	10.9	0.8
RL + Ov	59.5	1339.5	0.0	0.0
RL + UP-UN-Ov	**59.7**	1352.3	2.7	0.2
RL + UN-UP-Ov	59.1	1342.1	10.9	0.8
RL + BestStrategy	59.5	1338.6	9.9	0.7
RL + BestPosRule	59.6	1339.5	0.0	0.0
RL + BestRule	59.6	1338.8	10.2	0.8
IG				
RL + UP	**60.2**	1715.4	0.0	0.0
RL + UN	58.7	1661.5	37.1	2.2
RL + Ov	59.3	1684.5	0.0	0.0
RL + UP-UN-Ov	60.1	1715.2	5.2	0.3
RL + UN-UP-Ov	59.4	1685.3	38.2	2.3
RL + BestStrategy	59.7	1673.3	29.5	1.8
RL + BestPosRule	59.9	1673.6	0.0	0.0
RL + BestRule	59.8	1673.4	36.3	2.2

Table D.9: Average accuracy and the average number of rules for the classification of the 20NG-A dataset using χ^2 and IG for feature selection and the FP-2 representation

Strategies	Average Acc (%)	Average # of rules	Average # of rules with negation	% of rules with negation
χ^2				
RL + UP	63.2	1563.8	0.0	0.0
RL + UN	63.0	1544.5	7.4	0.5
RL + Ov	63.0	1544.7	0.0	0.0
RL + UP-UN-Ov	63.2	1563.8	3.5	0.2
RL + UN-UP-Ov	63.1	1560.8	7.4	0.5
RL + BestStrategy	**63.4**	1547.3	6.4	0.4
RL + BestPosRule	**63.4**	1547.3	0.0	0.0
RL + BestRule	**63.4**	1547.3	7.4	0.5
IG				
RL + UP	61.2	1832.7	0.0	0.0
RL + UN	61.3	1780.9	41.3	2.3
RL + Ov	**61.6**	1799.6	0.0	0.0
RL + UP-UN-Ov	61.2	1832.5	7.9	0.4
RL + UN-UP-Ov	61.1	1815.7	42.6	2.3
RL + BestStrategy	61.3	1783.2	21.7	1.2
RL + BestPosRule	61.5	1782.1	0.0	0.0
RL + BestRule	61.4	1782.0	39.3	2.2

Table D.10: Average accuracy and the average number of rules for the classification of the 20NG-B dataset using χ^2 and IG for feature selection and the FP-2 representation

D.2 Experiments Using the Reuters-8 Dataset

D.2.1 Results and Analysis for Multi-Class Classification

This sub-section discusses the evaluation results obtained using the eight strategies in the proposed IRL mechanism for the multi-class classification of the Reuters-8 dataset.

Strategies	Average Acc (%)	Average # of rules	Average # of rules with negation	% of rules with negation
χ^2				
RL + UP	85.5	1076.5	0.0	0.0
RL + UN	85.2	725.2	92.7	12.8
RL + Ov	**87.5**	968.8	0.0	0.0
RL + UP-UN-Ov	85.5	1076.4	2.0	0.2
RL + UN-UP-Ov	84.4	784.8	96.9	12.3
RL + BestStrategy	85.9	768.5	79.5	10.3
RL + BestPosRule	86.7	793.9	0.0	0.0
RL + BestRule	86.1	772.0	101.9	13.2
IG				
RL + UP	75.0	918.2	0.0	0.0
RL + UN	80.3	616.6	130.7	21.2
RL + Ov	**81.3**	839.0	0.0	0.0
RL + UP-UN-Ov	74.9	917.9	9.1	1.0
RL + UN-UP-Ov	79.2	668.6	134.6	20.1
RL + BestStrategy	80.4	656.4	96.6	14.7
RL + BestPosRule	80.5	692.5	0.0	0.0
RL + BestRule	80.2	659.7	140.7	21.3

Table D.11: Average accuracy and the average number of rules for the classification of the Reuters-8 dataset using χ^2 and IG for feature selection and the FP-1 representation

The TFPC algorithm with its four phrase selection strategies were compared to the RL strategies. Table D.11 shows the average accuracy obtained and the average number of rules generated with respect to the Reuters-8 dataset for the eight strategies in the proposed IRL mechanism using χ^2 and IG and the FP-1 representation. Table D.12 shows the same results but using the mixed representation in comparison to the TFPC algorithm with its four phrase selection strategies.

Table D.11 shows that, regardless of the feature selection technique used, the best RL strategy was RL + Ov when the FP-1 representation was used. When χ^2 was used, the worst RL strategy was RL + UN-UP-Ov, which was 3.1% less accurate than RL + Ov. This translated to an additional average of 205.9 (out of 6,643) documents being correctly classified by RL + Ov. When IG was used, the worst RL strategy was RL + UP-UN-Ov. It was 6.4% less accurate than RL + Ov, indicating that RL + Ov accurately classified an additional average of 425.2 (out of 6,643) documents. The use

of χ^2 enabled the RL strategies to achieve higher average accuracies compared to when IG was used.

Strategies	Average Acc (%)	Average # of rules	Average # of rules with negation	% of rules with negation
		χ^2		
RL + UP	95.3	535.2	0.0	0.0
RL + UN	92.7	190.3	54.1	28.4
RL + Ov	94.6	330.6	0.0	0.0
RL + UP-UN-Ov	95.3	535.2	0.0	0.0
RL + UN-UP-Ov	92.2	258.3	75.4	29.2
RL + BestStrategy	94.4	312.4	55.7	17.8
RL + BestPosRule	**95.5**	377.7	0.0	0.0
RL + BestRule	95.0	300.4	95.4	31.8
		IG		
RL + UP	94.5	493.8	0.0	0.0
RL + UN	88.8	173.2	71.1	41.1
RL + Ov	94.3	315.4	0.0	0.0
RL + UP-UN-Ov	94.5	493.8	0.0	0.0
RL + UN-UP-Ov	88.6	197.3	84.1	42.6
RL + BestStrategy	94.2	269.4	66.5	24.7
RL + BestPosRule	**94.9**	355.1	0.0	0.0
RL + BestRule	94.8	260.1	102.0	39.2
TFPC-DelSN_contGO	76.7	1084.8	0.0	0.0
TFPC-DelSN_contGW	74.8	1399.7	0.0	0.0
TFPC-DelSO_contGN	0.0	0.0	0.0	0.0
TFPC-DelSO_contGW	72.6	996.0	0.0	0.0

Table D.12: Average accuracy and the average number of rules for the classification of the Reuters-8 dataset using χ^2 and IG for feature selection and the mixed representation

Table D.12 shows that RL + BestPosRule was the best RL strategy while RL + UN-UP-OV was the worst RL strategy when the mixed representation was used, regardless of the feature selection technique used. When χ^2 was used, RL + BestPosRule was 3.3% more accurate, which meant that an additional average of 219.2 (out of 6,643) documents were being correctly classified. When IG was used, the difference rose to 6.3%; corresponding to an additional average of 418.5 (out of 6,643) documents being correctly classified. When compared to the best phrase selection strategy in the TFPC algorithm, the higher of the two RL + BestPosRule results was 18.8% more accurate, which translated to an additional average of 1,248.9 (out of 6,643) documents being correctly classified by RL + BestPosRule.

Overall, the RL strategies that did not generate rules with negation performed better with respect to the multi-class classification of the Reuters-8 dataset, regardless

of the feature selection technique or text representation used. When compared to the TFPC algorithm, all the RL strategies performed better.

D.2.2 Results for Using the FP-2 Representation

Strategies	F_1	Average # of rules	Average # of rules with negation	% of rules with negation
χ^2				
RL + UP	0.878	1235.2	0.0	0.0
RL + UN	0.876	937.9	102.5	10.9
RL + Ov	0.879	1110.3	0.0	0.0
RL + UP-UN-Ov	0.878	1235.2	8.1	0.7
RL + UN-UP-Ov	0.875	986.6	105.6	10.7
RL + BestStrategy	**0.880**	951.2	99.8	10.5
RL + BestPosRule	0.878	970.7	0.0	0.0
RL + BestRule	**0.880**	951.0	117.7	12.4
IG				
RL + UP	0.754	1139.5	0.0	0.0
RL + UN	0.816	855.0	143.6	16.8
RL + Ov	0.810	1003.3	0.0	0.0
RL + UP-UN-Ov	0.753	1139.5	16.1	1.4
RL + UN-UP-Ov	0.806	903.2	151.3	16.8
RL + BestStrategy	**0.816**	868.1	104.8	12.1
RL + BestPosRule	0.815	904.4	0.0	0.0
RL + BestRule	**0.816**	868.1	163.2	18.8

Table D.13: Micro-averaged F_1-measure and the average number of rules for the classification of the Reuters-8 dataset using χ^2 and IG for feature selection and the FP-2 representation

Techniques	χ^2	Rank	IG	Rank
FP-2				
RL	0.880	1	0.816	4
SMO	0.876	2	0.824	2
NB	0.672	6	0.675	6
JRip	0.772	5	0.757	5
OlexGreedy	0.847	4	0.819	3
OlexGA	0.858	3	0.831	1

Table D.14: Micro-averaged F_1-measure for the classification of the Reuters-8 dataset using χ^2 and IG for feature selection for the best RL strategy in comparison with the other machine learning techniques using the FP-2 representation

Strategies	Average Acc (%)	Average # of rules	Average # of rules with negation	% of rules with negation
χ^2				
RL + UP	83.4	1235.2	0.0	0.0
RL + UN	82.9	937.9	102.5	10.9
RL + Ov	**85.0**	1110.3	0.0	0.0
RL + UP-UN-Ov	83.4	1235.2	8.1	0.7
RL + UN-UP-Ov	82.6	986.6	105.6	10.7
RL + BestStrategy	83.2	951.2	99.8	10.5
RL + BestPosRule	83.7	970.7	0.0	0.0
RL + BestRule	83.2	951.0	117.7	12.4
IG				
RL + UP	71.1	1139.5	0.0	0.0
RL + UN	75.8	855.0	143.6	16.8
RL + Ov	**77.1**	1003.3	0.0	0.0
RL + UP-UN-Ov	**77.1**	1139.5	16.1	1.4
RL + UN-UP-Ov	74.1	903.2	151.3	16.8
RL + BestStrategy	76.3	868.1	104.8	12.1
RL + BestPosRule	76.7	904.4	0.0	0.0
RL + BestRule	76.1	868.1	163.2	18.8

Table D.15: Average accuracy and the average number of rules for the classification of the Reuters-8 dataset using χ^2 and IG for feature selection and the FP-2 representation

D.3 Experiments Using the SAVSNET Dataset

D.3.1 Results and Analysis for Multi-Class Classification

The evaluation of the eight strategies in the proposed IRL mechanism for the multi-class classification of the SAVSNET dataset is discussed in this sub-section. Table D.16 shows the average accuracy obtained and the average number of rules generated with respect to the SAVSNET dataset for the eight strategies in the proposed IRL mechanism using χ^2 and IG and the FP-1 representation. Table D.17 shows the same results but using the mixed representation in comparison to the TFPC algorithm with its four phrase selection strategies.

Strategies	Average Acc (%)	Average # of rules	Average # of rules with negation	% of rules with negation
χ^2				
RL + UP	76.5	129.2	0.0	0.0
RL + UN	77.9	123.2	4.1	3.3
RL + Ov	77.8	126.7	0.0	0.0
RL + UP-UN-Ov	76.5	129.2	0.0	0.0
RL + UN-UP-Ov	76.5	128.9	5.0	3.9
RL + BestStrategy	**78.6**	124.3	2.3	1.9
RL + BestPosRule	78.2	124.2	0.0	0.0
RL + BestRule	77.8	124.0	4.1	3.3
IG				
RL + UP	75.0	137.2	0.0	0.0
RL + UN	77.0	125.1	7.7	6.2
RL + Ov	77.2	131.5	0.0	0.0
RL + UP-UN-Ov	75.0	137.2	0.9	0.7
RL + UN-UP-Ov	74.4	134.1	8.6	6.4
RL + BestStrategy	**77.4**	125.9	2.9	2.3
RL + BestPosRule	**77.4**	126.1	0.0	0.0
RL + BestRule	76.5	125.7	5.9	4.7

Table D.16: Average accuracy and the average number of rules for the classification of the SAVSNET dataset using χ^2 and IG for feature selection and the FP-1 representation

When χ^2 was used with the FP-1 representation, the best RL strategy was RL + BestStrategy, as shown in Table D.16. This RL strategy generated rules with negation. RL + UP, RL + UN and RL + UP-UN-Ov were equally the worst. The difference between the best and worst RL strategies was 2.1%, which translated to an additional average of 17.4 (out of 828) documents being correctly classified by the best RL strategy. When IG was used, RL + BestStrategy and RL + BestPosRule recorded equal best results, although RL + BestStrategy could be considered better because it produced a slightly smaller ruleset. The worst RL strategy was RL + UN-UP-Ov, which was 3.0%

less accurate or which, on average, correctly classified 24.8 (out of 828) fewer documents compared to the best RL strategy. It was observed that the use of χ^2 enabled the RL strategies to achieve higher average accuracies compared to the use of IG.

Strategies	Average Acc (%)	Average # of rules	Average # of rules with negation	% of rules with negation
		χ^2		
RL + UP	93.1	91.5	0.0	0.0
RL + UN	91.1	64.5	18.7	29.0
RL + Ov	**93.6**	77.9	0.0	0.0
RL + UP-UN-Ov	93.1	91.5	0.0	0.0
RL + UN-UP-Ov	91.1	65.2	18.7	28.7
RL + BestStrategy	92.3	67.8	12.8	18.9
RL + BestPosRule	93.1	77.9	0.0	0.0
RL + BestRule	93.0	68.4	19.3	28.2
		IG		
RL + UP	92.5	85.9	0.0	0.0
RL + UN	61.5	57.7	16.7	28.9
RL + Ov	92.5	87.1	0.0	0.0
RL + UP-UN-Ov	92.5	85.9	0.0	0.0
RL + UN-UP-Ov	61.4	58.8	16.7	28.4
RL + BestStrategy	92.9	69.8	10.0	14.3
RL + BestPosRule	**93.8**	82.9	0.0	0.0
RL + BestRule	93.1	69.7	19.3	27.7
TFPC-DelSN_contGO	38.4	1614.0	0.0	0.0
TFPC-DelSN_contGW	41.7	1541.1	0.0	0.0
TFPC-DelSO_contGN	39.5	1742.2	0.0	0.0
TFPC-DelSO_contGW	43.0	1587.6	0.0	0.0

Table D.17: Average accuracy and the average number of rules for the classification of the SAVSNET dataset using χ^2 and IG for feature selection and the mixed representation

When the mixed representation was used, the RL strategies that did not generate any rules with negation recorded the best results, as shown in Table D.17. Using χ^2, RL + Ov outperformed all the other RL strategies while RL + UN and RL + UN-UP-OV were the worst strategies. The difference in average accuracies between the best and worst RL strategies was 2.5%, whereby RL + Ov correctly classified an additional average of 20.7 (out of 828) documents. RL + BestPosRule was the best RL strategy when IG was used while RL + UN-UP-Ov was the worst. A substantial difference of 32.4% in average accuracy was recorded between these two RL strategies, which translated to an additional average of 268.3 (out of 828) documents being correctly classified by the best RL strategy. When compared to the best phrase selection strategy in the TFPC algorithm, all the RL strategies were substantially better. The best RL

strategy was 50.8% more accurate than the best phrase selection strategy in the TFPC algorithm. This translated to an additional average of 420.6 (out of 828) documents being correctly classified by the best RL strategy.

Overall, the RL strategies that generated rules with negation performed best when the FP-1 representation was used while the RL strategies which did not generate rules with negation were much better when the mixed representation was used. The TFPC algorithm recorded very low accuracies in comparison with the RL strategies.

D.3.2 Results for Using the FP-2 Representation

Strategies	F_1	Average # of rules	Average # of rules with negation	% of rules with negation
χ^2				
RL + UP	0.751	133.2	0.0	0.0
RL + UN	0.752	125.5	2.1	1.7
RL + Ov	0.753	127.9	0.0	0.0
RL + UP-UN-Ov	0.751	133.2	0.9	0.7
RL + UN-UP-Ov	**0.754**	132.0	2.1	1.6
RL + BestStrategy	0.749	127.1	1.9	1.5
RL + BestPosRule	0.750	127.0	0.0	0.0
RL + BestRule	0.750	127.0	2.0	1.6
IG				
RL + UP	0.736	144.1	0.0	0.0
RL + UN	0.750	131.3	6.2	4.7
RL + Ov	**0.752**	135.4	0.0	0.0
RL + UP-UN-Ov	0.736	144.1	1.5	1.0
RL + UN-UP-Ov	0.738	141.7	6.2	4.4
RL + BestStrategy	0.750	133.5	4.4	3.3
RL + BestPosRule	0.750	133.3	0.0	0.0
RL + BestRule	0.750	133.3	6.1	4.6

Table D.18: Micro-averaged F_1-measure and the average number of rules for the classification of the SAVSNET dataset using χ^2 and IG for feature selection and the FP-2 representation

Techniques	χ^2	Rank	IG	Rank
FP-2				
RL	0.754	2	0.752	2
SMO	0.723	4	0.724	4
NB	0.626	6	0.630	6
JRip	0.658	5	0.663	5
OlexGreedy	0.742	3	0.742	3
OlexGA	0.769	1	0.776	1

Table D.19: Micro-averaged F_1-measure for the classification of the SAVSNET dataset using χ^2 and IG for feature selection for the best RL strategy in comparison with the other machine learning techniques using the FP-2 representation

Strategies	Average Acc (%)	Average # of rules	Average # of rules with negation	% of rules with negation
χ^2				
RL + UP	76.1	133.2	0.0	0.0
RL + UN	75.8	125.5	2.1	1.7
RL + Ov	76.0	127.9	0.0	0.0
RL + UP-UN-Ov	76.1	133.2	0.9	0.7
RL + UN-UP-Ov	**76.4**	132.0	2.1	1.6
RL + BestStrategy	75.8	127.1	1.9	1.5
RL + BestPosRule	75.9	127.0	0.0	0.0
RL + BestRule	75.8	127.0	2.0	1.6
IG				
RL + UP	72.8	144.1	0.0	0.0
RL + UN	74.1	131.3	6.2	4.7
RL + Ov	74.1	135.4	0.0	0.0
RL + UP-UN-Ov	72.8	144.1	1.5	1.0
RL + UN-UP-Ov	73.2	141.7	6.2	4.4
RL + BestStrategy	74.2	133.5	4.4	3.3
RL + BestPosRule	**74.4**	133.3	0.0	0.0
RL + BestRule	74.2	133.3	6.1	4.6

Table D.20: Average accuracy and the average number of rules for the classification of the SAVSNET dataset using χ^2 and IG for feature selection and the FP-2 representation

Bibliography

[1] R. Agrawal, T. Imielinski, and A. Swami. Mining association rules between sets of items in large databases. In *Proceedings of the ACM SIGMOD Internationalo Conference on Management of Data*, pages 207–216, 1993.

[2] R. Agrawal and R. Srikant. Fast algorithms for mining association rules in large databases. In *Proceedings of the 20th International Conference on Very Large Data Bases*, VLDB '94, pages 487–499, San Francisco, CA, USA, 1994. Morgan Kaufmann Publishers Inc.

[3] M-L. Antonie and O. R. Zaïane. An associative classifier based on positive and negative rules. In *Proceedings of the 9th ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery*, pages 64–69, Paris, France, 2004.

[4] C. Apté, F. J. Damerau, and S. M. Weiss. Automated learning of decision rules for text categorization. *ACM Transactions on Information Systems*, 12(3):233–251, 1994.

[5] J. Bakus and M. Kamel. Document classification using phrases. In T. Caelli, A. Amin, R. Duin, D. de Ridder, and M. Kamel, editors, *Structural, Syntactic, and Statistical Pattern Recognition*, volume 2396, pages 341–354. Springer Berlin / Heidelberg, 2002.

[6] E. Baralis and P. Garza. Associative text categorization exploiting negated words. In *Proceedings of the ACM Symposium on Applied Computing*, pages 530–535, 2006.

[7] J. Blitzer, M. Dredze, and F. Pereira. Biographies, bollywood, boom-boxes and blenders: Domain adaptation for sentiment classification. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics*, Prague, Czech Republic, June 2007.

[8] A. Z. Broder. On the resemblance and containment of documents. In *Proceedings of Compression and Complexity of Sequences (SEQUENCES'97)*, pages 21–29, 1997.

[9] C. Brunk and M. Pazzani. Noise-tolerant relational concept learning algorithms. In *Proceedings of the 8th International Workshop on Machine Learning*, Ithaca, New York, 1991. Morgan Kaufmann.

[10] P. Calado, M. Cristo, E. S. D. Moura, N. Ziviani, B. A. Ribeiro-Neto, and M. A. Gonçalves. Combining link-based and content-based methods for web document classification. In *Proceedings of the12th ACM International Conference on Information and Knowledge Management (CIKM)*, pages 394–401, New Orleans, US, 2003. ACM Press, New York, US.

[11] M. F. Caropreso, S. Matwin, and F. Sebastiani. A learner-independent evaluation of the usefulness of statistical phrases for automated text categorization. In A. G. Chin, editor, *Text Databases and Document Management: Theory and Practice*, pages 78–102. Idea Group Publishing, 2001.

[12] M. Ceci and D. Malerba. Classifying web documents in a hierarchy of categories: A comprehensive study. *Journal of Intelligent Information Systems*, 28(1):37–78, February 2007.

[13] A. K. M. Chai, H. T. Ng, and H. L. Chieu. Bayesian online classifiers for text classification and filtering. In Micheline Beaulieu, Ricardo Baeza-Yates, Sung Hyon Myaeng, and Kalervo Järvelin, editors, *Proceedings of the 25th ACM International Conference on Research and Development in Information Retrieval*, pages 97–104, Tampere, FI, 2002. ACM Press, New York, US.

[14] M. Chang and C. K. Poon. Using phrases as features in email classification. *Journal of Systems and Software*, 82:1036–1045, June 2009.

[15] S. Chua, F. Coenen, and G. Malcolm. Rule learning with negation: Issues regarding effectiveness. In *Proceedings of the 6th International Conference on Intelligent Information Processing (IIP'10), IFIP*, pages 193–202, 2010.

[16] G. Cleuziou and C. Poudat. On the impact of lexical and linguistic features in genre and domain-based text categorization. In *Proceedings of the 8th International Conference on Intelligent Text Processing and Computational Linguistics*, Mexico City, Mexico, February 2007.

[17] F. Coenen. Some notes and example schema for the LUCS-KDD DN (discretization/normalization software version 2. `http://www.csc.liv.ac.uk/~frans/KDD/Software/LUCS-KDD-DN/exmpleDNnotes.html`, 2008.

[18] F. Coenen. The LUCS-KDD data discretization/normalization (DN) Java software for classification association rule mining version 2. `http://www.csc.liv.ac.uk/~frans/KDD/Software/LUCS-KDD-DN/lucs-kdd_DN.html`, 2011.

[19] F. Coenen and P. Leng. Obtaining best parameter values for accurate classification. In *Proceedings of the IEEE International Conference on Data Mining (ICDM)*, pages 597–600, 2005.

[20] F. Coenen, P. Leng, R. Sanderson, and Y. J. Wang. Statistical identification of key phrases for text classification. In *Proceedings of the International Conference on Machine Learning and Data Mining (MLDM)*, pages 838–853. Springer, 2007.

[21] F. Coenen, P. Leng, and L. Zhang. Threshold tuning for improved classifcation association rule mining. In *Proceedings of the Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD)*, pages 216–225. Springer, 2005.

[22] W. Cohen. Fast effective rule induction. In *Proceedings of the 12th International Conference on Machine Learning (ICML)*, pages 115–123, San Francisco, CA, 1995. Morgan Kaufmann.

[23] W. W. Cohen, V. R. Carvalho, and T. M. Mitchell. Learning to classify email into "speech acts". In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 309–316, Barcelona, Spain, July 2004. Association for Computational Linguistics.

[24] F. Crestani. Learning strategies for an adaptive information retrieval system using neural networks. In *Proceedings of the IEEE International Conference on Neural Networks*, San Francisco, CA, 1993.

[25] H. Cui, V. Mittal, and M. Datar. Comparative experiments on sentiment classification for online product reviews. In *Proceedings of the 21st National Conference on Artificial Intelligence*, pages 1265–1270, July 2006.

[26] S. J. Cunningham, J. Littin, and I. H. Witten. Applications of machine learning in information retrieval. Technical report, Computer Science Department, University of Waikato, Waikato, New Zealand, 1997.

[27] Y. Diao, H. Lu, and D. Wu. A comparative study of classification-based personal e-mail filtering. In Takao Terano, Huan Liu, and Arbee L.P. Chen, editors, *Proceedings of the 4th Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 408–419, Kyoto, JP, 2000. Springer Verlag, Heidelberg, DE. Published in the "Lecture Notes in Computer Science" series, number 1805.

[28] S. Dumais, J. Platt, D. Heckerman, and M. Sahami. Inductive learning algorithms and representations for text categorization. In *Proceedings of the 7th International Conference on Information and Knowledge Management*, CIKM '98, pages 148–155, New York, NY, USA, 1998. ACM.

[29] S. M. Z. Eissen and B. Stein. Genre classification of web pages. In Susanne Biundo, Thom Frühwirth, and Günther Palm, editors, *Proceedings of the 27th German Conference on Artificial Intelligence*, Ulm, DE, 2004. Published in the "Lecture Notes in Computer Science" series, number 3238.

[30] U. Fayyad, G. Piatetsky-Shapiro, and P. Smyth. From data mining to knowledge discovery in databases. In *AI Magazine*, volume 17, pages 37–54, 1996.

[31] U. Fayyad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy, editors. *Advances in Knowledge Discovery in Databases*. AAAI/MIT Press, 1996.

[32] A. Finn, N. Kushmerick, and B. Smyth. Genre classification and domain transfer for information filtering. In Fabio Crestani, Mark Girolami, and Cornelis J. Van Rijsbergen, editors, *Proceedings of the 24th European Colloquium on Information Retrieval Research*, pages 353–362, Glasgow, UK, 2002. Springer Verlag, Heidelberg, DE. Published in the "Lecture Notes in Computer Science" series, number 2291.

[33] A. Frank and A. Asuncion. UCI Machine Learning Repository, 2010.

[34] J. Fürnkranz. A study using n-gram features for text categorization. Technical report, Austrian Research Institute for Artificial Intelligence, 1998.

[35] J. Fürnkranz, T. Mitchell, and E. Riloff. A case study in using linguistic phrases for text categorization on the WWW. In *Working Notes of the AAAI/ICML Workshop on Learning for Text Categorization*, pages 5–12. AAAI Press, 1998.

[36] J. Fürnkranz and G. Widmer. Incremental reduced error pruning. In *Proceedings of the 11th International Conference on Machine Learning (ICML)*, New Brunswich, New Jersey, 1994. Morgan Kaufmann.

[37] L. Galavotti, F. Sebastiani, and M. Simi. Experiments on the use of feature selection and negative evidence in automated text categorization. In *Proceedings of the 4th European Conference on Research and Advanced Technology for Digital Libraries*, pages 59–68, 2000.

[38] M. Gamon. Sentiment classification on customer feedback data: Noisy data, large feature vectors, and the role of linguistic analysis. In *Proceedings of Coling 2004*, pages 841–847, Geneva, Switzerland, August 2004. COLING.

[39] M. F. García-Constantino. Overview of the data used in the questionnaire trend mining research. Technical report, Department of Computer Science, University of Liverpool, 2011.

[40] J. Han and M. Kamber. *Data Mining: Concepts and Techniques*. Morgan Kaufmann, 2006.

[41] P. J. Hayes and S. P. Weinstein. CONSTRUE-TIS: A system for content-based indexing of a database of news stories. In *Proceedings of the 2nd Conference on Innovative Applications of Artificial Intelligence (IAAI)*, pages 49–66, Menlo Park, 1990. AAAI Press.

[42] G. Holmes and L. Trigg. A diagnostic tool for tree based supervised classification learning algorithms. In *Proceedings of the 6th International Conference on Neural Information Processing (ICONIP)*, volume II, pages 514–519, Perth, Western Australia, 1999.

[43] T. Joachims. Text categorization with support vector machines: Learning with many relevant features. In *Proceedings of the 10th European Conference on Machine Learning (ECML)*, pages 137–142, 1998.

[44] T. Joachims. Transductive inference for text classification using support vector machines. In Ivan Bratko and Saso Dzeroski, editors, *Proceedings of the16th International Conference on Machine Learning (ICML)*, pages 200–209, Bled, SL, 1999. Morgan Kaufmann Publishers, San Francisco, US.

[45] T. Joachims. Estimating the generalization performance of a SVM efficiently. In Pat Langley, editor, *Proceedings of the17th International Conference on Machine Learning (ICML)*, pages 431–438, Stanford, US, 2000. Morgan Kaufmann Publishers, San Francisco, US.

[46] T. Joachims. Training linear SVMs in linear time. In *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 217–226, 2006.

[47] D. E. Johnson, F. J. Oles, T. Zhang, and T. Goetz. A decision-tree-based symbolic rule induction system for text categorization. *The IBM Systems Journal, Special Issue on AI*, 41:428–437, 2002.

[48] A. Kolcz and A. Chowdhury. Avoidance of model re-induction in SVM-based feature selection for text categorization. In *Proceedings of the 20th International Joint Conference on Artifical Intelligence*, IJCAI'07, pages 889–894, San Francisco, CA, USA, 2007. Morgan Kaufmann Publishers Inc.

[49] M. Kongovi, J. C. Guzman, and V. Dasigi. Text categorization: An experiment using phrases. In Fabio Crestani, Mark Girolami, and Cornelis J. Van Rijsbergen, editors, *Proceedings of the 24th European Colloquium on Information Retrieval*

Research, pages 213–228, Glasgow, UK, 2002. Springer Verlag, Heidelberg, DE. Published in the "Lecture Notes in Computer Science" series, number 2291.

[50] K. Lang. Newsweeder: Learning to filter netnews. In *Proceedings of the 12th International Conference on Machine Learning*, pages 331–339, 1995.

[51] P. Langley, W. Iba, and K. Thompson. An analysis of Bayesian classifiers. In *Proceedings of the 10th National Conference on Artificial Intelligence*, pages 223–228. MIT Press, 1992.

[52] Y-B. Lee and S. H. Myaeng. Text genre classification with genre-revealing and subject-revealing features. In Micheline Beaulieu, Ricardo Baeza-Yates, Sung Hyon Myaeng, and Kalervo Järvelin, editors, *Proceedings of the 25th ACM International Conference on Research and Development in Information Retrieval*, pages 145–150, Tampere, FI, 2002. ACM Press, New York, US.

[53] D. D. Lewis. An evaluation of phrasal and clustered representations on a text categorization task. In *Proceedings of the 15th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, 1992.

[54] D. D. Lewis. Feature selection and feature extraction for text categorization. In *Proceedings of the Speech and Natural Language Workshop*, pages 212–217. Morgan Kaufman, 1992.

[55] D. D. Lewis. *Representation and learning in information retrieval.* PhD thesis, Computer Science Department, University of Massachusetts, Amherst, MA, 1992.

[56] D. D. Lewis. Reuters-21578 text categorization test collection, Distribution 1.0, README file (v 1.3), 2004.

[57] D. D. Lewis and M. Ringuette. A comparison of two learning algorithms for text categorization. In *Proceedings of the 3rd Annual Symposium on Document Analysis and Information Retrieval*, pages 81–93, 1994.

[58] W. Li, J. Han, and J. Pei. CMAR: Accurate and efficient classification based on multiple class-association rules. In *Proceedings of the IEEE International Conference on Data Mining*, pages 369–376, 2001.

[59] Z. Li, P. Li, W. Wei, H. Liu, J. He, T. Liu, and X. Du. AutoPCS: A phrase-based text categorization system for similar texts. In Qing Li, Ling Feng, Jian Pei, Sean Wang, Xiaofang Zhou, and Qiao-Ming Zhu, editors, *Advances in Data and Web Management*, volume 5446 of *Lecture Notes in Computer Science*, pages 369–380. Springer Berlin / Heidelberg, 2009.

[60] B. Liu, W. Hsu, and Y. Ma. Integrating classification and association rule mining. In *Proceedings of the 4th International Conference on Knowledge Discovery and Data Mining*, pages 80–86, 1998.

[61] O. L. Mangasarian and W. H. Wolberg. Cancer diagnosis via linear programming. *SIAM News*, 23(5):1–18, 1990.

[62] T. Masuyama and H. Nakagawa. Applying cascaded feature selection to SVM text categorization. In *Proceedings of the DEXA Workshops*, pages 241–245, 2002.

[63] A. McCallum and K. Nigam. A comparison of event model for Naive Bayes text classification. In *Proceedings of the AAAI-98 Workshop on Learning for Text Categorization*, pages 41–48, 1998.

[64] A. Mesleh. Chi square feature extraction based SVMs Arabic language text categorization system. *Journal of Computer Science*, 3(6):430–435, 2007.

[65] A. Mesleh. Support vector machines based Arabic language text classification system: Feature selection comparative study. In Tarek Sobh, editor, *Advances in Computer and Information Sciences and Engineering*, pages 11–16. Springer Netherlands, 2008.

[66] G. A. Miller, R. Beckwith, C. Fellbaum, D. Gross, and K. J. Miller. Introduction to Wordnet: An on-line lexical database. *International Journal of Lexicography*, 3(4):235–244, 1990.

[67] D. Mladenić and M. Grobelnik. Word sequences as features in text-learning. In *Proceedings of the 17th Electrotechnical and Computer Science Conference*, pages 145–148, 1998.

[68] D. Mladenić and M. Grobelnik. Feature selection on hierarchy of web documents. *Decision Support Systems*, 35(1):45–87, 2003.

[69] P. M. Murphy and D. W. Aha. UCI repository of machine learning databases. University of California, Department of Information and Computer Science., 1992.

[70] A. Pietramala, V. Policicchio, P. Rullo, and I. Sidhu. A genetic algorithm for text classification rule induction. In Walter Daelemans, Bart Goethals, and Katharina Morik, editors, *Machine Learning and Knowledge Discovery in Databases*, volume 5212 of *Lecture Notes in Computer Science*, pages 188–203. Springer Berlin / Heidelberg, 2008.

[71] J. R. Quinlan. Induction of decision trees. *Machine Learning*, 1(1):81–106, 1986.

[72] J. R. Quinlan. *C4.5: Programs for machine learning*. Morgan Kaufman, 1993.

[73] J. R. Quinlan. MDL and categorical theories (continued). In *Proceedings of the 12th International Conference on Machine Learning (ICML)*, pages 464–470, San Francisco, CA, 1995. Morgan Kaufmann.

[74] A. Radford, A. Tierney, K. P. Coyne, R. M. Gaskell, P. J. Noble, S. Dawson, C. Setzkorn, P. H. Jones, I. E. Buchan, J. R. Newton, and J. G. Bryan. Developing a network for small animal disease surveillance. *Veterinary Record*, 167:472–474, 2010.

[75] M. Radovanović and M. Ivanović. Document representations for classification of short web-page descriptions. In *Proceedings of the 8th International Conference on Data Warehousing and Knowledge Discovery (DaWaK)*, volume 4081 of *Lecture Notes in Computer Science*, pages 544–553, Krakow, Poland, 2006. Springer-Verlag.

[76] M. Radovanović and M. Ivanović. Interactions between document representation and feature selection in text categorization. In S. Bressan, J. Küng, and R. Wagner, editors, *Database and Expert Systems Applications*, volume 4080 of *Lecture Notes in Computer Science*, pages 489–498. Springer Berlin / Heidelberg, 2006.

[77] M. E. Ruiz and P. Srinivasan. Hierarchical text categorization using neural networks. *Information Retrieval*, 5(1):87–118, 2002.

[78] P. Rullo, C. Cumbo, and V. L Policicchio. Learning rules with negation for text categorization. In *Proceedings of the 2007 ACM Symposium on Applied Computing*, pages 409–416, Seoul, Korea, 2007. ACM.

[79] P. Rullo, V. Policicchio, C. Cumbo, and S. Iiritano. Olex: Effective rule learning for text categorization. *IEEE Transactions on Knowledge and Data Engineering*, 21:1118–1132, 2009.

[80] S. Scott and S. Matwin. Feature engineering for text classification. In *Proceedings of the 16th International Conference on Machine Learning (ICML)*, pages 379–388, 1999.

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

[82] A. Selamat and S. Omatu. Web page feature selection and classification using neural networks. *Information Sciences*, 158(1):69–88, 2004.

[83] R. Tailby, R. Dean, B. Milner, and D. Smith. Email classification for automated service handling. In *Proceedings of the 21st Annual ACM Symposium on Applied Computing*, Dijon, France, April 2006.

[84] C.-M. Tan, Y.-F. Wang, and C.-D. Lee. The use of bigrams to enhance text categorization. *Journal of Information Processing and Management*, 38(4):529–546, July 2002.

[85] P. Thompson. Automatic categorization of case law. In *Proceedings of the 8th International Conference on Artificial Intelligence and Law*, pages 70–77, St. Louis, US, 2001. ACM Press, New York, US.

[86] P. D. Turney. Learning to extract keyphrases from text. Technical report, National Research Council of Canada, Institute of Information Technology, 1999.

[87] V. N. Vapnik. *The Nature of Statistical Learning Theory*. Springer-Verlag, 1995.

[88] F. Vasile, A. Silvescu, D-K. Kang, and V. Honavar. TRIPPER: Rule learning using taxonomies. In *Proceedings of the 10th Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD)*, pages 55–59, Singapore, 2006.

[89] Y. J. Wang. *Language-independent pre-processing of large documentbases for text classifcation*. PhD thesis, Department of Computer Science, University of Liverpool, 2008.

[90] Y. J. Wang, Q. Xin, and F. P. Coenen. Hybrid rule ordering in classification association rule mining. *Transactions on Machine Learning and Data Mining in Pattern Recognition*, 1(1):1–16, 2008.

[91] S. M. Weiss and N. Indurkhya. Optimized rule induction. *IEEE Expert: Intelligent Systems and Their Applications*, 8(6):61–69, 1993.

[92] I. H. Witten, E. Frank, and M. A. Hall. *Data Mining: Practical Machine Learning Tools and Techniques, 3rd Edition*. Morgan Kaufmann, 2011.

[93] X. Wu, C. Zhang, and S. Zhang. Mining both positive and negative association rules. In *Proceedings of the 19th International Conference on Machine Learning*, pages 658–665, Sydney, Australia, 2002.

[94] Y. Yang and X. Liu. A re-examination of text categorization methods. In *Proceedings of SIGIR-99, 22nd ACM International Conference on Research and Development in Information Retrieval*, pages 42–49, Berkeley, CA, 1999.

[95] Y. Yang and J. Pedersen. A comparative study on feature selection in text categorization. In *Proceedings of the 14th International Conference on Machine Learning (ICML)*, pages 412–420, Nashville, TE, USA, 1997.

[96] X. Yin and J. Han. CPAR: Classification based on predictive association rules. In *Proceedings of the 3rd SIAM International Conference on Data Mining*, pages 331–335, San Francisco, CA, 2003.

[97] Y. Yoon and G. G. Lee. Text categorization based on boosting association rules. In *Proceedings of the IEEE International Conference on Semantic Computing*, pages 136 –143, aug. 2008.

[98] W. Zaghloul, S. M. Lee, and S. Trimi. Text classification: Neural networks vs support vector machines. *Industrial Management & Data Systems*, 109(5):708–717, 2009.

[99] D. Zhang and W. S. Lee. Question classification using support vector machines. In Jamie Callan, Gordon Cormack, Charles Clarke, David Hawking, and Alan Smeaton, editors, *Proceedings of the 26th ACM International Conference on Research and Development in Information Retrieval*, pages 26–32, Toronto, CA, 2003. ACM Press, New York, US.

[100] Z. Zheng. Feature selection for text categorization on imbalanced data. *ACM SIGKDD Explorations Newsletter*, 6:2004, 2004.

[101] Z. Zheng and R. Srihari. Optimally combining positive and negative features for text categorization. In *Proceedings of the International Conference on Machine Learning (ICML), Workshop on Learning from Imbalanced Datasets II*, Washington D. C., 2003.