Knowledge-Enhanced Text Classification: Descriptive Modelling and New Approaches

Miguel Martinez-Alvarez

University of London

Thesis submitted for the degree of Doctor of Philosophy

at Queen Mary, University of London

June 2014

Declaration of originality

I, Miguel Martinez-Alvarez, confirm that the research included within this thesis is my own work or that where it has been carried out in collaboration with, or supported by others, that this is duly acknowledged below and my contribution indicated. Previously published material is also acknowledged below.

I attest that I have exercised reasonable care to ensure that the work is original, and does not to the best of my knowledge break any UK law, infringe any third party's copyright or other Intellectual Property Right, or contain any confidential material.

I accept that the College has the right to use plagiarism detection software to check the electronic version of the thesis.

I confirm that this thesis has not been previously submitted for the award of a degree by this or any other university.

Signature: Miguel Martinez-Alvarez

Date: 13 January 2014

Some parts of this work have been previously published as:

- Martinez-Alvarez, M. and Roelleke, T. (2013). Mathematical Specification and Logic Modelling in the context of IR. In *Proceedings of the 4th International Conference on Theory on Information Retrieval (ICTIR)*. Springer

- Roelleke, T., Bonzanini, M., and Martinez-Alvarez, M. (2013b). On the Modelling of Ranking Algorithms in Probabilistic Datalog. In *Proceedings of the 7th International Workshop on Ranking in Databases (DBRank)*. ACM

- Martinez-Alvarez, M., Bellogin, A., and Roelleke, T. (2013). Document Difficulty Framework for Semi-Automatic Text Classification. In *Proceedings of the 15th International Conference on Data Warehousing and Knowledge Discovery (DaWak)*. ACM

- Martinez-Alvarez, M., Yahyaei, S., and Roelleke, T. (2012). Semi-automatic Document Classification: Exploiting Document Difficulty. In *Proceedings of the 34th European Conference on Information Retrieval (ECIR)*, volume 7224 of *Lecture Notes in Computer Science*, pages 468–471. Springer
- Martinez-Alvarez, M. (2011). Descriptive modelling of text classification and its integration with other IR tasks. In *Proceedings of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, pages 1317–1318. ACM
- Martinez-Alvarez, M. and Roelleke, T. (2011). A Descriptive Approach to Classification. In *Proceedings of the 3rd International Conference on the Theory of Information Retrieval (ICTIR)*, volume 6931 of *Lecture Notes in Computer Science*, pages 297–308. Springer
- Smeraldi, F., Martinez-Alvarez, M., Frommholz, I., and Roelleke, T. (2011). On the Probabilistic Logical Modelling of Quantum and Geometrically-Inspired IR. In *Proceedings of the 2nd Italian Information Retrieval Workshop (IIR)*, volume 704 of *CEUR Workshop Proceedings*. CEUR-WS.org
- Martinez-Alvarez, M. and Roelleke, T. (2010). Modelling Probabilistic Inference Networks and Classification in Probabilistic Datalog. In *Proceedings on the 4th International Conference on Scalable Uncertainty Management (SUM)*, volume 6379 of *Lecture Notes in Computer Science*, pages 278–291. Springer
- Martinez-Alvarez, M., Smeraldi, F., and Roelleke, T. (2010). A Descriptive Approach to Modelling Learning. In *Proceedings of the 1st Spanish Conference on Information Retrieval (CERI)*, pages 183–194

Other publications and collaborations:

- Roelleke, T., Azzam, H., Bonzanini, M., Martinez-Alvarez, M., and Lalmas, M. (2013a). The D2Q2 Framework: On the Relationship and Combination of Language Modelling and TF-IDF. In *Proceedings of the Workshop-Woche: Lernen, Wissen & Adaptivitaet (LWA)*.
- Bonzanini, M., Martinez-Alvarez, M., and Roelleke, T. (2013). Extractive Summarisation via Sentence Removal: Condensing Relevant Sentences into a Short Summary. In *Proceedings of the 36th International SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*. ACM
- Bonzanini, M., Martinez-Alvarez, M., and Roelleke, T. (2012b). Opinion summarisation through sentence extraction: an investigation with movie reviews. In *Proceedings of the*

35th International SIGIR Conference on Research and Development in Information Retrieval (SIGIR), pages 1121–1122. ACM

- Bonzanini, M., Martinez-Alvarez, M., and Roelleke, T. (2012a). Investigating the Use of Extractive Summarisation in Sentiment Classification. In *Proceedings of the 3rd Italian Information Retrieval Workshop (IIR)*, volume 835 of *CEUR Workshop Proceedings*, pages 45–52. CEUR-WS.org

Abstract

The knowledge available to be exploited by text classification and information retrieval systems has significantly changed, both in nature and quantity, in the last years. Nowadays, there are several sources of information that can potentially improve the classification process, and systems should be able to adapt to incorporate multiple sources of available data in different formats. This fact is specially important in environments where the required information changes rapidly, and its utility may be contingent on timely implementation. For these reasons, the importance of adaptability and flexibility in information systems is rapidly growing. Current systems are usually developed for specific scenarios. As a result, significant engineering effort is needed to adapt them when new knowledge appears or there are changes in the information needs.

This research investigates the usage of knowledge within text classification from two different perspectives. On one hand, the application of descriptive approaches for the seamless modelling of text classification, focusing on knowledge integration and complex data representation. The main goal is to achieve a scalable and efficient approach for rapid prototyping for Text Classification that can incorporate different sources and types of knowledge, and to minimise the gap between the mathematical definition and the modelling of a solution.

On the other hand, the improvement of different steps of the classification process where knowledge exploitation has traditionally not been applied. In particular, this thesis introduces two classification sub-tasks, namely Semi-Automatic Text Classification (SATC) and Document Performance Prediction (DPP), and several methods to address them. SATC focuses on selecting the documents that are more likely to be wrongly assigned by the system to be manually classified, while automatically labelling the rest. Document performance prediction estimates the classification quality that will be achieved for a document, given a classifier. In addition, we also propose a family of evaluation metrics to measure degrees of misclassification, and an adaptive variation of *k*-NN.

Contents

List of Figures

List of Tables

Acknowledgements

The completion of this thesis has been a long and hard process that could have not being accomplished without the support and encouragement of several people to whom I am deeply grateful.

Firstly, I am greatly in debt, both in a personal and professional level, to my supervisor Thomas Roelleke, for his unconditional support, and for all the opportunities and experiences he has provided me with. Also, the lectures and colleagues at QMUL: Anastasios Tombros, Fabrizio Smeraldi, Hengzy Wu, Hany Azzam, Sirvan Yahyaei, Marco Bonzanini, Nuzhah Gooda Sahib, Yihan Tao, and Kayras Bhesania. Their advice has been very important in my development as a researcher, and their input is heavily reflected in my research. Moreover, the pool sessions and occasional beer I shared with some of them probably saved me from insanity. Also, I cannot but recognise all the support, knowledge and friendship of Alejandro Bellogin, who has always had time to discuss all my ideas, despite his busy schedule.

This thesis would have been very different without my interactions with the industrial world. I would like to thank all the people I worked with at Globe Business Publishers, especially Chris Proud, for the great opportunity and all the moments (and bottles of tequila) we have shared. Furthermore, my colleagues at Signal: David Benigson and Wesley Hall for their understanding and flexibility while I was simultaneously working with them and finishing this thesis.

Last, but no least, I thank the people closer to me: My grandfather Jesus Alvarez Alonso for his lessons, and for being one of my inspirations even after he passed away. My family in general, and my parents in particular, for their support and guidance during the long periods without visiting home, and for their unbreakable faith on me. And, to my marvellous girlfriend Varvara Vladimirova, who suffered me the most and endured my long working days (and nights) and the toughest moments without any complain, always supporting me.

Chapter 1

Introduction

1.1 Motivation

Available data has importantly changed in the last years, both in quantity and quality. Its amount and velocity (i.e., how quickly new data is created) have significantly increased, both in the Web and business environments. As a result, there has been an explosion of new possibilities of how this information can be exploited. At the same time, such data, or at least our representation of it, has become more complex and richer. This includes new types of information such as entities with fields or attributes and relationships, instead of just individual textual elements. This approach adds more complexity to the processing of information, but it represents better the original data.

Several models have been proposed to exploit knowledge for classification, mainly for data augmentation [Gabrilovich and Markovitch, 2006]. However, *"even if some algorithms have achieved improvements, a consistent and significant quality increasing has not been obtained"* [Wang et al., 2007]. Moreover, we claim that a common framework to explain and integrate such techniques is also missing. In addition, the vast majority of knowledge-enhanced techniques in text classification focus on data representation, while other steps such as evaluation have not been explored in detail.

Exploiting complex entities and relational data presents a challenge for current classification systems, especially if multiple types of data are allowed. It is also becoming clearer that different tasks can be applied together in order to improve each other (e.g., Learning to rank [Liu, 2009]), and that there is a large overlap between their foundations [Belkin and Croft, 1992] [Bellogín et al., 2012]. Despite the possibilities of exploiting this richer information, representations based on a *bag of words*, where each document is represented as the set of terms that it contains, are still the most common approach for data representation in text classification and information retrieval. Another factor is that if the assumption that all the data has the same nature (e.g., textual documents) is not applied (i.e., heterogeneous environments), most of the traditional techniques for information retrieval and machine learning cannot be directly applied, or they achieve poor results [Lu and Getoor, 2003]. One of the main strategies to address this challenge is to transform the different types of objects into a common representation, for instance using a textual description for each element. However, quality will probably decrease because information is lost in the process. The consideration of these types of complex information narrows the gap between different fields: Textual data has traditionally been exploited in information retrieval and text classification, while relations and linked data are mainly analysed using data mining and database techniques.

A "perfect" information system, as defined by [Abrol et al., 2001], should solve *"the need to seamlessly and scalably combine structured (e.g., relational) as well as unstructured information in a document for search, as well as for organisational purposes (clustering, classification, etc.) and for personalisation"*. Nowadays, the integration or adaptation of methods between tasks is difficult to achieve given the current design of information systems. The main reason for this challenge is that information systems are usually developed for specific cases, therefore, leading to rewrite large portions of the original code for other purposes. The majority of these systems commonly involve complicated knowledge transfer and maintainability processes. Moreover, large amounts of code are produced, that are usually difficult to understand, use and adapt for new developers or researchers. This situation can be compared to what happened in the software industry, when software engineering evolved from programs focused on one specific context, to the development of frameworks for general tasks that could be adapted for specific ones. Furthermore, due to their particular nature, information systems can be seen as prototype-based environments, where new models or data need to be quickly tested. The reason for this is that decisions have to be made according to the analysis of dynamic data and requirements, where different sources of ambiguous or even contradictory information need to be combined. Therefore, productivity and fast changeability are important characteristics. All these factors support the claim that the importance of flexibility in information retrieval systems is rapidly growing, and that traditional architectures are too rigid to allow quick modifications [Cornacchia and de Vries, 2007]. In addition, too much engineering effort is needed to adapt them when new knowledge appears, or there are changes in the requirements [Hiemstra and Mihajlovic, 2010].

1.2 Research Questions

The research questions this thesis addresses are shown below. All of them are related to the two main concepts of this research: How to improve the modelling of classifiers using descriptive approaches, and how to exploit knowledge throughout the classification process.

1. To what degree is the expressiveness of descriptive approaches, represented by probabilistic Datalog, enough to model and customise traditional classifiers?
2. To what degree can models expressed in a descriptive approach be automatically translated to a mathematical formulation to observe and verify its semantics?
3. Can a formal framework generalise and integrate the main knowledge exploitation techniques for data augmentation and score modification in text classification?
4. To what degree descriptive approaches can seamlessly integrate textual and relational classifiers with heterogeneous data?
5. Is a descriptive approach abstract enough to apply flexible task integration and adaptation?
6. Given a set of documents to be classified and a limited amount of human resources. What quality can be achieved if those resources are optimised to focus only on the documents that the automatic system is more likely to misclassify?, and, are the category thresholds and classification scores useful for this task?
7. With what confidence can a set of documents be ranked according to their expected classification quality?, and, are the category thresholds and classification scores useful to solve this task?
8. Given a set of classifiers and their quality evaluation, will their relative quality or their ranking change using an evaluation metric that includes degrees of misclassification?

9. How can the number of neighbours in a *k*-NN classifier be automatically optimised per category without dividing it into multiple binary classifiers?, and, what quality improvement will be achieved compared to traditional *k*-NN?

1.3 Contributions

The main contributions of this thesis are: Firstly, the seamless modelling of text classification using a descriptive approach, leading to a high-level abstraction that allows a flexible and adaptable environment for prototyping and knowledge exploitation in complex environments. Secondly, several tasks and methods to improve usually neglected aspects of the classification process using diverse knowledge. The specific contributions, listed by chapter, are summarised as follows:

- **Chapter 4: DESCRIPTIVE MODELLING OF TEXT CLASSIFICATION**

  Modelling of the Classification Process A seamless modelling of the text classification process using a descriptive approach. This abstraction includes the definition of a generic data schema and the modelling of different families of classifiers using probabilistic Datalog, namely Naive Bayes and k-Nearest Neighbours.

  Declarative Text Classification Customisation Abstraction for flexible task customisation, using a global predicate dictionary.

  Mathematical Translation of Classifiers Modelling The capability to translate the modelling of a solution, in probabilistic Datalog, to its mathematical definition in order to improve maintainability and model verification.

  Evaluation Empirical confirmation of the quality of the classifiers modelled with probabilistic Datalog. Furthermore, an efficiency study investigates the applicability of the approach.

- **Chapter 5: DESCRIPTIVE MODELLING OF KNOWLEDGE-ENHANCED TEXT CLASSIFICATION**

  Knowledge Exploitation Framework A conceptual definition of the types of knowledge that can be used, and a knowledge exploitation framework for text classification. This also includes the modelling of the framework using probabilistic Datalog.

Combination of Textual and Relational Classifiers The ability to model and combine relational and textual classifiers within heterogeneous environments.

High-level Task Integration High level integration capability between information retrieval tasks and text classification.

Relational and Hybrid Classification Evaluation Quality values for relational and hybrid (relational and content-based) classifiers using descriptive approaches to combine both approaches.

- **Chapter 6: NEW APPROACHES OF KNOWLEDGE EXPLOITATION FOR THE CLASSIFICATION PROCESS**

Semi-Automatic Text Classification The introduction of the task of Semi-Automatic Text Classification (SATC), where the goal is to optimise the available human resources and the predictions made by an automatic system. This also includes several methods to address it, based on the category thresholds and the classification scores.

Document Performance Prediction The introduction and definition of Document Performance Prediction (DPP) and different methods to address it. DPP main focus is to predict the classification performance of a document.

Misclassification Degrees in Classification Evaluation A family of metrics that consider levels of misclassification for classification, based on category dependencies.

Dynamic *k*-NN An adaptive variation of *k*-NN that optimises the number of neighbours per category, instead of globally, without creating multiple binary classifiers.

1.4 Thesis Structure

Chapter 2: General knowledge related to the classification process.

Chapter 3: Literature review of content representation, classification and descriptive approaches.

Chapter 4: Seamless modelling, using a descriptive approach, of the text classification process. It illustrates the high level customisation and the automatic translation from probabilistic Datalog programs to their mathematical formulation.

Chapter 5: Modelling of Knowledge-Enhanced and relational classifiers in probabilistic Datalog. It includes a framework for knowledge exploitation in text classification, the combination of relational and textual classification in heterogeneous environments, and the high level integration of information retrieval tasks.

Chapter 6: Strategies and tasks to exploit knowledge in different steps of the classification process. The tasks of Semi-Automatic Text Classification (SATC) and Document Performance Prediction (DPP) are presented, and several algorithms are introduced to address them. An evaluation framework using class dependency measures to capture near-misses is introduced. Also, a variation of k-NN that adapts the number of neighbours per class is used to improve the scoring step.

Chapter 7: The conclusions and findings, as well as the main limitations and research outlook.

Chapter 2

Text Classification Background

This chapter introduces the general background for text classification, explaining the steps in the classification process and defining its different subtypes. Text classification, also known as text categorisation, is the process of assigning classes, from a preselected set, to a specific document. These decisions are based on the concepts represented by each of the classes and the documents. The set of classes is given, as well as a sample of previously labelled documents to learn from. Text classification is a pivotal task for information systems that is usually applied to organise the massive amount of data that users deal with in a daily basis. It has been applied in a variety of applications such as news categorisation or spam detection [Sebastiani, 2002]. In addition, it has been proven to be helpful in conjunction with other information retrieval tasks such as Web search via Learning to Rank [Liu, 2009].

The classification process starts with a document collection of manually labelled documents. After this, the collection is divided in three disjoint subsets, namely train, validation and test sets. The training set is used by the system to learn the patterns that will be used to infer the classes for new, unseen documents. The validation test is needed to optimise the parameters of a classifier (e.g., the number of neighbours in k-NN) with a set of documents that have not been used to train the model, nor are used to evaluate their performance. In some cases, this validation set is not explicitly specified and the tuning of parameters is done via cross-validation using the training set. This approach can be seen as implicitly creating different validation sets (one per fold). Finally, the testing set is used to evaluate the quality of any given model by comparing its decisions to the human judgements. There are different strategies to split the collection in these sets. Some collections provide specific split information in order to ensure replicability and to allow direct comparisons. If no split strategy is specified, a random approach is usually applied, where the ratios for each subset have to be decided.

Once the split of the collection is defined, the next step is to decide what documents are going to be considered by the algorithm to learn from. The trivial (and most common) option is to select all the training documents. The main rationale behind this option is that the more data available for the classifier, the higher quality can be achieved. However, the labelling of documents is costly, and in some cases, the class assignments are known to be unreliable [Voorhees, 2000] [Webber and Pickens, 2013]. For these reasons, a subset of the training documents might be used instead. The field addressing the problem of selecting which documents should be selected is known as Active Learning [Lewis and Gale, 1994].

The next step in the process is to represent the content of the documents. Similarly to information retrieval, two techniques are usually applied: Stop-words removal and stemming. Stop-words removal [Luhn, 1960] eliminates words that are considered without a meaning (e.g., "the", "a", "until", ...). It is usually done by comparing every term in the document against a list of stop-words, that are language dependent. The (english) stop-words list that has been traditionally used in information retrieval and text processing is the one presented in [Salton, 1971]. The results of this thesis use the same list of stop-words. Nonetheless, other word lists are used by different systems and domains. Stemming refers to the process of obtaining the syntactic root of a word by, usually, removing prefixes and suffixes. This helps to minimise the matching problem by mapping words with the same root. This approach assumes that words with the same root are semantically related. For instance, "play", "playing" and "played" will be represented by the token "play". Different algorithms apply different stem rules. The most commonly used is the Porter stemmer [Porter, 1980].

After stop-words removal and stemming are applied, each document is represented as a bag of tokens. The next step is to transform it to a vectors of weights, where each component measures the importance of a specific term for the document, for instance, using the term frequency values. Other non-textual features can be used (e.g., the number of nouns in the document or the number of terms) depending on the specific classification task. One of the main challenges of the vector representation is that its dimensionality can be in the order of tens of thousands. However, most of those features have almost no discriminative power, this is, they do not carry any information on how to classify new documents into the classes. Therefore, they can be treated as noise. This situation appears, for terms that occur almost uniformly distributed over the different classes. Furthermore, high dimensionality can be an issue for some of the models, related to efficiency and/or effectiveness. For these reasons, feature selection is usually applied. A subset with the most informative features is selected as the representation for the documents, while the others are ignored [Yang and Pedersen, 1997].

The classification step, referred to as scoring to differentiate it from the whole classification process, is applied after all documents have been represented. The scoring has two independent sub-steps, the model learning and the class prediction for unseen documents. The former learns how to perform the classification based on the training data, whereas a score for each class is provided in the prediction phase. Some methods, such as traditional SVM [Joachims, 1998], compute a binary decision value, indicating if the document belongs to the class or not, instead of a numeric score. In general, there are variations for this type of classifiers to provide a confidence score for each assignment [Platt, 1999].

The final goal of the classification process is to decide which classes a document belongs to, based on the document-class scores produced by the classifier. The transformation from scores to binary decisions is performed in the thresholding step [Yang, 2001].

The last step of the process is the quality evaluation, where the decisions made by the system are compared to the ground truth. These judgements are assumed to be correct and complete. Nonetheless, research have shown that this is not true, as the disagreement between assessors for both text classification and document retrieval is relatively high [Voorhees, 2000] [Webber and Pickens, 2013].

The concepts previously explained summarise a general classification processes. Nonetheless, there are several sub-types of classification problems with specific characteristics that can alter the general data flow. The types of classification problems and the methods for each step of the classification process are illustrated in the next sections.

2.1 Types of Classification Problems

Different types of classification problems can be defined with respect to different dimensions such as the structure of the data or the output of the system, altering the flow of the general classification process. The main classification types are summarised in the following sections. Most of the details of this section are based on the survey presented by [Sebastiani, 2002].

2.1.1 Single-Class vs Binary vs Multi-Class vs Multi-Label

A classification task can be required to decide if a document belongs or not to a unique class. For instance, in the case of email spam detection, where an email has to be classified as being spam or not. This scenario is known as a binary classification. If the training set only contains documents labeled in one class, it becomes a single-class problem. This challenge is closely related to the fields of outlier detection and novelty detection [Moya et al., 1993] [Khan and Madden, 2010].

On the other hand, if each document is required to be labelled just in one class, but there are multiple classes to choose from, the problem is referred to as multi-class classification. Language detection is a good example of a multi-class problem, where the (unique) language of a document has to be detected from a set of languages.

If the tasks allows a document to belong to multiple classes simultaneously, it becomes a multi-label problem. One example is news categorisation, where a document can be focused on multiple topics simultaneously (e.g., `sports` and `finance` for an article about the purchase of a football team).

2.1.2 Soft vs Hard

A classification system can be used by itself, being completely automatic and independent, or it can be used to support a decision process. In the former case, a decision if a document belongs to each class is expected, whereas a numeric confidence value is provided for the latter. These two cases are referred to as hard and soft classification respectively.

Examples for soft classification include systems that use the confidence in further inference steps such as support systems, where the classification output is interpreted by an human expert (e.g., medical diagnostics), and systems that expect a ranking of categories.

2.1.3 Flat vs Hierarchical

The class taxonomy is the set of considered classes and their structure. Based on it, two different sub-types of classification are defined: Flat (or traditional) classification if there are no relationships between classes, and hierarchical if the taxonomy has a structure with some classes being sub-topics of other classes. Hierarchical classification presents interesting challenges for classification because the relationships between classes should be taken into account, both in the class prediction and the evaluation steps [Sun and Lim, 2001].

2.1.4 Preferential Classification

In multi-label classification, each category assignment is usually treated equally, meaning that all labels for a document have the same importance. However, in some environments this might not be the case. For instance, given a document related to company acquisitions, where one of the companies is a football team: Although, the document is labelled in the classes `finance` and `sports`, it could be argued that the topic `finance` is much more relevant than `sports`.

Preferential Classification, also known as Preferential Learning, addresses the situation where there is a explicit preference of some classes over others. Preferential classification has been formally defined as follows [Aiolli et al., 2009]:

Definition 1. *The attribution to a textual document d_i not of a subset $C_i \subseteq C$ of the set of categories C (as in standard multi-label aka n-of-m text categorisation), but of a partial ordering among the set of categories C; this partial ordering specifies which category applies more than (or is preferred to) which other category to the document.*

The specific characteristics of preferential learning imply that new evaluation measures taking into account the order of the assignments are needed, and that only collections with non-binary assessments can be used. Aioli *et al.* proposed a new evaluation metric for preferential learning that exploits a weighted combination of different F_1 measures, where the weights represent the effect of considering erroneous swapping between different priorities of classes [Aiolli et al., 2009].

2.1.5 Cost-Sensitive Learning

One main assumption that is usually applied in classification problems is that all errors have the same cost. However, this is not true in most real problems. For instance, in advertisement, the cost of sending an email to a non-respondent is very small, but the cost of not mailing someone who would respond is the entire profit lost [Domingos, 1999]. Cost-Sensitive Learning assumes that some misclassifications are more "costly" than others. For this reason, a cost matrix C is developed, where a specific entry $C_{i,j}$ represents the cost of predicting class i when the true class is j. If $i = j$ the prediction is correct, and it is incorrect otherwise.

Table 2.1: Misclassification Costs.

		Expert Judgements	
		NO	YES
Judgements	NO	C(0, 0) = c_{00}	C(0, 1) = c_{01}
Classifier	YES	C(1, 0) = c_{10}	C(1, 1) = c_{11}

The optimal prediction (in binary classification) for a document d is the class i that minimises the following equation, where $P(j|x)$ is the probability of class j given the document d [Elkan, 2001]:

$$\mathrm{L}(d,i) = \sum_j P(j|d) \cdot C(i,j) \tag{2.1}$$

Conceptually, the cost of labelling a true positive has to be lower (potentially zero) than the cost of labelling it incorrectly. More specifically, following the notation presented in Table 2.1. it should be always the case that $c_{10} \geq c_{11}$ and $c_{01} > c_{00}$. Most of the methods that address this problem follow the principle that, for the binary case, the optimal prediction is class C_1 if and only if the cost of its prediction is less than or equal to the expected cost of predict class C_0, as the following equation illustrates:

$$P(j=0|x) \cdot c_{10} + P(j=1|x) \cdot c_{11} \leq P(j=0|x) \cdot c_{00} + P(j=1|x) \cdot c_{01} \tag{2.2}$$

Given $p = P(j=1|x)$, this is equivalent to:

$$(1-p) \cdot c_{10} + p \cdot c_{11} \leq (1-p) \cdot c_{00} + p \cdot c_{01} \tag{2.3}$$

Cost-Sensitive learning has been addressed extensively in the literature, mainly for two-class problems, but very little research is focused on multi-label environments [Zhou and Liu, 2010].

2.2 Document Selection

The standard strategy for classification is to use all available (labelled) documents, following the intuition that better quality will be achieved with more data. However, human labelling is a costly and time consuming process. Moreover, in some cases, a collection could be known to contain inconsistent or noisy data. For these reasons, different techniques have been proposed to optimise the labelling process. Active learning addresses the selection of documents to be labelled and used as training examples [Lewis and Gale, 1994], where the main goal is to achieve, at least, comparable performance with respect to supervised learners, while requiring fewer documents to be labelled, therefore reducing the cost. There are four main approaches for active learning, depending on the specific assumptions applied:

- Uncertainty Sampling [Lewis and Gale, 1994]. The documents with higher uncertainty are selected first as it is assumed that they are more informative.
- Relevance Sampling [Lewis and Gale, 1994]. The most relevant documents are selected. The rationale is that they carry more relevant information and less noise about the topic.
- Expected-error reduction [Yan et al., 2003]. This technique minimises the expected error on the unlabelled data. Its main challenge is that it is computationally expensive.
- Committee-based [Liere and Tadepalli, 1997] [Tong and Koller, 2001]. Conceptually similar to uncertainty sampling using a committee of classifiers. The selected documents are those with higher uncertainty, based on the degree of disagreement between the classifiers (more disagreement implies more uncertainty).

If the principles of active learning are applied over time, as new data arrives, the task is referred to as Incremental Learning [Lewis and Gale, 1994]. The main goal in this situation is to iteratively select documents to be manually labelled, trying to optimise the quality gain by using them as training examples in the future. Extensive research has been done related to single-label environments [Lewis and Gale, 1994] [Tong and Koller, 2001], where it is assumed that every document belongs to only one class. However, very limited research has tried to address the same problem in a multi-label environment [Esuli and Sebastiani, 2009] [Yang et al., 2009], even though the majority of classification problems (especially in text classification) have a multi-label nature.

2.3 Feature Selection

Feature selection filters the features that are more discriminative between the categories within a collection. In other words, those features that best differentiate between classes. This step is applied to eliminate the noise created by features that appear uniformly distributed over different classes, as well as to increase the efficiency of the system. Three of the most commonly used methods, namely document frequency, information gain and χ^2 are presented in this section. These methods achieve similar performance over different text classification collections, and there is a strong correlation between them [Yang and Pedersen, 1997]. The optimum number of features has to be observed empirically for each collection, classification method and feature selection model. The removal of the terms that appear in fewer documents than a specific threshold (document frequency feature selection) can be seen as a over-simplistic measure that has been traditionally used in order to increase efficiency rather than effectiveness. However, its performance has been reported to be comparable to more advance methods [Yang and Pedersen, 1997]. The assumption is that rare terms are not informative for category prediction, nor influential enough to affect global performance. The definition of document frequency (df) for a given term is defined as follows, where Tr is the set of training documents and $n_D(t)$ is the number of documents a term appears in:

$$\mathrm{df}(t) = n_D(t) = |\{d \in Tr : t \in d\}| \tag{2.4}$$

Information Gain is a common technique that has been used in machine learning to measure the discriminative power of a feature [Quinlan, 1986] [Mitchell, 1997]. Although it was originally developed within the context of decision trees, Information Gain has also been applied for feature selection in text classification [Sebastiani, 2002]. It measures the numbers of bit of information obtained for category prediction by knowing the presence or absence of a term in a document. Following the same formalism presented by Sebastiani, t_k and $\overline{t_k}$ represent the event that the term t_k appears or not in a document. Similarly, c_i and $\overline{c_i}$ represent that a document belongs or not to the class c_i. Using this formulation, probabilities are interpreted on an event space of documents. For instance, $P(t_k, \overline{c_i})$ measures the probability that, given a random document, t_k appears in the document and the document does not belong to c_i. Information gain is then defined as follows:

$$\mathrm{IG}(t_k, c_i) = \sum_{[c \in \{c_i, \overline{c_i}\}]} \sum_{[t \in \{t_k, \overline{t_k}\}]} P(t,c) \cdot \log \frac{P(t,c)}{P(t) \cdot P(c)} \tag{2.5}$$

[Yang and Pedersen, 1997] proposed a different formalism for multi-class problems, where the goodness of each term is measured globally with respect to all categories on average. The information gain for a term t is defined as follows, where C is the set of classes:

$$\begin{aligned} \mathrm{IG}(t) \quad = \quad & -\sum_{c_i \in C} P(c_i) \cdot \log P(c_i) \\ & + P(t) \sum_{c_i \in C} P(c_i|t) \cdot \log P(c_i|t) \\ & + P(\bar{t}) \sum_{c_i \in C} P(c_i|\bar{t}) \cdot \log P(c_i|\bar{t}) \end{aligned} \tag{2.6}$$

The χ^2 statistic measures the lack of independence between two events. In the case of text classification, the events are, for a document space, the occurrence of a term t and the labelling in a class c. Following the same notation previously explained, where $|T_r|$ is the set of training documents, χ^2 is defined as follows:

$$\chi^2(t_k, c_i) = \frac{|Tr|[P(t_k, c_i) \cdot P(\overline{t_k}, \overline{c_i}) - P(\overline{t_k}, c_i) \cdot P(t_k, \overline{c_i})]^2}{P(t_k) \cdot P(\overline{t_k}) \cdot P(c_i) \cdot P(\overline{c_i})} \tag{2.7}$$

All the strategies, with the exception of the reformulation of information gain introduced by Yang *et al.*, compute a value that measures the discriminativeness of a term for each category. Nonetheless, a unique value per term is required in order to select the more meaningful features globally. For this reason, an aggregation operation should be applied. Three different functions have been proposed in the literature [Sebastiani, 2002] [Yang and Pedersen, 1997]. Adding the values for each class, applying a weighed sum (usually based on the relative number of documents being labelled in each class), and using the maximum value. These methods are formalised below, where $f(t,c)$ measures the discriminative power of a term t for class c:

$$f_{sum}(t) = \sum_{c_i \in C} f(t, c_i) \tag{2.8}$$

$$f_{wsum}(t) = \sum_{c_i \in C} w(c_i) \cdot f(t, c_i) \tag{2.9}$$

$$f_{max}(t) = \max_{c_i \in C} f(t, c_i) \tag{2.10}$$

Research suggest that feature selection can reduce the number of features by 90% for text classification while, at least, maintaining the same levels of quality [Yang and Pedersen, 1997].

2.4 Content Representation

The most common document representation method in information retrieval and text classification is to analyse every document as a bag of words, and to observe the importance of its terms as a vector of feature weights. The term weights are usually based on two factors: A frequency factor that measures how common the term is in the document, and an informativeness factor that represents how relevant the term is in the collection. The most representative terms are common in the document, while being rare in the collection. This concept is one of the foundations of information retrieval and it has been applied for more than four decades [Sparck-Jones, 1972] [Salton, 1988] [Buckley et al., 1994].

Several algorithms following these principles have been presented in the literature, where the two most commonly used in text classification are `tfc` (term frequency count) and `ltc` (logarithmic term frequency count) [Joachims, 1998] [Yang and Liu, 1999] [Lewis et al., 2004]. The former exploits the total term count for the frequency factor, while the inverse document frequency (idf) [Sparck-Jones, 1972] is used to compute the term informativeness. The mathematical definition of `tfc` is shown below, where $n_L(t,d)$ represents the number of occurrences (locations) for a term t in document d, $n_D(t)$ is the number of documents t appears in, and Tr is the set of train documents:

$$\text{tfc}(t,d) = n_L(t,d) \cdot \log \frac{|Tr|}{n_D(t)} \tag{2.11}$$

`ltc` applies a logarithm normalisation to the term count to decrease the relative importance of every new occurrence of a term. The `ltc` definition is shown below, following the same notation as the one shown for `tfc`:

$$\mathrm{ltc}(t,d) = (1 + \log(n_L(t,d))) \cdot \log \frac{|Tr|}{n_D(t)} \tag{2.12}$$

In addition to the document and collection dimensions, some authors have proposed to use category information to represent documents specifically for text classification tasks. These weighting approaches, known as supervised weighting strategies, use metrics traditionally used in feature selection, mainly measuring the terms discriminative power [Batal and Hauskrecht, 2009].

2.5 Traditional Classification Methods

Classifiers produce a score for each category and document to be classified. However, the mechanisms to compute such value is dependent on the specific classifier. This section introduces three of the most commonly used and well-known families of algorithms for text classification, namely Bayesian, *k*-NN and Support Vector Machines.

2.5.1 Bayesian Classifiers

Bayesian classifiers use the Bayes Theorem to infer the categories for a document. In particular, they calculate the probability of a class given a document as follows, where d is a document and c one of the classes:

$$P(c|d) = \frac{P(d|c) \cdot P(c)}{P(d)} \tag{2.13}$$

This equation could be extended by reformulating $P(d|c)$ and $P(d)$ based on the terms inside document d:

$$P(c|d) = \frac{P(d|c) \cdot P(c)}{P(d)} = \frac{P(t_1, t_2, \ldots, t_n|c) \cdot P(c)}{P(t_1, t_2, \ldots, t_n)} \tag{2.14}$$

In text classification, independence between features is usually assumed, given the context of a class. Under this assumption, the joint probability from the general equation for Bayesian classifiers is modified as follows:

$$P(t_1,t_2,...,t_n|c) = P(t_1|c) \cdot P(t_2|c)... \cdot P(t_n|c) \tag{2.15}$$

Then, we can define the probability of a document being labelled in a class as follows, where $n_L(t,d)$ is the number of times that the term t appears in document d:

$$P(c|d) = \frac{P(c)}{P(d)} \cdot \prod_{t \in d} P(t|c)^{n_L(t,d)} \tag{2.16}$$

Classifiers that make these assumptions are usually referred as Naive-Bayes, even if there are differences between them. Two variations of this family, based on different distributions, are illustrated in the next sections: a multi-variate Bernoulli and a multinomial Naive-Bayes classifiers [McCallum and Nigam, 1998]. Furthermore, the concept of of the zero-probability problem and different smoothing techniques are also explained.

2.5.1.1 Multi-variate Bernoulli

A multi-variate Bernoulli distribution represents different features using a binary vector that indicates the occurrence or not of terms in a document. This model computes the class prior probability by the maximum likelihood estimate as follows, where Tr is the set of train documents:

$$P(c) = \frac{\sum\limits_{d \in Tr} P(c|d)}{|Tr|} \tag{2.17}$$

Assuming $P(d) = \frac{1}{|T_r|}$, the document prior is then formalised as shown below:

$$P(d) = \sum_{c \in C} P(c) \cdot P(d|c) \tag{2.18}$$

$P(d|c)$ and $P(t|c)$ are specified as follows, where $B_{d,t}$ is the binary value indicating if term t appears in document d:

$$P(d|c) = \prod_{t \in d} (B_{d,t} \cdot P(t|c) + (1 - B_{d,t}) \cdot (1 - P(t|c))) \tag{2.19}$$

$$P(t|c) = \frac{\sum\limits_{d \in D} B_{d,t} \cdot P(c|d)}{\sum\limits_{d \in D} P(c|d)} \tag{2.20}$$

This model applies the independence assumption and it explicitly takes into account the non-occurrence probability of features that are not in the document.

2.5.1.2 Multinomial

Multinomial Bayes uses a non-binary vector for representing different features, exploiting the frequency of the terms in each document. Then, the probability of a document given a class is defined as follows, where $n_L(t,d)$ represents the number of times a term t occurs in document d.

$$P(d|c) = P(|d|) \cdot |d|! \prod_{t \in d} \frac{P(t|c)^{n_L(t,d)}}{n_L(t,d)!} \tag{2.21}$$

The probability of each term given a class, is shown below:

$$P(t|c) = \frac{1 + \sum\limits_{[d \in D]} n_L(t,d) \cdot P(c|d)}{|T| + \sum\limits_{[t \in d]} \sum\limits_{[d \in D]} n_L(t,d) \cdot P(c|d)} \tag{2.22}$$

Class and document priors are computed as they were in the Bernoulli model. The multinomial naive-based classifier has shown better performance for the specific task of text classification compared to the multi-variate Bernouilli [McCallum and Nigam, 1998].

2.5.1.3 Smoothing

One challenge for the Bayesian classifiers is that, based on the available information, the probability of a term given a class might be zero. As a result, no document containing such term can be assigned to the class. To avoid this case, known as zero probability problem, an estimate of within-class term probability $P(t|c)$ is defined. Two options are commonly applied: A Laplace-based smoothing that virtually adds once occurrence of each term to each class, and a mixture-based approach that combines the probability based on training documents (foreground model) and a background model based on the collection.

2.5.2 k-Nearest Neighbours (k-NN)

k-NN is a lazy learning instance-based method that categorises documents taking into account what training examples are the "nearest", based on a similarity measure [Dasarathy, 1991]. There are several strategies to compute the score for each class. The two most common are voting, where each of the top neighbours is considered equally; and a weighting approach, where each neighbour contribution to the final score is weighted by its similarity with respect to the document to be classified. In both cases, the score of each class is computed as the sum of scores for each neighbour labelled in the class, observing only the k most similar documents.

A similarity function has to be defined in order to compare and rank the training documents with respect to the document to be classified. The cosine similarity is the usual method for this task. Given two documents d_i and d_j, represented as vectors of term weights, their cosine similarity is defined as follows:

$$\mathrm{sim}(d_i, d_j) = \frac{d_i \cdot d_j}{||d_i|| \cdot ||d_j||} \tag{2.23}$$

Once the similarity function if specified, the mathematical definition of k-NN can be formalised as follows, where $NN_k(d)$ represents the set of k nearest neighbours of d and y models the class association between a document and a class:

$$\mathrm{score}_k\text{-NN}(c,d) = \sum_{d' \in NN_k(d)} \mathrm{sim}_k(d,d') \cdot y(d',c) \tag{2.24}$$

In addition to the similarity function, the number of neighbours (k) has to be tuned. This is usually done based on a validation set, optimising k with respect to a quality measure.

A probabilistic formulation of k-NN based on the total probability theorem, computed over the set of nearest neighbour documents, has also been proposed in the literature [Gövert et al., 1999].

$$P(c|d) = \sum_{d' \in \mathrm{NN}_k(d)} P(d'|\mathrm{NN}_k(d)) \cdot P(c|d') \cdot P(d'|d) \tag{2.25}$$

$$P(d'|d) = \sum_t P(d'|t) \cdot P(t|d) \approx \mathrm{sim}(d,d') \tag{2.26}$$

$P(d'|NN_k(d))$, the normalisation factor, is required since the documents are not viewed as disjoint events.

2.5.3 Support Vector Machines (SVM)

Support Vector Machines (SVM) [Cortes and Vapnik, 1995] is a relatively recent machine learning technique that is based on the structural risk minimisation principle [Vapnik, 1995]. Its main principle is to find a hypothesis *h* for which we can guarantee the lowest error for an unseen and randomly selected test example. SVM was suggested to be a good candidate for text classification problems [Joachims, 1998]. The main reason for this assumption is that text classification represents a high dimensional problem that cannot be easily reduced to a small set of relevant features because there are few truly irrelevant features (i.e., terms that have no impact whatsoever in the category decision). SVM suits this type of environment because it provides over-fitting protection that does not necessarily depend on the number of features. In other words, SVM can deal automatically with very large number of features without over-fitting to the specific set of training examples. Furthermore, SVMs are also well suited to problems with sparse document vectors (i.e., where most of the features are zero) such as text classification. In addition to the theoretical arguments, SVMs have been empirically proven to be very effective for document categorisation and other tasks [Joachims, 1998]. SVM maximises the distances between categories by finding the hyperplane that better divide them. In its basic form, it represents a binary linear classifier. Extensions of the original model allow non-perfect matching with training data (to avoid outliers) and non-linear functions. Moreover, methods have also been proposed to apply it to multi-label collections using "one-vs-rest" approaches.

2.5.4 Classifiers Committee

Classifiers committee, also known as ensemble classifiers, are learning algorithms that base their decision on a set of classifiers. The main assumption is that such combination can achieve better and more consistent quality than any of them being applied separately. It has been suggested that, the more different the theoretical foundations of the models are, the better the quality is expected to be [Larkey and Croft, 1996]. On the other hand, classifiers with too different quality levels do not achieve significant improvements [Tumer and Ghosh, 1999]. Therefore, the optimum configuration is to use multiple classifiers based on different theoretical grounds but with comparable quality. Other research has shown that *"A necessary and sufficient condition for an ensemble of classifiers to be more accurate than any of its individual members is if the classifiers are accurate and diverse"* [Dietterich, 2000].

Two characteristics have to be specified for traditional ensemble methods: The classifiers to be used, and the combination function [Sebastiani, 2002]. For the latter, there are four traditional algorithms:

- Majority Voting [Li and Jain, 1998]. Decisions agreed by the majority are applied.
- Weighted Linear Combination [Larkey and Croft, 1996] according to the expected quality of each classifier.
- Dynamic Classifier Selection [Li and Jain, 1998] only taking into account the best classifier for the m most similar examples based on validation data.
- Adaptive Classifier Combination [Li and Jain, 1998] where the weight of each classifier is the quality achieved for the m most similar examples.

Other ensemble methods manipulate data, while the same classifier is used, iteratively modifying the training document set [Dietterich, 2000]. For instance, Bagging uses several samples of the same dataset to train a classifier. Boosting is a variation of this type of ensembles, where the weights of each training example change every iteration. The weights for the documents that were misclassified in the previous step is increased to place more importance on them. This type of learning is represented by the AdaBoost method [Schapire et al., 1998]. A slightly different technique to combine classifiers, known as stacking, is to use the output of a set of classifiers as input for a “second-level” meta-classifier [Wolpert, 1992] [Ting and Witten, 1999].

All these techniques are document-independent learning methods, in the sense that they perform the same steps independently of the documents to be classified. A meta-classifier that includes global (e.g., output of classifiers in the first level) and document specific features (e.g., document length) was presented by [Bennett et al., 2005]. The authors proposed to use reliability indicators and included four types of data: the amount of information present in the original document (e.g., document length or number of unique terms), the information loss with a specific representation (e.g., percentage of terms removed in feature selection), the sensitivity of the decision to evidence shift, and voting statistics.

2.6 Thresholding Strategies

After the classification (or scoring) step is performed, a score for each class and document to be classified is available. Then, a decision for each pair document-class has to be made to assign the document to the category or not. Different thresholding strategies have been presented in the literature to address this challenge. This section summarises the most common techniques, as presented by [Yang, 2001]. A cross-validation process based on a validation set should be applied for all the thresholding methods that require parameter tuning to optimise a quality measure.

2.6.1 RCut

Given a document, RCut sorts the classification scores for each category, and it decides that the document should be classified in the top r of them. The value of r ranges from one to m (the number of categories). The special case of $r = 1$ has been extensively used in the machine learning community for single-label problems [Joachims, 1998].

2.6.2 PCut

PCut uses a category-based approach. Given a specific class c_i, it sorts the scores for each document, where the top-k_i documents are assigned to the category. The calculation of k_i is shown below, where x is the average number of documents assigned to a class, and C is the set of classes. x is optimised based on a global quality metric.

$$k_i = P(c_i) \cdot x \cdot |C| \tag{2.27}$$

When x is the number of documents in the class, the system behaves like "Mr.YES", where all documents are assigned to all classes. On the other hand, $x = 0$ implies that no documents are assigned to any class [Yang, 2001], also known as a "Mr. NO" behaviour.

2.6.3 SCut

SCut is similar to PCut, being also a category-based approach. Given a category, the scores for each document are sorted. Then, a numerical threshold is optimised, with respect to a quality measure, for each class. As a result, a different threshold per class is obtained, therefore, this method does not guarantee a global optimum.

2.6.4 SCutFBR

When a class has very few examples, SCut can produce a threshold that is too high, or too low based on the validation set. In the first case, relevant documents for the class will not be classified, lowering the macro-averaged[1] quality. On the other hand, if the threshold is too low, several documents will be incorrectly classified in the class, therefore, decreasing both micro and macro-averaged quality values[1] [Yang, 2001]. A variation of SCut addresses these challenges by treating differently the classes with quality lower than a specific value known as `fbr`. This modification, coined SCutFBR [Yang, 2001], has two sub-variations: SCutFBR$_{0.0}$ sets the class threshold to infinite for those classes with a lower quality than the `fbr` value, and SCutFBR$_{1.0}$ that sets it to the score of the top ranked document.

2.7 Quality Evaluation Measures

The most common metrics to evaluate the quality of text classification systems are precision, recall and F$_1$ [van Rijsbergen, 1979]. All these measures are based on a contingency table (see Table 2.2) that compares the ground truth and the decisions made by a classification system for a category C_i [Sebastiani, 2002].

Table 2.2: Contingency Table for Class C_i.

		Expert Judgements	
		YES	NO
Classifier	YES	TP_i	FP_i
Judgements	NO	FN_i	TN_i

A contingency table illustrates four possible scenarios for each document that has been classified (either as positive or negative) with respect to a category. True Positive (TP) when the system and the assessment agree that the document belongs to the class; False Positive (FP) when the system incorrectly assigns the document to the class; False Negative (FN) for the case of not assigning a document that should be in the class and; finally, True Negative (TN), if the system and the judgement specify that the document do not belong to the class. The number of assignments of each type are aggregated over the set of classified documents to provide the number of TP, FP, FN

[1]Micro and macro-averaged quality measures are explained in Chapter 2.7

and TN per class. Precision, Recall and F_1 are based on such definitions, and their formulation for a specific class (C_i) is shown below:

$$\mathrm{Pr}_i = \frac{TP_i}{TP_i + FP_i} \tag{2.28}$$

$$\mathrm{Re}_i = \frac{TP_i}{TP_i + FN_i} \tag{2.29}$$

$$F_{1i} = \frac{2 \cdot \mathrm{Pr}_i \cdot \mathrm{Re}_i}{\mathrm{Pr}_i + \mathrm{Re}_i} \tag{2.30}$$

All measures can be computed across a set of classes using two different strategies: micro-averaging, if each class influences the final score depending on its number of documents; and macro-averaging, where each class has the same importance. The definition of precision, recall and F_1 using both micro (μ) and macro (M) averaging strategies are formulated below, where C represents the set of classes.

$$\mathrm{Pr}^{\mu} = \frac{\sum\limits_{c_i \in C} TP_i}{\sum\limits_{c_i \in C} (TP_i + FP_i)} \tag{2.31}$$

$$\mathrm{Re}^{\mu} = \frac{\sum\limits_{c_i \in C} TP_i}{\sum\limits_{c_i \in C} (TP_i + FN_i)} \tag{2.32}$$

$$F_1^{\mu} = \frac{2 \cdot \mathrm{Pr}^{\mu} \cdot \mathrm{Re}^{\mu}}{\mathrm{Pr}^{\mu} + \mathrm{Re}^{\mu}} \tag{2.33}$$

$$\mathrm{Pr}^{M} = \frac{1}{|C|} \sum_{c_i \in C} \mathrm{Pr}_i \tag{2.34}$$

$$\mathrm{Re}^{M} = \frac{1}{|C|} \sum_{c_i \in C} \mathrm{Re}_i \tag{2.35}$$

$$F_1^{M} = \frac{1}{|C|} \sum_{c_i \in C} \mathrm{F}_{1i} \tag{2.36}$$

For the specific case of F_1, some authors have incorrectly computed its macro-averaged value, by multiplying the macro-averaged recall and precision, instead of averaging the F_1 values for each class. As a result, a different, usually higher, quality is computed [Yang, 2001].

Other evaluation metrics have been proposed but their use is minimal in text classification. Several of these measures such as Break-Even point, accuracy or fallout are explained in detail in [Sebastiani, 2002].

2.8 Collections

Different collections are used in this thesis to evaluate tasks, models and approaches from different perspectives. Such collections are explained in this section, and Table 2.3 summarises their main characteristics.

Table 2.3: Collections Statistics.

Collection	**#Docs(train/test)**	**avg. class/doc**	**#Classes**
`20-newsgroups`	9840/6871	1	20
`Reuters-21578`	7770/3019	1.24	90
`Reuters-21578-115`	9603/3299	1.06	115

2.8.1 20-newsgroups

`20-newsgroups`, also known as `20news` or `20-newstories`, is a collection of approximately 20,000 newsgroup documents related to 20 categories with almost uniform distribution of documents over categories. Cross-posting emails have not been considered. As a result, it is a single-label collection, where each document belongs to one, and only one, category. The train/test split is based on time, as it is suggested[2].

2.8.2 Reuters-21578

`Reuters-21578` contains structured information about newswire articles that can be assigned to several classes[3]. It has a a highly skewed distribution of documents over categories. This is arguably the most commonly used multi-label collection for text classification. Two variations of the "ModApte" split are used: `Reuters-21578` and `Reuters-21578-115`: `Reuters-21578`

[2]Obtained from http://people.csail.mit.edu/jrennie/20Newsgroups

[3]Available at http://www.daviddlewis.com/resources/testcollections/reuters21578

uses only documents that belong to classes with at least one training and one test document. As a result, there are 7770 documents for training, 3019 for testing, and 90 categories. This is the same configuration used by [Yang, 2001]. `Reuters-21578-115` uses documents belonging to classes with at least one training or testing document. This configuration has 9603 documents for training, 3299 for testing, and 115 classes. It is the same collection used by [Berardi et al., 2012].

2.8.3 DBLP

A subset of the DBLP (http://www.informatik.uni-trier.de/ ley/db/) repository of scientific publications was introduced by [Cai et al., 2005] to investigate relational classification. This collection[4] contains 14376 articles, 14475 authors and 20 conferences belonging to four different topics, namely database (DB), data mining (DM), information retrieval (IR) and artificial intelligence (AI). The collection contains textual information (i.e., the title) for research papers and relational information representing who are the authors of each publication and what venue it was publish in. In addition, a small percentage of entities were manually labelled to be used for testing and evaluation. In particular, the test set has 4057 authors, 100 papers and all 20 conferences. The split is done by randomly selecting 50% of the documents for training and the rest for testing.

[4]Downloaded from http://web.engr. illinois.edu/ mingji1/DBLP_four_area.zip

Chapter 3

Literature Review

This chapter introduces the literature review for several fields that are directly related to the contents of this thesis, namely descriptive modelling, knowledge exploitation and performance prediction.

3.1 Descriptive Approaches

Descriptive approaches, also referred to as declarative, define the solution to a specific problem, or the conditions for such solution, rather than the steps to solve it. As a result, a higher degree of abstraction is obtained, compared to other approaches. This abstraction has been reported to increase productivity, because the modelling is clearer and it implies that less time is spent understanding, debugging and upgrading code [Lloyd, 1994]. A definition of descriptive approaches, as proposed by Lloyd, is shown below:

Definition 2. *Programming paradigm that involves stating* **what** *is to be computed, but not necessary* **how** *it is to be computed.*

Descriptive approaches have significant improvements over algebraic methods, specially when flexibility and changeability over short periods of time are required, and when the data or the information needs are ambiguous or dynamic. Abstraction provides an open and flexible environment where models and tasks can be easily integrated.

Descriptive approaches can be used to address some of the main challenges of enterprise search. Some of these challenges are *"the enormous scale, fluid collection definition, great heterogeneity, unfettered interlinking, democratic publishing, the presence of adversaries and most of all the diversity of purposes for which Web search may be used"* [Hawking, 2004]. The same author claims that the reason for the poor performance of enterprise search systems in real-environments is the extensive engineering effort that is needed in order to adapt methods developed in the laboratory. Furthermore, a "perfect" enterprise search is required *'to seamlessly and scalably combine structured (e.g., relational) as well as unstructured information in a document for search, as well as for organisational purposes (clustering, classification, etc.) and for personalisation"* [Abrol et al., 2001].

There has been a continuous line of research regarding abstraction layers using descriptive technologies for different information tasks. For instance, a declarative specification language (Dyna) has been used to model Natural Language Processing (NLP) algorithms [Eisner et al., 2005]. The authors claim that a declarative specification is helpful, even if it is slower than hand-crafted code. Other example is the description of a framework that synthesises and extends deductive and semi-ring parsing, adapting them for translation [Lopez, 2009]. This work shows that logic make *"an attractive shorthand for description, analysis and construction of decoding algorithms for translation"*, and that such technology is beneficial when implementing large-scale translation systems, identified by the authors as a major engineering challenge. In addition, they also claim that the logical description helped them to understand and compare different models.

The concept of abstraction in information retrieval has been addressed in the literature, usually applying concepts from the database community or proposing mathematical abstractions to define information retrieval problems. One of the first approaches of this type integrates structured data and text using the relational model [Grossman et al., 1997], concluding that the relational model offers scalable performance, with the ability to integrate text and relations in a portable fashion. Another example is a parameterised search system that allows flexibility in user queries and provide an easy mechanism for system engineers to customise search strategies [Cornacchia and de Vries, 2007]. This is possible using a declarative language that is based on a mathematical abstraction known as the Matrix Framework [Roelleke et al., 2006] that describes information retrieval concepts as matrix spaces and operations. The authors also claim that the algorithms should be modelled as similar as possible to the problem definition, abstracting away any other details.

The idea of decoupling search strategies from algorithms and data structures is known in the database community as *data independence*. The same concept was proposed in Information Retrieval almost twenty years ago [Fuhr, 1996]. Furthermore, DeVries also included the notion of *content independence* to measure the lack of dependency between search strategies and data representation [de Vries, 2001]. Similar arguments are used to propose a database approach with a relational model that allows rapidly developing of applications that are *"easy to understand, document and teach"* [Hiemstra and Mihajlovic, 2010]. The same research also suggests that information retrieval is still in an early stage that can be compared to the 1960's for database systems, where there was no general application program interfaces, nor standard query languages.

Among descriptive approaches, Logic and Probabilistic Logics have been applied for modelling and reasoning with knowledge in different environments [Hunter and Liu, 2010]. For instance Problog [Raedt et al., 2007] and P-Log [Gelfond et al., 2006]. The language that is used in this thesis, Probabilistic Datalog (explained in Section 3.1.1), is one of its representatives.

Most descriptive approaches share two main challenges that have to be balanced: Expressiveness and scalability. Abstraction causes a lack of control over the program flow (i.e., how to compute the solution) that implies that these solutions are usually slower. One of the strategies to partially address this challenge is to simplify or limit their expressiveness. As a result, not all models can be defined.

The next subsection illustrates the characteristics of Datalog and probabilistic Datalog (the language used in this thesis) in detail. After that, the descriptive approaches that have been applied for classification-related tasks, and those that are based on probabilistic Datalog are explained.

3.1.1 Probabilistic Datalog

Datalog is a variation of predicate logic based on function-free Horn clauses that has been extensively used as a query language for deductive databases [Ullman, 1989]. Rules have the form "*head* $\leftarrow$ *body*", where body has a set of subgoals $(b_1, b_2, ...b_n)$ that denote literals with variables and constants as arguments.

The original version of probabilistic Datalog [Fuhr, 1995] [Fuhr and Roelleke, 1997], referred to as 1st generation Datalog from now on, extends these concepts by attaching probabilistic weights

Traditional Datalog		
fact	::=	NAME '(' constants ')'
rule	::=	head ':-' body
head	::=	goal
body	::=	subgoals
goal	::=	NAME '(' arguments ')'
subgoal	::=	pos_subgoal | neg_subgoal
pos_subgoal	::=	atom
neg_subgoal	::=	'!' atom
atom	::=	NAME '(' arguments ')'
argument	::=	constant | variable
constant	::=	NAME | STRING | NUMBER
variable	::=	VAR_NAME
arguments	::=	| argument ',' arguments
constants	::=	| constant ',' constants
subgoals	::=	| subgoal ',' subgoals
1st Generation Probabilistic Datalog		
prob_fact	::=	prob fact
prob_rule	::=	prob rule

Figure 3.1: 1st Generation probabilistic Datalog.

to facts and rules. Figure 3.1 describes the syntax of Datalog and 1st generation probabilistic Datalog. Roelleke *et al.* extended PRA (Probabilistic Relational Algebra), the language probabilistic Datalog is built upon, improving its expressiveness and scalability to model information retrieval ranking functions [Roelleke et al., 2008] [Roelleke et al., 2013b]. Syntactically, it extends the syntax of 1st generation (see Figure 3.1) by providing Bayesian atoms and assumptions and probability estimation, including score aggregation (SUM, PROD). A simplified version (for improving readability) of the syntax specification for the 2nd generation probabilistic Datalog is outlined in Figure 3.2.

The assumption between predicate name and argument list is the so-called *aggregation* assumption (aggAssump). For example, for disjoint events, the sum of probabilities is the resulting tuple probability. In this case, the assumptions 'DISJOINT' and 'SUM' are synonyms, and so are 'INDEPENDENT' and 'PROD'. The assumption in a conditional is the so-called *estimation* assumption (estAssump). For example, for disjoint events, the subgoal "term(Term, Doc) | DISJOINT(Doc)" expresses the conditional probability $P(Term|Doc)$ derived from the statistics in the relation called "term". Complex assumptions such as DF (for document frequency) and MAX_IDF (max inverse document frequency) can be specified to describe probabilistic parameters commonly used in information retrieval. Expressions with complex assumptions can be

goal	::=	tradGoal | aggGoal
atom	::=	tradAtom | bayesAtom
subgoal	::=	tradSubgoal | aggGoal
tradGoal	::=	see 1st Generation
tradSubgoal	::=	see 1st Generation
bayesAtom	::=	tradAtom '|' {estAssump} evidenceKey
evidenceKey	::=	'(' variables ')'
aggGoal	::=	NAME {aggAssump} '(' arguments ')'
aggSuboal	::=	NAME {aggAssump} '(' arguments ')'
tradAssump	::=	'DISJOINT' | 'INDEPENDENT'
	::=	| 'SUBSUMED'
irAssump	::=	'SEMI_SUBSUMED' | 'DF'
	::=	| 'MAX_IDF' | ...
probAssump	::=	tradAssump | irAssump
algAssump	::=	'SUM' | 'PROD'
aggAssump	::=	probAssump
estAssump	::=	probAssump | complexAssump

Figure 3.2: 2nd Generation probabilistic Datalog: Bayesian goals.

decomposed in probabilistic Datalog programs with traditional assumptions only. However, for improving the readability and processing (optimisation), complex assumptions can be specified. The decomposition of complex assumptions is shown in [Roelleke et al., 2008]. The language also incorporates "special predicates" to specify how the engine should treat some tuples. For instance, "_sort" indicates the engine that the tuples of a given predicate (e.g., score) should be sorted according to their probabilities. Moreover, other information retrieval predicates such as idf were also presented by the same authors.

Table 3.3 shows the normalised term frequency and the category labels of a set of documents in a tabular format. On the other hand, Figure 3.4 uses a logical format to represent exactly the same information, where the value is added as a probability for each fact.

tf_sum		
Value	Term	Document
0.23	economy	d40
0.52	expectation	d23
0.12	provider	d23
0.16	reuters	d1

part_of		
Value	Document	Class
1	d1	cocoa
1	d5	grain
1	d5	wheat
1	d5	oil

Figure 3.3: Tabular Data Representation.

In both cases, tf_sum and $part\_of$ become predicates whereas, "(term, document)" and "(document, class)" are their contexts respectively. A more advanced example is shown in Figure 3.5, where the probability of a student obtaining a specific grade, $P(grade|student)$, is computed, based on probabilities of grades given subjects from the previous year. The example uses probability estimation and an aggregation assumption (SUM) which is needed for the score calculation. Figure 3.6 illustrates the input data and output values based on this modelling.

```
# Normalised Term Frecuency for terms and documents
0.23 tf_sum(economy, d40);
0.52 tf_sum( expectation , d23);
0.12 tf_sum( provider , d23);
0.16 tf_sum( reuters , d1);

# Categories Labelled for documents
 part_of (d40, cocoa);
 part_of (d23, grain );
 part_of (d23, wheat);
 part_of (d1, oil );
```

Figure 3.4: Probabilistic Logical Data Representation.

```
#P(grade|degree): Learned from knowledge base.
p_grade_degree SUM(Grade, Degree) :− grade(Student, Grade, Degree) |(Degree);

#P(grade|person): Inferred using P(grade|degree)
p_grade_person (Grade, Person) :−
        p_grade_degree (Grade, Degree) & register (Person, Degree);
```

Figure 3.5: Student grade prediction based on historic average grades per subject modelled using probabilistic Datalog.

```
# Input data about past year student grades
grade(John, B, Art);
grade(Mary, B, Maths);
grade(Anna, A, Art);

# Input data for new enrolled students
 register (Matt, Art);
 register (Mike, Maths);

# Probabilities of grades for each of the new students
0.5 p_grade_person (A, Matt);
0.5 p_grade_person (B, Matt);
1.0 p_grade_degree (B, Mike);
```

Figure 3.6: Example data and results for the student grade prediction based on historic average grades per subject modelled using probabilistic Datalog.

3.1.2 Descriptive Approaches for Classification

Probabilistic Datalog has been used to model rules for information classification and transformation [Nottelmann and Fuhr, 2001]. This work focuses on extracting probabilistic rules for information classification and matching, and they also propose different methods to adapt their approach to problems with higher dimensions such as text classification.

Logic programming and domain ontologies have been combined for text classification, where logic is used as the categorisation rule language, and the ontologies are applied as the formal representation of the domain knowledge [Cumbo et al., 2004]. The authors use Datalog^f, a variation of Datalog that includes aggregate functions [Dell'Armi et al., 2003]. This work shows how the expressive power of descriptive approaches can capture the semantics of the taxonomy and describe complex patterns that should be present in the documents. In addition, they illustrate that the modelling of such solutions (referred to as encoding in their work) is very concise, simple and elegant because of the abstraction based on logic.

Outside the textual domain, the exploitation of first-order logic rules, obtained using Inductive Logic Programming (ILP) [Muggleton, 1991], has been explored to define musical genres based on harmony [Anglade et al., 2009]. The same authors successfully applied such rules for genre classification using rules that are *"transparent, human-readable and more meaningful than previous representations"* [Anglade and Dixon, 2008]. A continuation of this research suggests that *"this [abstraction and first-order logic rules] approach bring us one step closer to modelling music in the way it is conceptualised by musicians"* [Dixon et al., 2011].

In addition to using descriptive approaches for classification, there are other similarities between these approaches and the work presented in this thesis. For instance, [Nottelmann and Fuhr, 2001] apply the same language (although not the same "generation" as we shall see in the next section) and engine (HySpirit [Fuhr and Rölleke, 1998]) as we do. The strategies presented in [Cumbo et al., 2004] such as exploiting a synonym predicate can be seen as knowledge-enhanced expert rules. This is related to the modelling of knowledge-enhanced classification (see Section 5). These investigations also support the applicability of descriptive approaches in order to provide abstraction and high-level definitions. In this sense, the most similar research compared to this thesis are [Anglade et al., 2009] [Dixon et al., 2011]. Nonetheless, this thesis is also focused on flexibility and adaptability.

The main differentiation with respect to our work is that the methods presented on this section are focused on rule-based classification, specialising either on how to learn the rules, or on how to apply them for classification. Whereas, we address the challenge of model traditional classifiers and knowledge.

3.1.3 Descriptive Approaches based on Datalog and Probabilistic Datalog

Probabilistic Datalog has been used as a probabilistic logical retrieval framework [Fuhr, 2000]. In addition, probabilistic Datalog was extended (Probabilistic Datalog++) to create a extensible information retrieval engine [Nottelmann, 2005]. As this research shows, one of the major advantages of logical frameworks in information retrieval is that external knowledge such as term associations, hyperlinks, annotations or contextual information can be easily incorporated into the retrieval process. The reason to incorporate a new variation of the language was the lack of features such as aggregation operators and the possibility of specifying probabilistic assumptions between events (e.g., independence). Similar modifications, among others, were added in the 2nd generation of probabilistic Datalog [Roelleke et al., 2008] (explained in Section 3.1.1).

Another information retrieval task where this approach has proven to be beneficial is annotated document retrieval. The main goal of this task is to exploit the knowledge from annotations to improve document retrieval. To address this challenge, a Probabilistic Object-oriented Logics for Annotation-based Retrieval (POLAR) was introduced [Frommholz, 2007]. POLAR is based on four-valued (true, false, inconsistent and unknown information) probabilistic Datalog [Fuhr and Rölleke, 1998]. Further development of this research allowed to apply these concepts, in combination with machine learning techniques, to the task of polarity detection in discussion search [Frommholz and Lechtenfeld, 2008].

Summarisation has also been addressed using an abstraction based on probabilistic Datalog, referred to as POLIS [Forst et al., 2007b]. One of the main conclusions of this work are that POLIS shows the advantages of abstraction such as robustness, flexibility and re-usability. In addition, it allows experienced users to express *how* a summary should be generated, rather than implement the summarisation models.

The applicability of probabilistic Datalog for expert search was also investigated, concluding that it is possible to express strategies for such task in a simple, straightforward way, without making any changes to the underlying system [Forst et al., 2007a].

Logical abstractions have also been used to model semantic search, where a knowledge-oriented approach was presented to facilitate the transformation of term-based retrieval models into semantic-awared ones [Azzam and Roelleke., 2010]. Their approach consists of semantic propositions such as relationships and classification of objects. The authors illustrate how their framework acts as a logical layer that decouples the retrieval models from the physical representation of the data, therefore, achieving data independence.

Patent retrieval presents a complex environment, where traceability, abstraction and descriptive power are useful. Logic-based retrieval (based on probabilistic Datalog) has been used to address this task, showing that the *open-box*, high-level abstraction provided by logical retrieval is superior in such environments [Klampanos et al., 2010]. Patent retrieval requires reasoning about objects and modelling retrieval strategies in such way that users (patent searcher) can understand and modify the ranking and the reasoning process. This research illustrates that probabilistic Datalog has such abstraction and a similar nature to the query languages used by Intellectual Property professionals. Klampanos *et al.*also address the scalability challenge by applying distributed information retrieval approaches, showing how to seamlessly apply data source selection and result fusion within a logical retrieval system.

Traditional Datalog has also been applied for information extraction to provide a cleaner and more powerful way to *"compose small extraction modules into larger programs that are easy to understand, debug and modify"* [Shen et al., 2007]. In particular, Datalog is extended to include embedded extraction predicates. The authors also show how to apply query optimisation techniques to the definitions of information extraction solutions in order to improve the scalability of their solution.

3.2 Knowledge-Enhanced Text Classification

Knowledge has been applied in text classification in numerous occasions, mainly for the document representation and scoring steps. This section introduces different knowledge-enhanced strategies that have been proposed in the literature. The term knowledge-enhanced classifiers is used throughout this thesis to refer to any model that considers more than just the words in the documents in the classification process.

3.2.1 Exploiting Term Dependencies

The traditional document representation based on a bag of words implies that learning algorithms are *"restricted to detect patterns in the used terminology only, while conceptual patterns remain ignored"* [Bloehdorn and Hotho, 2004]. Term dependencies have been proven to be beneficial both to change the weights of terms present in the document, and to augment such representation with relevant, yet not present, terms. Both strategies are explained bellow.

3.2.1.1 Boosted Term Weights

One strategy to incorporate knowledge is to modify the importance of certain terms, for instance, based on the structure of the document. Following this approach, the weight of terms that appear in parts of the document that are considered more important are boosted. The title of a document, or the first paragraph are common cases where information is assumed to be more important [Robertson et al., 2004]. In addition, richer document representations allows more detailed strategies. For example, exploiting emails headers (e.g., From and Subject) for spam detection [Klimt and Yang, 2004]. There are two main techniques to boost the importance of a term. On one hand, the number of occurrences of the term can be artificially increased, as suggested by [Robertson et al., 2004]. On the other hand, the feature weight can be modified to reflect more importance [Ogilvie and Callan, 2003].

3.2.1.2 Augmented Representation

The goal of augmented representation is to better capture the meaning of a document, instead of being limited by the terms that it contains. To achieve this objective, knowledge can be used to incorporate new terms into a document representation, usually exploiting term dependencies or similarities based on different measures such as co-occurrences or synonyms. Furthermore, these metrics can be based on external datasets. For example, the Open Directory Project (ODP) [Gabrilovich and Markovitch, 2006] or Wikipedia [Wang et al., 2007]. WordNet [Miller, 1995] is usually applied for synonym exploitation [Bloehdorn and Hotho, 2004]. Other steps such as feature selection can also be improved by exploiting such data. For instance, using data as from ODP [Gabrilovich and Markovitch, 2005] or Wikipedia [Gabrilovich and Markovitch, 2006] as background knowledge.

3.2.2 Exploiting Relationships

3.2.2.1 Link-Based and Relational Classification

Most real-world data such as the Web, bibliometric data, or social networks are heterogeneous and interconnected in nature. However, most learning methods assume “flat” data representations, therefore much of its information is lost [Getoor et al., 2003]. Link-based (and relational) learning exploits relationships between different elements for classification. It represents an intersection between the fields of link analysis, hypertext mining, relational learning and inductive logic programming and graph mining [Lu and Getoor, 2003]. Furthermore, according to some authors "*A key challenge for machine learning is to tackle the problem of mining more richly structured datasets*" [Lu and Getoor, 2003].

The foundational difference with respect to traditional content-based learning methods implies that different approaches are applied. While information retrieval and machine learning techniques are used to solve content-based classification, the methods for relational classification have their foundation in fields such as databases, data mining and artificial intelligence. The exploitation of these relations has proven to be very beneficial in domain specific scenarios. For instance web similarity search using exclusively relational information [Dean and Henzinger, 1999]. Another research illustrates how traditional classifiers perform well with collections containing independent documents such as `Reuters-21578` (explained in Section 2.8), whereas they underperform with collections where there are underlying relationships between the elements (i.e., patents and web pages) [Chakrabarti et al., 1998]. One common approach is to apply Statistical Relational Learning (SRL), an approach that combines logic and probability to learn a probabilistic logical model. SRL is usually seen as an intersection of the machine learning and inductive logic programming (ILP) communities [Getoor and Mihalkova, 2011].

The combination of relational and textual classification (in hypertext domains) was suggested in [Slattery and Craven, 1998], where the authors claim that such approach will make a more informed and accurate decision. Their results support their claims, however, their work present a combination of two different techniques rather than their integration. Furthermore, they argue that their approach is also well suited for other domains that involve both relational and large feature spaces. The combination of textual and relational features has also been investigated in the literature, where a combination of object features (i.e., based on term frequencies) and link features are used to classify new documents [Lu and Getoor, 2003]. The authors study the exploitation of link features that describe the categories of the link type based on relational information, rather than explicitly representing the much more dense link incidence matrix (i.e., the relational information between all the nodes).

In addition to the nature of data being mainly relational, another relatively recently recognised challenge is the fact that information is heterogeneous in the "real world" [Ji et al., 2010]. Both the relations and the “entities” to be classified are usually assumed to be homogeneous (i.e., the same type or nature). However, as different authors have shown, this is not the case and both relations [Cai et al., 2005] and elements in the network [Ji et al., 2010] [Deng et al., 2011] can have different types. The combination of relational and textual data in heterogeneous environments presents several challenges [Ji et al., 2010]:

- The complexity of the network structure by itself, where some authors transform it into a homogeneous network and apply traditional classification methods. However, this simplification necessarily ignores type-specific information (i.e., co-citation of papers in a bibliographical collection) by assuming a “general type” of entities.
- The transformation of link information into features requires very high dimensional and sparse data.
- The features for different types of objects will be in different event spaces.
- In heterogeneous environments labels for some elements might be easier to obtain than others, with the extreme case of some types being completely unlabelled.

3.2.3 Exploiting Class Dependencies

The knowledge derived from the relationships between classes can be very useful in classification. Although multi-label environments are the main focus of these metrics, multi-class problems can also benefit from observing and analysing the category dependencies. This section introduces the approaches that exploit this type of information in two different steps of the classification process, the scoring and the evaluation.

3.2.3.1 Class Dependency-Aware Classification

Multi-label classification problems are usually solved by transforming them into a number of independent binary classifications, where the results are aggregated to provide the set of assigned categories for each document. This strategy requires lower computational power, and it allows the application of traditional classification techniques. However, it does not exploit the relations and interactions between classes. The main strategies to incorporate the relations between classes into the classification step that have been proposed in the literature are presented below:

Label power-set reformulates the problem as a binary class problem where the new set of classes are defined as all possible combinations of the original set [Tsoumakas and Katakis, 2007]. Although effective for collections with few classes, its computational complexity impairs its applicability for collections with large number of classes.

Chain classifiers use conditional dependencies between classes to build a classifier chain that uses one binary classifier per class, linked in a chain that includes the prediction of other classifiers for the document [Read et al., 2009] [Zaragoza et al., 2011]. The mathematical formulation for this approach is shown below, where d is the document to be classified, pa defines the parents (classes on which C_i depends), and $|C|$ is the number of classes:

$$P(C_i|d) = \prod_{i=1}^{|C|} P(C_i|pa(C_i), d) \tag{3.1}$$

Other approaches include Multidimensional Bayesian Networks Classifiers [Waal and Gaag, 2007]. The main limitation of this type of classifiers is that they require much more computational power than other strategies, therefore affecting their scalability and applicability.

3.2.3.2 Evaluation with Degrees of Misclassification

Degrees of misclassification allow to assign different penalisations for different mistakes in the classification. This strategy has been applied in hierarchical and preferential classification, where explicit differences between mistakes are defined.

Hierarchical classification presents structural relations between different classes (e.g., sub-classes). Therefore, it is critical to apply some notion of partially correct assignments. In particular, measures based on category similarities and distances are used [Sun and Lim, 2001]. For instance, a lower penalisation (if any at all) will be applied if the incorrect category is the parent of the correct one.

In preferential learning, multiple degrees of relevance are specified. As a result, different mistakes should be penalised accordingly. Therefore, it requires an evaluation metrics that take into account the multi-graded relevance. While degrees of misclassification have been exploited in these two subfields, it has been almost completely neglected in traditional classification. One research has tried to answer the question of *"Can the evaluation of a multi-label classifier be improved when taking the semantic relatedness of concepts into account?"* [Deloo and Hauff, 2013] by calculating the degree of relatedness of the set of categories assigned by the system and the set labelled by the human annotators (i.e., ground truth). The authors apply the same strategy presented in [Nowak et al., 2010], where every label of both sets (assigned by the system vs labelled by humans) is matched with the label of the other set that maximises relatedness. Furthermore, they also present the results of a human study, where the correlation between the expert ratings (given by three new human experts) and two evaluation metrics (their "semantically-enhanced" method and F_1). Although the experimental setting is quite limited, mainly due to its scale (25 documents and three expert users), it supports the claim that degrees of misclassification represent better the user satisfaction.

3.2.4 Adaptive k-NN

The number of neighbours in k-NN is arguably the most important factor in the performance of the classifier. Traditionally, its value is optimised via cross-validation. Although k-NN has proven to achieve good results in several applications, including text classification [Joachims, 1998] [Yang, 2001], it has two main issues: Firstly, assuming that the same k is optimum for different classes is counter-intuitive. Secondly, if the problem involves biased class distributions, as it is common in text classification, the optimisation of the number of neighbours is dominated by the large classes [Baoli et al., 2004]. Adaptive k-NN methods address the challenges of choosing different k values depending on different parameters, instead of using a global value. For instance, a model that assigns different number of neighbours based on the number of categories [Baoli et al., 2004]. This research assumes that large categories are more influential over the estimation of k and that the category size information is enough to specify the best k per class. They present different models to estimate the best k per category. In addition, they also show two variations of k-NN for the score computation, where $NN_k(c)$ represents the set of k most similar documents for the category and $y(d,c)$ represents the assignment function for training examples (i.e., one if d belongs to the category c, and zero otherwise). These methods apply the traditional k-NN score formulation presented in Section 2.5.2 with a normalisation factor.

$$\text{score}_{\text{k-NN}}(c,d) = \frac{\sum_{d \in NN_k(c)} y(d_t,c)}{k} \tag{3.2}$$

$$\text{score}_{\text{k-NN}}(c,d) = \frac{\sum\limits_{d \in NN_k(d)} \text{sim}_k(d,d_t) \cdot y(d_t,c)}{\sum\limits_{d \in c} \text{sim}_k(d,d_t)} \tag{3.3}$$

The number of neighbours are decided as shown in Equation 3.4, where α is a non-negative integer to maintain a minimum reasonable number of neighbours, without which the performance can be unstable due to the very few neighbours being considered, $n_D(c)$ represents the number of documents labelled in class c, and k is the number of neighbours [Baoli et al., 2004]. A similar method to minimise the penalisation of small categories was also presented by [Yang et al., 2000].

$$NN_k(c) = min(\alpha + \frac{k \cdot n_D(c)}{\max_{c_i \in C}(n_D(c_j))}, k, n_D(c)) \tag{3.4}$$

A different approach focuses on the locality of the elements to be classified (i.e., the space surrounding a document) in order to select the best number of neighbours for each specific document [Wettschereck and Dietterich, 1994]. Their main assumption is that different points in the feature space account for different characteristics of the collection (e.g., noise or irrelevant features). The authors present a variation of k-NN that stores, for each training example, the k values that would classify it correctly if it was a new, unseen document (i.e., virtually moving the example from the training set to a test set). In order to classify a new document, the M nearest neighbours are selected, and the k value that would have correctly assigned most of them is assumed to be the best number of neighbours for the current document. Two variations of this algorithm are proposed to filter the documents based on the locality to those assigned to the considered category; or to those that belong to a common cluster (after clustering the collection).

3.3 Query Performance Prediction

Query Performance Prediction (QPP) refers to the estimation of the quality that will be achieved by a retrieval system in response to a specific query [Cronen-Townsend et al., 2002]. It also relates to the appropriateness of a query as an expression for a user information need. Prediction methods have been classified into two groups depending on the data used for prediction: pre-retrieval approaches, which make the prediction before the retrieval stage, and post-retrieval, which use the rankings and scores produced by the retrieval engine [Carmel and Yom-Tov, 2010].

The prediction methods researched in the literature use a variety of available data, such as the query, its properties with respect to the retrieval space [Cronen-Townsend et al., 2002], the output of the retrieval system [Carmel et al., 2006], or the output of other systems [Aslam and Pavlu, 2007].

Pre-retrieval approaches have the advantage that the prediction can be taken into account to improve the retrieval process itself. Nonetheless, they have the potential handicap, with regards to their accuracy, that the extra retrieval effectiveness cues available after the system response are not exploited. These predictors are typically based on query or collection statistics such as the inverse document frequency [He and Ounis, 2006] or linguistic properties.

Post-retrieval predictors use the retrieved results by means of the output scores or the ranked documents. Although techniques in this category provide better prediction accuracy, computational efficiency is usually a challenge for them. Furthermore, the predictions cannot be used to improve the retrieval strategies unless some kind of iteration is applied, as the output from the retrieval system is needed to compute the predictions [Amati et al., 2004] [Yom-Tov et al., 2005].

QPP techniques can be used in many applications by placing the problem from different perspectives [Amati et al., 2004] [Yom-Tov et al., 2005] [Balasubramanian and Allan, 2010] such as:

- Provide feedback to the user in order to direct the search, so that query expansion or relevance feedback techniques could then be used.
- Allow to address the problem of retrieval consistency by distinguishing poorly performing queries, where different ranking functions could be exploited based on the output from the performance predictors.
- Decide which search engine should be used, or how much weight to give it when its results are combined for distributed information retrieval systems.

The specific nature and algorithms of pre and post-retrieval QPP predictors are analysed in the next sections.

3.3.1 Pre-retrieval Predictors

Pre-retrieval query performance has been studied from two main perspectives: based on probabilistic methods (and more generally, on collection statistics), and based on linguistic approaches. Most research on the topic has followed the former approach.

Some researchers have explored inverse document frequency (IDF) and related features as predictors, along with other collection statistics. IDF can be used as a measure of the specificity of terms and thus as an indicator of their discriminatory power, displaying moderate correlation with query performance [He and Ounis, 2006]. Other authors have taken the similarity of the query into account [He et al., 2008], and the inter-similarity of documents containing query terms as a measure of coherence [Hauff et al., 2008]. Other research based on morphological, syntactic, and semantic query features have found that the only feature positively correlated with performance was the number of proper nouns [Mothe and Tanguy, 2005].

3.3.2 Post-retrieval Predictors

Some of the most effective predictors are based on language models and the clarity score, which captures the lack of ambiguity in a query with respect to a result set. These methods compute the Kullback-Leibler divergence between the query and the collection language model [Cronen-Townsend et al., 2002]. The concept of query clarity has inspired a number of similar techniques, such as comparing the divergence of query-term frequencies before and after the retrieval is exploited [Amati et al., 2004]. Other predictors are the query perturbation, that is, the difference in ranking between the original input and after the query is processed [Vinay et al., 2006]; and the Jensen-Shannon divergence [Carmel et al., 2006].

Besides query clarity, other post-retrieval predictors exploit the score distribution of the results for each query [Zhou and Croft, 2007], or the standard deviation for a subset of retrieved documents [Pérez-Iglesias and Araujo, 2009] [Cummins et al., 2011]. Similar research proposed an utility estimation framework that relies on the standard deviation of the scores (normalised query commitment) [Kurland et al., 2011].

3.3.3 Predictors Combination

Performance prediction techniques aim to model a continuous variable $\mathbf{y}$ (usually, the average precision of a query). In addition, linear regression models obtain the (linear) relationship between the quality metric and the predictor estimations [Hauff et al., 2009]. This concept can be formalised as follows, where $\vec{x}$ denotes the vector of quality predictions, such that the i-th element x_i is paired with the quality of the i-th query y_i, and β is the weight learnt by the model:

$$\vec{y} = \vec{x}\beta + \varepsilon \tag{3.5}$$

In this context, it is straightforward to combine several predictors by using multiple linear regression. This implies that the model becomes $\vec{y} = \vec{X}\vec{\beta} + \varepsilon$, where $\vec{\beta}$ is a vector of weights, and the j-th row indicates the weight corresponding to the predictor in the j-th row of the matrix $\vec{X}$, evidencing, thus, the importance of each function in the final combination. This method has successfully obtained strong performance predictors with respect to the average precision metric [Jones and Diaz, 2007] [Hauff et al., 2009].

3.3.4 Predictors Evaluation

Once the retrieval quality and the value of the performance prediction have been assessed for each query, the predictor quality is computed using an assessment function that measures the agreement between the true value of performance and the quality estimation. There are several methods to measure the quality of a performance prediction function. For instance, capturing linear relations, comparing the importance implied by the scores or the ordering given for each variable (true and estimated performance), and exploiting the implicit partitions derived from the method. The most commonly used quality function is correlation, which is traditionally measured using three well-known metrics: Pearson's, Spearman's and Kendall's correlation coefficients. Pearson's r correlation captures monotonic linear dependencies between the variables. Whereas, Spearman's ρ and Kendall's τ evaluate non-linear monotonic relationships.

Chapter 4

Descriptive Modelling of Text Classification

Available data has importantly changed in the last years, both in quantity and quality. Its amount and velocity (i.e., how quickly new data is created) has significantly increased, both in the Web and business environments. As a result, there has been an explosion of new possibilities of how this information can be exploited.

It can be argued that custom-build text classification systems do not provide methodologies to allow rapid and understandable prototyping. There are open systems/frameworks (e.g., Weka [Hall et al., 2009] or RapidMiner [Mierswa et al., 2006]) that are well suited if standard algorithms, evaluation, and collections are used. However, deeper changes of the core algorithms and the incorporation of new dimensions of knowledge are challenging because new applications and information usually require *"reimplementing or introducing new APIs, new query languages, and even new indexing and storage structures"* [Hiemstra and Mihajlovic, 2010]. In general, the majority of information systems commonly involve complicated knowledge transfer and maintainability processes. Moreover, significant engineering effort is needed to adapt them when new knowledge appears, or there are changes in the requirements [Hiemstra and Mihajlovic, 2010]. In addition, information systems can be seen as prototype-based environments, where new models or data need to be quickly tested. Therefore, productivity and fast changeability are important characteristics. These factors support the claim that the importance of flexibility in information retrieval systems is rapidly growing, and that traditional architectures are too rigid to allow quick modifications [Cornacchia and de Vries, 2007].

This chapter investigates the modelling of the text classification process using a descriptive approach. Probabilistic Logics allows to model more compact and shorter definitions than other approaches, and, at the same time, it minimises the gap between the mathematical concept and its modelling. The main challenges are the efficiency and the expressiveness. This research illustrates the following benefits of using a descriptive approach to model text classification:

- High-level modelling that produces models as compact as their mathematical definitions.
- High-level customisation based on a predicate dictionary, where minimum (or none) engineering effort is needed.
- Translation of models to their mathematical definition for checking and correctness validation.

We start by defining the data representation. After this, the modelling of the classification steps in Probabilistic Datalog are shown. Finally, a section illustrates how this approach can be applied for model verification and translation to mathematical definitions.

4.1 Data Representation

This research uses the knowledge representation model proposed by [Fuhr et al., 1998], and extended by [Azzam and Roelleke, 2011] [Azzam et al., 2012]. Figure 4.1 shows the main relations of this schema.

Relation	Attributes
term	(Term, DocId)
type	(TypeName, Object, Context)
class	(CName, Object, Context)
relationship	(RName, Subject, Object, Context)
attribute	(AName, Object, Value, Context)
part_of	(SubObject, SuperObject)

Figure 4.1: Generic, Object-Oriented Knowledge Representation Schema.

The knowledge representation schema shows the standard information retrieval representation, "term (Term, DocId)" to capture occurrences of a term in a document. Also, there are relations regarding the object-oriented content modelling. This includes relations to model object classification (defining types of entities), relationships, attributes, and aggregation. The classification-

Relation	Attributes
test_term_orig	(Term, DocId)
rep_term	(Term)
test_term	(Term, DocId)
train_term	(Term, DocId)
score	(Class, DocId)

Figure 4.2: Classification-Oriented Knowledge Representation Schema.

```
# Document space
is_document FIRST(Doc) :− part_of(Doc, Class);

# Class space
 is_class  FIRST(Class) :− part_of(Doc, Class);

# Term space
 is_term  FIRST(Term) :− term(Term, Doc);
```

Figure 4.3: Modelling of Common Data Representation.

oriented representation schema extends the concepts previously explained with predicates that are specific for the problem of classification (see Figure 4.2).

The main predicates for the classification process are `term` (term and train_term refer to the same information, the terms occurrences in train documents) and `part_of`. They represent the terms that appear in documents and the document-category assignment. Furthermore, other commonly used predicates represent the document, class and term spaces as shown in Figure 4.3. Other common predicates used by several models such as the terms contained in a document or the document frequency for each term are illustrated in Figure 4.4.

Figure 4.5 illustrates a document example that belongs to the categories Finance and Tourism. Figure 4.6 shows the same information represented using probabilistic Datalog.

```
## Commonly used predicates
# Binary occurrence of a term in a document
bin_term FIRST(Term, Doc) :− term(Term, Doc);

# Terms contained in a document
contains(Doc, Term) :− bin_term(Term, Doc);

# Document Frequency
df SUM(Term) :− bin_term(Term, Doc) & is_document(Doc) | ();

# IDF
inv_df INV(Term) :− df(Term);
idf LOG(Term) :− inv_df(Term);
```

Figure 4.4: Modelling of Commonly Used Predicates.

```
## Example of content for document "d1"
UK tourism hits record 12 months after Olympics
 Britain welcomed more than one visitor every single second in June, as the reputation built up
during last years Olympics games helped to attract record numbers of tourists , official figures show.
```

Figure 4.5: Document Example.

```
## Data representation for sample document "d1"
# Term representation
term(uk, d1)
term(tourism, d1)
term( hits , d1)
term( record, d1)
term(months, d1)
term(olympics, d1)
 ...
term(olympics, d1)
 ...

# Assignment representation
 part_of (d1, finance )
 part_of (d1, tourism)
```

Figure 4.6: Data Representation Example.

In the rest of the section we will discuss the rules and usage of relations such as `test_term_origin`, the original terms in test documents and `rep_term` to model the representative terms (i.e., selected features). Both predicated lead to `test_term`, that represents the filtered features for the test documents. Moreover, there are relations regarding training documents and scores.

4.2 Feature Selection

Feature selection reduces the number of features that are considered in the classification, according to a representativity measure (e.g., document frequency). After such subset is selected, all information referring to other parameters is ignored by the algorithms. Figure 4.7 shows the probabilistic Datalog model for feature selection using 3000 features, where the metric to measure the term representativity is left unspecified to be parameterised using a configuration model. An example of a configuration file that defines the representativity as the document frequency (i.e., terms with higher values are preferred) is shown in Figure 4.8.

One of the benefits of the descriptive approach is that it is seamless to specify what specific metric should be used.

```
## Feature Selection
# 1. Select the 3000 terms with higher representative values.
_sort ( representativity );
  filtered_representativity (Term) :- representativity (Term):3000;

# 2. For each tuple, rep_term(Term) is 1.0 for the selected terms, there are no other tuples.
rep_term(Term) :- filtered_representativity (Term)|(Term);

#3. Modify the term occurrence information
term(Term, Doc) :- rep_term(Term) & term_orig(Term, Doc);
 test_term (Term, Doc) :- rep_term(Term) & test_term_orig (Term, Doc);
```

Figure 4.7: Generic Modelling of Feature Selection based on probabilistic Datalog.

```
# Define representativeness ( this rule would usually be defined in a customisation model)
 representativity (Term) :- df(Term);
```

Figure 4.8: Customisation of Feature Selection based on Document Frequency in probabilistic Datalog.

4.3 Term Weighting

Term weighting computes a value for each term to represent its importance within the context of a document. Most of the methods for term weighting are based on a combination of a within document importance (e.g., term frequency), and a value given the collection (e.g., inverse document frequency). Figure 4.9 illustrates the modelling of this approach.

```
# The within document and within collection weights are combined.
weight SUM (Term, Doc) :- document_weight(Term, Doc) & collection_weight(Term);
```

Figure 4.9: Generic Modelling Term Weighting Strategy in probabilistic Datalog.

Once the generic modelling is produced, specific algorithms can be used for document weighting. For instance, Figures 4.10 and 4.11 show how to parameterise the generic weighting schema using `tfc` and `ltc` strategies respectively (explained in Section 2.4). It can be seen how the modelling of `tfc` is almost trivial, while `ltc` requires additional steps to change the term space to include the additional occurrence required by the model.

One of the main benefits of this abstraction is that the rules are easy to adapt if a new model requires a different approach. For instance, Figure 4.12 illustrates how to apply a new weighting strategy that includes a category-based weighting factor. The great flexibility of Logics is underlined in this example, where the model is dynamically parameterised with new knowledge for which the original model was not designed.

```
## Term Frequency Count Weighting ( tfc )
# Within Doc weight applies term frequency
document_weight(Term, Doc) :− term SUM(Term, Doc);

# IDF is applied as the collection weight
 collection_weight (Term, Doc) :− idf (Term);
```

Figure 4.10: Modelling of tfc Weighting Strategy in probabilistic Datalog.

```
## Log Term Frequency Count Weighting ( ltc )
# One additional element has to be included for each term in a document
# to apply the log over the number of times a term appears in the document plus one
term_plus (Term, Doc) :− term(Term, Doc);
term_plus (Term, Doc) :− term SUM(Term, Doc) | (Term, Doc);

# Within Document weight applies log term frequency
 log_tf LOG(Term, Doc) :− term_plus SUM(Term, Doc);
document_weight(Term, Doc) :− log_tf (Term, Doc);

# IDF is applied as the collection weight
 collection_weight (Term, Doc) :− idf (Term);
```

Figure 4.11: Modelling of ltc Weighting Strategy in probabilistic Datalog.

```
## Extension of term−document weighting that includes term−category information
# Weighting strategy is overridden to include category information
weight(Term, Doc) :− document_weight(Term, Doc) & collection_weight (Term, Doc) & category_weight(Term)

# Within document weight applies term frequency
document_weight(Term, Doc) :− term SUM(Term, Doc);

# IDF is applied as the collection weight
 collection_weight (Term, Doc) :− idf (Term);

# The category weight is based on chi square statistic
category_weight (Term) :− chi_square (Term);
```

Figure 4.12: Modelling of Weighting Strategies that include Category Information in probabilistic Datalog.

4.4 Traditional Classifiers

Traditional classifiers use a broad range of concepts and techniques for their score computation. As a result, some classifiers are easier to model than others using probabilistic Datalog. This section shows the modelling of two traditional families of classifiers, namely Naive-Bayes classifiers and *kNN*. In addition, the modelling of polynomial kernel functions is also shown.

4.4.1 Naive Bayes

Figure 4.13 shows the general modelling of a Naive-Bayes classifier. The definition assumes that the probabilities of terms given classes (*p_t_c*) has been defined. This allows the seamless parameterisation of the classifier using different probability estimations or different smoothing techniques.

```
## Bayesian Classifier in PDatalog
# Likelihood: P(d|class) = prod_{t in d} P(t|class)
p_likelihood PROD(Doc, Class) :- test_term(Term, Doc) & p_t_c(Term, Class);

# Evidence prior: P(doc)
p_doc(Doc) :- is_doc(Doc) | (); # P(doc) = 1/N_Docs

# Hypothesis prior: P(class)
p_class SUM(Class) :- part_of(Doc, Class) & p_doc(Doc);

# score = prior * likelihood = P(class) * P(d|class)
score_bayes(Class, Doc) :- p_class(Class) & p_likelihood(Doc, Class);

# Normalisation: score / sum_i P(class_i | d)
norm_score_bayes(Class, Doc) :- score_bayes(Class, Doc) | (Doc);
```

Figure 4.13: Generic Modelling of Naive-Bayes classifier in probabilistic Datalog.

Figure 4.14 presents an example of how the probability of terms given classes can be estimated. The model represents location based probabilities with Laplace smoothing. It contains the rules that define the within-class term probability $P(t|c)$. Furthermore, it also configures the document prior, and the aggregation of the term probabilities to obtain $P(d|c)$. This example illustrates one of the main limitations when using Datalog in particular, and logic based systems in general. A straight forward mathematical operation such as add one "virtual" occurrence of each term in every class has to be modelled as a combination of the original occurrences with a joint between all the terms and classes known in the system.

These examples illustrate the level of abstraction that can be obtained by our approach, and its

```
## Modelling of Laplace smoothing in PDatalog
term_class (Term, Class) :- term(Term, Doc) & part_of (Doc, Class);

# One tuple for each term-class pair.
 all_term_class (Term, Class) :- is_term (Term) & is_class (Class);

# term_class event space with laplace correction
 term_class_laplace (Term, Class) :- term_class (Term, Class);
 term_class_laplace (Term, Class) :- all_term_class (Term, Class);

# P(t | c). Probability computation based on the Laplace space of terms in documents
p_t_c (Term, Class) :- term_class_laplace SUM(Term, Class) | (Class);
```

Figure 4.14: Modelling of Laplace correction in probabilistic Datalog.

similarity with the original mathematical formulations (see Equations 4.5- 4.7).

4.4.2 k-NN

Figure 4.15 shows the Probabilistic Datalog modelling of a generic *k*-NN classifier that uses 45 neighbours based on a customisable similarity measure. The score is computed as the sum of similarity scores of the training documents that belong to the respective class. The cosine similarity is the most common approach to measure the similarity in *k*-NN. Figure 4.16 shows its modelling, where the first two rules describe the Euclidean normalisation for each vector. Then, there is a rule to model the cosine similarity as the product of the normalised vectors.

```
## KNN-Classifier in PDatalog
# Specification of top-k documents.
 top_similarity (TrainDoc, TestDoc) :- similarity (TrainDoc, TestDoc):45;

# Score is the sum of normalised (per document) similarity scores:
score_knn SUM(Class, TestDoc) :-
          top_similarity (TrainDoc, TestDoc) | (TestDoc) & part_of (TrainDoc, Class);
```

Figure 4.15: Modelling of *k*-NN in probabilistic Datalog.

```
## Cosine similarity , given two sets of feature weights
# Tuple probabilities based on the Euclidean norm 1/sqrt(vec(Doc)^2).
 vec_train (Term, Doc) :- weight(Term, Doc) | EUCLIDEAN(Doc);
 vec_test (Term, Doc) :- weight(Term, Doc) | EUCLIDEAN(Doc);

# Product for the common terms based on the euclidean normalisations to produce the final score
cosine SUM(TrainDoc, TestDoc) :- vec_test(Term, TestDoc) & vec_train (Term, TrainDoc);
```

Figure 4.16: Modelling of Cosine Similarity in probabilistic Datalog.

4.4.3 Kernel Methods

Central to SVMs and other Kernel methods such as Kernel PCA is the use of a Mercer Kernel as a non-linear scalar product. Using a Mercer kernel $K(\vec{u},\vec{v})$ implicitly amounts to carrying out a non-linear mapping Φ of vectors $\vec{u}$, $\vec{v}$ from the original data space $\mathbb{R}^n$ into a higher–dimensional feature space $\mathbb{F}$, in which the standard scalar product is computed: $K(\vec{u},\vec{v}) = \Phi(\vec{u}) \cdot \Phi(\vec{v})$ [Courant and Hilbert, 1943]. The advantage is that the mapping Φ never needs to be computed explicitly, which allow handling high-dimensional or infinite-dimensional feature spaces $\mathbb{F}$ efficiently and transparently. Figure 4.4.3 illustrates the implementation in Probabilistic Datalog of two kernels widely used in SVM classifiers, belonging to the polynomial family $K(\vec{u},\vec{v}) = (\vec{u}\cdot\vec{v}+c)^d$. Namely, we implement the homogeneous quadratic kernel $K_H = (\vec{u}\cdot\vec{v})^2$ and the non-homogeneous quadratic kernel $K_N = (\vec{u}\cdot\vec{v}+1)^2$. In the case of a two–feature data space $\mathbb{R}^2$ we can make the mapping Φ explicit by expanding the definition of the kernel:

$$
\begin{aligned}
K_H = (\vec{u}\cdot\vec{v})^2 &= (u_1v_1 + u_2v_2)^2 && (4.1)\\
&= (u_1^2v_1^2 + 2u_1u_2v_1v_2 + u_2^2v_2^2) = \Phi_H(\vec{u}) \cdot \Phi_H(\vec{v}) && (4.2)
\end{aligned}
$$

from which we conclude that the implicit map is:

$$\Phi_H(\vec{u}) = (u_1^2, \sqrt{2}u_1u_2, u_2^2) \qquad (4.3)$$

that transforms vectors in the data space $\vec{R}^2$ into vectors in $\mathbb{F}_H = \mathbb{R}^3$. Similarly, by expanding the definition of K_N it can be seen that $\mathbb{F}_N = \mathbb{R}^6$:

$$\Phi_N(\vec{u}) = (u_1^2, \sqrt{2}u_1u_2, u_2^2, \sqrt{2}u_1, \sqrt{2}u_2, 1) \qquad (4.4)$$

Polynomial kernels implicitly compute the correlation and higher order moments of the original features, which arguably explains their effectiveness. In the general case of an n–dimensional data space $\mathbb{R}^n$ we have $F_H = \mathbb{R}^{n(n+1)/2}$ and $F_N = \mathbb{R}^{(n+1)(n+2)/2}$.

This line of research represents the first attempt to bring probabilistic Datalog to model kernel models and SVM. However, the lack of iteration in the language itself makes it unfeasible without deeper changes in the language and the engine.

```
## Polynomial kernel
# 1. Modelling the dot product between U and V and a constant (1)
kernel_match(Feature, U, V) :- value(Feature, U) & value(Feature,V);

# If the next line is ignored/commented the homogeneus kernel will be computed
kernel_match SUM (Feature, U, V) :- constant(Feature, U, V);

# 2. Aggregate of product and constant (UV + 1)
sum_kernel_match(Feature, U, V) :- kernel_match SUM (Feature, U, V);

# 3. Quadratic polynomy (UV + 1)^2 or (UV)^2
K_quad_polyn(U, V) :- sum_kernel_match(Feature, U, V) & sum_kernel_match(Feature, U, V);
```

Figure 4.17: Modelling of quadratic Mercer kernel in probabilistic Datalog.

```
## Modelling of evaluation measures
# Precision: correctly classified / total classified
precision(Class) :- classified(Doc, Class) | (Class) & part_of(Doc, Class);

# Recall: correctly classified / total labelled
recall(Class) :- classified(Doc, Class) & part_of(Doc, Class) |(Class);

# F1 computation
2.0 constant();
f1_norm(Class) :- recall(Class);
f1_norm(Class) :- precision(Class);
inv_f1_norm INV(Class) :- f1_norm SUM(Class);

f1(Class) :- constant() & precision(Class) & recall(Class) & inv_f1_norm(Class);
```

Figure 4.18: Modelling of Evaluation Measures in probabilistic Datalog.

4.5 Modelling of Evaluation

The evaluation of classifiers are usually based on ratios between correctly classified documents. For instance, the precision of a class is the probability of a random document being correctly classified, given that the system has assigned it to the category. On the other hand, recall can be seen as the probability of being correctly classified, given a labelled document.

Figure 4.18 illustrates how these type of metrics, based on ratios, can be easily modelled and combined using probabilistic Datalog.

4.6 Task Customisation

A dictionary is used to store all the available predicates and their input/output information and requirements. For instance, given a weight modelling, this dictionary can specify that the relations *weight_document(Term, Doc)* and *weight_collection(Term)* are needed, and that they should

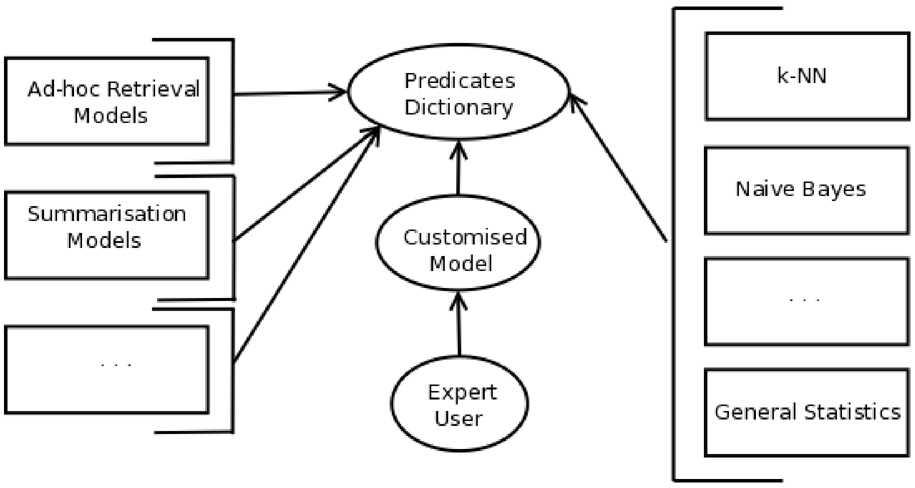


Figure 4.19: Dictionary-Based Architecture.

be mapped to the specific methods configured by expert users. Figure 4.19 illustrates a dictionary architecture with different predicates being obtained from specific models for classification and other related tasks. Figure 4.20 illustrates a dictionary that represents part of the predicates that exist in the system. Each one of them can be overwritten. Furthermore, each one of them can potentially be used to define new rules.

```
term(Term, Doc): Occurrences of terms in documents

part_of (Doc, Class): Document−class labels

p_c_d_bayes(Class, Doc): Score for class −document using NB classifier
        p_t_c (Term, Doc) has to be specified for the estimation of P(t|c)

cosine(Doc1, Doc2): Similarity score based on cosine distance
        final_test_weight (Term, Doc) is needed for measuring the importance of terms in test
            documents
        final_weight (Term, Doc) is needed for computing the importance of terms in train
            documents

score_knn(Class, Doc): Score for class −document using k−NN classifier
        top_similarity (Doc1, Doc2) is needed modelling the k most similar documents.
```

Figure 4.20: Predicate Dictionary Example showing the Description and Requirements of Different Predicates.

As a result, both the customisation of strategies and their applicability to address new problems is possible with minimum engineering effort, and without any modification of the core models. The main difference with other systems is that this is true at every level of the system, instead of at specific configuration points.

4.7 Correctness of Probabilistic Datalog Programs

One challenge of software development is to be able to check and verify the models represented in a program. A logical abstraction, and in particular Probabilistic Datalog, allows a translation between the implementation and a mathematical formula. Therefore, increasing the possibilities of software verification. The translation process is based on different operators such as relationships interaction (joins), probability aggregation (sum and product), probabilistic estimation (relational Bayes) or order-based (sort, first). In order to compare and translate the logical modelling and its translation, each operator should be able to be independently translated. As a result of the high level definition and the possibility to translate each operator, an iterative automatic translation process could be build, based on these definitions, in order to obtain the mathematical definition of a specific solution. The common notation that is followed throughout this section is presented in Table 4.1.

Symbol	**Explanation**	**Example**
x	atom	term(Term, Doc)
$r(x)$	relationship	term
$A(x)$	parameters for atom x	[Term, Doc]
$T(x)$	set of tuples of atom x	[term(car, d1), term(race, d1)...]
τ_a	value for attribute a in tuple τ	car
$w(\tau)$	probability of tuple τ	0.9

Table 4.1: Notation Summary for the Translation from probabilistic Datalog to Mathematical Formulation.

For each of the operators, two translation decisions have to be made: which tuples to keep or generate, and their probability computation. For instance, the rule `term(Term, Doc) & part_of(Doc, Class)` will keep all tuples that have a common document, while computing the new probability as the product of each pair of tuples, one from each relationship, that follows that pattern (e.g., `term(car, d1)` and `part_of(d1, sports)`). Therefore, a rule following the pattern r_1`(A(`x_1`) &` r_2`(A(`x_2`))` implies that any tuples from r_1 and r_2 that share the same values for the attributes that the rule required will be selected, while their probabilities will be multiplied. A similar set of rules for each of the main operators is shown in Table 4.2.

For instance, Equations 4.5- 4.7 illustrate one formulation of Naive Bayes, where Laplace smoothing is applied. Firstly, the probability of a term given the class is obtained, applying maximum likelihood in the term space, where $n_L(t,c)$ represents the number of times a term appears in a

Operator	**Operations**	**Condition**
Join	$\tau_n = \tau_1$ $w(\tau_n) = w(\tau_1) \cdot w(\tau_2)$	$\tau_1 \in T(x_1), \tau_2 \in T(x_2) \vert \forall a \in A(r_1) \cap A(r_2) \wedge (\tau_{1_a} = \tau_{2_a})$
Sum	$\tau_n = \tau'$ $w(\tau_n) = \sum_{\tau' \in T} w(\tau')$	$T' \subset T(x) \vert \forall a \in A(x) \wedge \forall \tau' \in T' \wedge (\tau'_{1_a} = \tau'_{2_a} = \ldots = \tau'_{n_a})$
Prod	$\tau_n = \tau'$ $w(\tau_n) = \prod_{\tau' \in T} w(\tau')$	$T' \subset T(x) \vert \forall a \in A(x) \wedge \forall \tau' \in T' \wedge (\tau'_{1_a} = \tau'_{2_a} = \ldots = \tau'_{n_a})$
Log	$\tau_n = \tau$ $w(\tau_n) = \ln w(\tau)$	-
First	$\tau_n = \tau'$ $w(\tau_n) = w(\tau)$	$\arg\max w(\tau)$
Top (k)	$\tau_n = \tau'$ $w(\tau_n) = w(\tau')$	$\tau' \in T' \subset t(r) \vert \forall \tau_1 \in T', \tau_2 \notin \tau_{1_a} \geq \tau_{2_a} \wedge \vert T' \vert = k$

Table 4.2: Translation for the main operators of 2nd generation probabilistic Datalog, τ_n is a new tuple.

class and T represents the set of unique terms. After this probability distribution is estimated, $P(d|c)$ and $P(c|d)$ are computed using the traditional definitions, assuming term independence and applying the Bayes rule. On the other hand, Figure 4.21 shows the modelling of the same mathematical model in probabilistic Datalog. It can be seen that the translation between both representations is almost perfect. The main differences are the modelling of the smoothing strategy and the space selection of aggregation functions. In the probabilistic Datalog modelling, the Laplace-smoothing is applied by adding one tuple for each term and class to the space that models the terms in classes (`term_class`). Then, the term space for the document ($t \in d$) used in the product aggregation in Equation 4.6 is translated (in the line 4) as an intersection with respect to the predicate `t_in_doc`. Both expressions are semantically equivalent, and an automatic translation will infer that the product iterates over the attribute T, but the condition would be specified as $t \in$ `t_in_doc` $\wedge\, t \in$ `p_t_d`, because these are the two predicates involved in the operation. This translation is also equivalent to $t \in$ `t_in_doc` because the second condition (which implies having previously seen the term in the collection) is implicit in the mathematical definition. The predicate *p_t_c* is based on the relational Bayes operator, where the probability of each tuple is divided by the sum of probabilities of all tuples following the condition that they belong to the same class.

$$P_{Laplace}(t|c) = \frac{1 + n(t,c)}{|T| + \sum_{t_i \in T} nL(t_i, c)} \tag{4.5}$$

$$P(d|c) = \prod_{t \in d} P(t|c) := \prod_{t \in d} P_{Laplace}(t|c) \tag{4.6}$$

$$P(c|d) = \frac{P(d|c) \cdot P(c)}{P(d)} \tag{4.7}$$

```
## Compact modelling of Naive−Bayes with Laplace smoothing
# P( t | c ) using Laplace correction
 term_class_laplace (Term, Class):− term_class (Term,Class);
 term_class_laplace (Term, Class) :− is_term (Term) & is_class (Class);
 p_laplace_t_c  SUM(Term, Class) :− term_class_laplace (Term, Class) | (Class);

# P( c | d )
p_d_c  PROD(Doc, Class) :− t_in_doc(Term, Doc) & p_laplace_t_c (Term, Class);
p_c_d(Class, Doc) :− p_c(Class) & p_d_c(Doc, Class) & p_d INV(Doc);
```

Figure 4.21: Stand-Alone Modelling of Naive Bayes in Probabilistic Datalog with Laplace Smoothing.

4.8 Evaluation

This section evaluates the descriptive modelling of traditional text classification from two different perspectives: Firstly, a quality evaluation is performed to confirm that it achieves the same quality as other approaches. Secondly, efficiency and scalability values are reported to show that real-scale collections can be processed in a reasonable amount of time.

4.8.1 Set-up

The experiments use the `20-newsgroups` and `Reuters-21578` collections. A (shared) server running CentOS 5.7 with two quad-core X5570 CPU at 2.93GHz and 48 GB of RAM, and the engine HySpirit [1] [Rolleke et al., 2001] have been used for the executions. All the experiments have been run using a single thread. In all cases SCutFBR.1 has been used as the thresholding strategy [Yang, 2001], based on micro-averaged F_1. Documents are represented using either `tfc` or `ltc`.

The experiments show several candidates based on different parameters and classifiers (k-NN or NB). For the case of k-NN, the weighting algorithm (`ltc` or `tfc`) and the number of neighbours are indicated. The Naive Bayes naming convention includes the assumptions of proba-

[1]version 2.9.8

Table 4.3: Reuters-21578 Micro and Macro-averaged F_1 quality values for different models, optimised using SCutFBR.

Model	MicroF1	MacroF1
bayes_prod_laplace_norm	0.7843	0.3340
bayes_prod_laplace_uniform_norm	0.7839	0.3336
bayes_sum_log_laplace_norm	0.7311	0.3177
knn_45_ltc_cosine	0.8514	0.5576
knn_45_tfc_cosine	0.8334	0.5344
baseline knn	0.8495	0.6245
baseline bayes	0.7250	0.3547

Table 4.4: 20-newsgroups Micro and Macro-averaged F_1 quality values for different models, optimised using SCutFBR.

Model	MicroF1	MacroF1
bayes_prod_laplace_norm	0.7526	0.7655
bayes_prod_laplace_uniform_norm	0.7526	0.7656
bayes_sum_log_laplace_norm	0.6800	0.6982
knn_45_ltc_cosine	0.7455	0.7590
knn_45_tfc_cosine	0.6902	0.7039
baseline knn	0.7644	0.7586
baseline bayes	0.7968	0.7907

bility estimation, smoothing, probability priors, and normalisation (in that order). For instance, `bayes_prod_laplace_uniform_norm` represents a Naive Bayes classifier which applies a normalised value per document computed as the product over the values of $P(t|c)$ with Laplace smoothing and uniform class probability ($P(c) = \frac{1}{|C|}$).

4.8.2 Quality Evaluation

Tables 4.3 and 4.4 illustrate the quality evaluation for different configurations using k-NN and NB classifiers. The results are provided to show that descriptive approaches can achieve the same results that have been reported in the literature. The baselines reported in the table are obtained by our own implementation of k-NN and the Weka implementation of Naive Bayes [Hall et al., 2009].

This provides empirical confirmation for the model correctness. As expected, the difference between micro and macro-averaged F_1 in Reuters-21578 is significant, due to the large difference between the number of documents per class. On the other hand, such difference is minimal in 20-newsgroups.

Table 4.5: Efficiency Comparison between probabilistic Datalog and a Java-based Engine for Text Classification.

		Probabilistic Datalog Engine		Java/Weka Engine	
Collection	Classifier	train(min)	test(s/doc)	train(sec)	tests(s/doc)
Reuters-21578	NB_3000	15m 25s	0.32s	03.6s	0.001s
	kNN_60_3000	15m 46s	0.80s	02.5s	0.009s
20 newsgroups	NB_3000	33m 52s	0.16s	10.34s	0.001s
	kNN_60_3000	39m 04s	1.17s	04.49s	0.013s

4.8.3 Efficiency and Scalability Study

To compare the execution of probabilistic Datalog programs against another system, a Java-based engine (using Weka libraries, [Hall et al., 2009]) was built. The probabilistic Datalog engine can be described as a light-weight DB system,[Rolleke et al., 2001] being based on persistent data like a traditional DB management system, but with special index support and processing techniques, regarding probabilistic reasoning. The purpose-built Java-based engine is main-memory based. It loads all training and test data to memory and executes the classification of all test documents in one go. Differently, the probabilistic Datalog engine executes document by document, and retrieves probabilities and tuples from a persistent knowledge base. From this point of view, the comparison of processing times does not really tell a detailed story, since an I/O-intensive system is compared with a main-memory-based solution. Only one representative per model is represented in the table due to the minimal changes between the specific configurations within a classification family. Table 4.5 shows the processing times for indexing (including all the knowledge representation) and classification. The efficiency results show for the two data sets, and for the two classifiers. Our purpose-built Java-based classification engine is able to process documents in milliseconds.The probabilistic Datalog engine is started in a batch document-by-document, and the average time is around or less than one second per document. Although this is significantly slower than the purpose-built system, it is acceptable for a prototyping environment. Whereas in early Probabilistic Datalog implementation, the creation of the knowledge base and indexing phase took several hours, and the classification took minutes, the indexing phase now takes 15-40 minutes, and the classification is in most cases sub-second. In classification, the real-time requirements are usually less critical than in ad-hoc retrieval. However, it is essential to achieve a reasonable throughput, even though the classification of different documents is independent, and it can be processed in parallel.

4.9 Summary

This chapter has shown the benefits of modelling classifiers using a descriptive approach. The compact high-level definitions and customisation lead to a flexible framework where expert users can model specific strategies with minimum engineering effort. In addition, it allows to verify the correctness of the models by translating them to a mathematical formulation.

One of the main limitations of this approach is the impossibility of model iteration within the framework. As a result, some optimisation methods cannot be addressed. The main two implications are that the thresholding process has to be partially externalised (i.e., to iterate over the set of scores) and that the modelling of SVM is not feasible at the moment. In addition, some operations such as the Laplace smoothing require a non-intuitive modelling. These factors support the idea of representing some of these operations as special functions within the language itself.

The experiments support the claim that descriptive approaches are suited for the modelling of traditional Text Classification. The results illustrate how the high level, and highly customisable approach can be applied to perform classification in a reasonable amount of time, while obtaining comparable quality values as the algebraic counterparts. The efficiency and scalability challenges are still present. Nonetheless, the current system is suited for prototyping environments. In addition, these experiments should be considered as a maximum boundary of the required time. Any improvement made to the underlying system will potentially increase its efficiency.

Chapter 5

Descriptive Modelling of Knowledge-Enhanced Text Classification

Nowadays, there is a vast amount of knowledge that can be exploited to increase the performance of classification. For instance, statistical data such as co-occurrence between terms, synonyms augmentation, or relational information (e.g., citations between scientific articles). Several models have been proposed to exploit different types of knowledge for classification, mainly for representation augmentation [Gabrilovich and Markovitch, 2006] or classification scores modification [Zelikovitz and Hirsh, 2002]. However, *"even if some algorithms have achieved improvements, a consistent and significant quality increasing has not been obtained"* [Wang et al., 2007]. Moreover, we claim that a common framework to explain and integrate such techniques is also missing. In addition, exploiting complex entities and relational data presents additional challenges that are even greater in heterogeneous environments.

These concepts are related to the Semantic Web [Berners-Lee et al., 2001] where the main goal is to interlink and communicate data from several sources and integrate it in a common knowledge base that could then be queried with specific information needs. In fact, some technologies such as RDF, OWL and RuleML are similar or even alternatives to some methods proposed in this research. A similar parallelism is studied in detail in [Hunter and Liu, 2010], where the authors investigate formalisms for representing and reasoning with scientific knowledge. Another similar research has shown how to exploit RDF ontologies in order to improve text classification by producing a ”bag of concepts” instead of the traditional ”bag of words” [Xiaoyue and Rujiang, 2009]. A deeper comparison between the semantic web technologies (i.e., RDF, OWL and SPARQL) and Datalog can be found in [Polleres, 2007].

This chapter focuses on the descriptive modelling of knowledge exploitation within the context of text classification. The first part defines a conceptual framework that generalises the methods proposed in the literature. This includes the discussion and categorisation of knowledge for augmented representation and scoring text classification. This approach leads to easier adaptation of new information needs or data sources. It divides the main two objectives of using knowledge as augmented representation and score modification. After that, this research underlines how such framework can be modelled using a descriptive approach, while keeping the same level of abstraction as the original mathematical definition and allowing flexible modifications.

The second goal is to investigate the classification of complex objects with relational data. Although this scenario is becoming more common, its modelling and exploitation still involve significant challenges due to its complexity and variability. Probabilistic Datalog uses the flexibility of Logics to be able to represent these type of “entities” and to apply decisions about their category assignments. Furthermore, following the same approach, the combination of textual and relational classifiers in heterogeneous networks is illustrated, considering different types of objects with or without textual components and relational information.

In addition to these goals, this chapter also shows the seamless combination of different information retrieval tasks within a prototyping environment.

5.1 A Framework for Knowledge-Enhanced Text Classification

Several approaches that exploit knowledge to improve classification have been proposed in the literature, mainly for the document representation or the scoring step. Most of them follow similar strategies, nonetheless, to the best of our knowledge, no common framework generalises the knowledge exploitation within the text classification task. This section introduces a conceptual framework for knowledge-enhanced classifiers. A definition and division of knowledge dimensions are explained to be used as a foundation for the framework. The main objective is to improve flexibility and adaptability when including new sources of information, and to integrate the different methods that exploit knowledge within a common framework.

Table 5.1: Examples of Knowledge-Enhanced Classifiers.

	Representation	**Scoring**
Term	Term Relationships using WordNet	Boosting based on classes names
Document	Term Boosting based on structured fields	Background Knowledge for indirect similarity between documents
Class	Class-based Weighting Schemes	Boosting based on classes names
Collection	Importance based on domain knowledge	Classifier committee based on quality

5.1.1 Knowledge Dimensions

Knowledge, from the classification perspective, is categorised in three dimensions to organise and to understand how it can be used to enhance classifiers: the *source* (internal vs external), the *objective* (representation vs scoring function), and the *entity type* (e.g., term, document, class). Table 5.1 illustrates, from the objective and the entity type dimensions, how different knowledge-enhanced strategies that have been proposed in the literature can fit into this structure.

5.1.1.1 Source: Internal or External

The source dimension defines knowledge as internal if it is based on information gathered from the collection, whereas it is external from any other source (e.g., dictionaries, WordNet or Wikipedia). The main difference between these two classes of sources is the pre-processing of the information and its nature, where external sources often provide more complex and diverse information than the collection.

5.1.1.2 Objective: Representation or Scoring

This dimension categorises approaches with respect to the objective for what the knowledge is exploited for. The two considered objectives are augmenting the documents representation, or modifying the scores produced by a classifier. These goals are referred to representation and scoring, respectively.

5.1.1.3 Entity: Term, Document, Class and Collection

Each strategy that uses knowledge focuses on one or more entity types. The main types for text classification are *term*, *document*, *class* and *collection*. However, other entity types such as user, paragraph or phrase could be used. The reason for this dimension is to investigate and generalise how knowledge is exploited in each case.

Term: As explained in Section 3.2, terms relationships have been commonly used as a source of knowledge in the literature, mainly for document augmented representation. The specific strategy to compute the similarity or dependency between terms can be based on co-occurrences [Gabrilovich and Markovitch, 2006] [Wang et al., 2007] [Shen et al., 2009] , or synonym exploitation [Bloehdorn and Hotho, 2004], among others.

Document: Documents have four main types of information: content, structure, metadata and relationships. A traditional assumption in text classification is that all the content of a document is equally important. However, its structured can also be exploited. For example, using the mail headers for email classification [Klimt and Yang, 2004]. Metadata includes knowledge such as authorship information, language, and characteristics such as length and creation date. Relationships between documents can be based on different information such as bibliographic coupling, co-citation or similarity.

Class: Knowledge obtained from classes includes the class name or the number of elements. Class relationships can also be exploited computing their dependencies or similarities using different approaches.

Collection: This includes any domain knowledge which is general to the collection such as classifier quality expectations and knowledge based on the category taxonomy.

5.1.2 Knowledge Augmentation

Based on the concepts previously explained, the knowledge augmentation can be used for two main goals: Improve the representation of documents, or modify the classification scores.

5.1.2.1 Augmented Representation

Each document is represented by a feature vector that specifies the weight of each term. Knowledge-enhanced methods can be used to modify the importance of each term for a document, and hence, the document representation. The first approach is to boost elements with specific characteristics. For instance, specific terms (e.g., those that are category names) or specific documents (e.g., those written by a specific author) can be treated differently if we assume that they are more meaningful.

The definition of "special" instances can be as flexible as needed. Following these principles, the weight augmentation based on special instances is defined as follows, where s is a function measuring how “special” an element is, and $w'(t,d)$ represents the initial weight of term t for document d.

$$w(t,d) = s(t) \cdot w'(t,d) \tag{5.1}$$

An example of the special element function $s(t)$ is shown in Equation 5.2, where terms are considered special if they appear in any of the classes names. Alternatively the same strategy can be applied for the document dimension.

$$s(t) = \begin{cases} 1 & \text{if } t \in \text{name}(c_i) : c_i \in C \\ 0 & otherwise \end{cases} \tag{5.2}$$

Another strategy is to apply dependencies between different instances of the same type. This approach can modify the feature space, as unseen terms in a document can now be part of its representation. Knowledge-augmentation based on term, and document dependencies are shown below, where $w(t,d)$ represents the original weight of term t in document d and the computation of dependencies - $P(t|t')$ and $P(d'|d)$ - remains completely flexible.

$$w(t,d) = \sum_{t \in T} P(t|t') \cdot w(t',d) \tag{5.3}$$

$$w(t,d) = \sum_{d' \in D} w(t,d') \cdot P(d'|d) \tag{5.4}$$

The term dependencies can be computed based on synonyms, co-occurrence ratios or semantic closeness, among others. In the case of documents, co-citation or annotations can be used to measure their dependencies. All these different augmentation techniques can be combined using a linear weighted combination as follows, where M represents the set of models considered for combination and $P(m)$ the weight for each model:

$$w(t,d) = \sum_{m \in M} P(m) \cdot w_m(t,d) \tag{5.5}$$

This is the usual application of the total probability theorem, and regarding terms, this approach is also known as “translation model”.

Table 5.2: Term importance augmentation based on different dimensions.

Origin	**Equation**
Term	$P_t(t\|d) = \sum_{t' \in T} P(t\|t') \cdot P(t'\|d)$
Document	$P_d(t\|d) = \sum_{d' \in D} P(t\|d') \cdot P(d'\|d)$
Collection	$P_c(t\|d) = \sum_{f \in F} P(t\|c) \cdot P(c\|d)$
Field	$P_f(t\|d) = \sum_{f \in F} P(t\|f) \cdot P(f\|d)$

5.1.2.2 Augmented Score

The score obtained by any given classifier can be modified, based on specific knowledge. For this approach, similar techniques to the ones explained for augmented representation can be applied to enhance the scores computed by a classifier. The formulations for such augmentation based on the document and category dimensions are illustrated below, where c is a category, d a document and $w(t,d)$ the importance of term t in document d.

$$score(c,d) = \sum_{d' \in D} score(c,d') \cdot P(d'|d) \tag{5.6}$$

$$score(c,d) = \sum_{c' \in T} P(c|c') \cdot score(c',d) \tag{5.7}$$

Similarly to the representation strategies, all the methods for score augmentation can be combined using a weighting linear combination:

$$score(c,d) = \sum_{m \in M} P(m) \cdot score_m(c,d) \tag{5.8}$$

Table 5.2 summarises four different definitions of term importance applying conditional probabilities and the theorem of total probability. These concepts can be generalised into more abstract concepts. The general formulation for augmented representation and score modification for a model m are shown in Equations 5.9 and 5.10 respectively, where E is the set of considered entities (i.e., term, document, collection and field) and E_i represents the set of specific elements of a given type (i.e., E_0 represents the term space).

$$w_{m,i}(t,d) = \sum_{[x \in E_i]} P(t|x) \cdot P(x|d) \tag{5.9}$$

$$score_{m,i}(c,d) = \sum_{[x \in E_i]} P(c|x) \cdot P(x|d) \tag{5.10}$$

As a results, it is possible to model any knowledge exploitation strategy that is formulated via the theorem of total probability. Each perspective provides additional information about the relevance of the term for a document, or the score for a document-category pair that can be aggregated based on the probability of the knowledge source to be relevant. The main benefit of this approach is that these complementary approaches can provide a more complete representation than any of them independently, while describing them in a common framework.

5.2 Modelling of the Knowledge Integration Framework

Classifiers should be able to evolve when new information is available or new knowledge can be exploited. Several methods have been proposed for the exploitation of knowledge. However, the challenge is to provide a general framework to describe and integrate them. This section introduces the modelling of the Knowledge Integration Framework for Text Classification that was proposed in the previous section, where knowledge-enhanced classifiers are divided according to their objective into augmented representation and scoring. They are modelled separately, while the source dimension is ignored due to the fact that after representing the knowledge, its source becomes irrelevant and they are similarly treated.

Figure 5.1 shows the rule-based modelling of the augmented representation. Each line, with the exception of the last one, shows a model to represent the weight for a term and a document following the framework for knowledge integration. It shows how to exploit the knowledge in the term, document and category spaces. In addition, the last line illustrates the weighted linear combination of the selected models (i.e., specified in a customisation step) for an aggregated final representation. As with the mathematical definition of the framework, a clear pattern can be observed for all dimensions, showing two types of rules: A boosting based on some ”type” of the entity (e.g., the term is a class name), and a boosting based on the relationship between entities at the same level. For example, in the class dimension a type could be defined as “having

```
# weight combination: Augmented weight is added using a prior
# Term Space−−
weight(Term, Doc, 'termSpecial', Type) :− weight(Term, Doc) & special (Term, Type);
weight(Term, Doc, 'termDep', Name2) :− weight(Term, Doc) & rel_term(Term, Term2, Name2);

# Doc Space−−
weight(Term, Doc, 'docSpecial', Type) :− weight(Term, Doc) & special (Doc, Type);
weight(Term, Doc, 'docDep', Name2) :− weight(Term, Doc) & rel_doc(Doc, Doc2, Name2);

# Class Space−−
weight(Term, Doc, 'classSpecial', Type) :− weight(Term, Doc) & special (Class, Type);
weight(Term, Doc, 'classDep') :− weight(Term, Doc) & rel_class (Class, Class2, Name2);

# Final aggregation with a weight for each model −−
 final_weight SUM(Term, Doc) :− prior(Name) & selected_weight(Term, Doc, Name);
```

Figure 5.1: Knowledge-Enhanced Representation.

```
# Score combination: Augmented score is added using a prior
# Term Space−−
score(Doc, Class, 'termSpecial', Type) :− score(Doc, Class) & term(Term, Doc) & special (Term, Type);
score(Doc, Class, 'termDep', Name2) :−
          score(Doc, Class) & term(Term, Doc) & rel_term (Term, Term2, Name2);

# Doc Space−−
score(Doc, Class, 'docSpecial', Type) :− score(Doc, Class) & special (Doc, Type);
score(Doc, Class, 'docDep', Name2) :− score(Doc, Class) & rel_doc (Doc, Doc2, Name2);

# Class Space−−
score(Doc, Class, 'classSpecial', Type) :− score(Doc, Class, Name) & special(Class, Type);
score(Doc, Class, 'classDep', Name2) :− score(Doc, Class) & rel_class (Class, Class2, Name2);

# Final aggregation with a weight for each model −−
 final_score SUM(Doc, Class) :− prior (Name) & selected_score (Doc, Class, Name);
```

Figure 5.2: Knowledge-Enhanced Scoring.

few training examples" with a certain probability. As a result, that rule will assign more score to classes with less training examples. In the same case, the relationship pattern, for multi-class collections, could be defined as being the overlap between classes. The abstraction in the modelling for representation and ranking helps to show and understand some patterns in the use of knowledge for text classification. Figure 5.2 shows a similar approach for the scoring objective.

Based on these models, a high level of customisation as it can be applied by specifying the relationships that should be exploited and their respective weights. Figure 5.3 illustrates how such customisation can be done for the knowledge representation. The first rule assigns a standard weighting function (i.e., TF-IDF) to be the basis of the augmented representation. This is followed by an augmentation based on the relationship between two terms. Then, there is a rule

```
## Knowledge−Enhanced Representation Modelling
# Exploitation of term statistics :
weight(Term, Doc, 'tf−idf') :− tf_idf (Term, Doc);

# Exploitation of relationships between synonyms:
rel_terms (Term, Term2, 'synonyms') :− synonym(Term, Term2)

# Exploitation of structure ( structured fields ).
0.7 p_field (' title ');
0.3 p_field ('body');
weight(Term, Doc, ' fields ', Field) :−
        term_field (Term, Field, Doc) & weight(Term, Doc) & p_field (Field);

# Boosting of terms that are class names
weight(Term, Doc, 'specialTerm', 'className') :− weight(Term, Doc) & is_class_name (Term);

# Strategy weighting (can be learned from training data):
0.4 prior ('synonyms');
0.4 prior (' fields ');
0.2 prior ('className');

# Selected strategies
selected_weight (Term Doc, Name) :− weight(Term, Doc, ' fields ');
selected_weight (Term Doc, Name) :− weight(Term, Doc, 'synonyms');
selected_weight (Term Doc, Name) :− weight(Term, Doc, 'className');
```

Figure 5.3: Knowledge-Enhanced Representation Customisation.

to describe the augmentation as derived from fields or attributes, where the prior "p_fields" is higher for important fields than less important fields. In this specific example, the title is considered more important than the body. The next rule exploits the fact that the occurrence of the class name as a word in a document is a strong indication for the classification. The last set of rules selects the weighting algorithms that will be applied.

5.3 Modelling of Object and Relational Classification

Heterogeneous networks are becoming more important as the result of the explosion of available semantic and relational data which provides complex and relational information between different types of objects. From a classification point of view, all data available should be exploited to try to improve the quality. Therefore, content, relationships and structure of different elements should be considered. However, dealing with these types of structures and information is still a challenge.

Figure 5.4 illustrates the evolution of the data, or at least our interpretation of it, from the perspective of text classification in a bibliographical collection of scientific documents. The first representation observes the collection as a set of documents with terms that have one or more assigned topics (e.g., Information Retrieval, Mathematics, Physics, ...). This representation allows topic classification of new documents uniquely based on their content. The mismatch vocabulary problem is ignored, as well as the fact that academic articles, and other documents such as patents, present a clear and defined structure, where some parts of the document are more representative (e.g., title and abstract). These limitations are addressed by the second interpretation, where the fields of the documents are considered as a source of information. Furthermore, external knowledge can also be exploited. This representation ignores any entity that is not a document and all the relationships between entities. The last perspective represents the complete data structure, for the specific case of scientific publications, where researchers write papers (using certain terms) and publish them in conferences. Moreover, all the three types of entities belong to one or more categories. Therefore, it is possible that a researcher that "belongs" to the machine learning community (i.e., his/her main topic is machine learning) publishes a paper related to information retrieval in a database conference. This approach allows to exploit all available information, and to classify any type of entity.

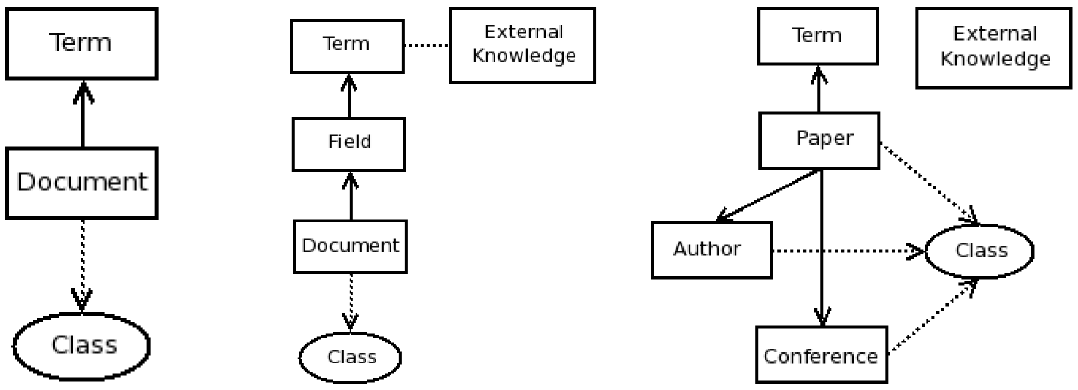


Figure 5.4: Evolution of Data Representation for Text Classification using Bibliographical Data.

Several link-based or relational classifiers have been proposed for homogeneous, non-attributed (without content) data. In addition, statistical relational learning provides the tools to classify heterogeneous and attributed data [Getoor et al., 2003]. However, *"very little research has been conducted on modelling the topics of documents as well as their associated objects simultaneously in heterogeneous networks"* [Deng et al., 2011]. The abstraction and inference capabilities from Logics allows to represent and intelligently combine the decisions obtained from both relational and content classifiers.

Figure 5.5 presents the modelling of a relational classifier that exploits the direct links with any other entity in the heterogeneous network based on conditional probability. Following this

```
## Relational  Classification  based on Direct Links.
# Category scores based on independent  relationships
rel_author (Doc, Class) :- paper_author(Doc, Author)|(Doc) & author_label (Author, Class);
rel_conf (Doc, Class) :- paper_conf(Doc, Conf) | (Doc) & conf_label(Conf, Class);
```

Figure 5.5: Relational Classification using Direct link Exploitation in Probabilistic Datalog.

```
## Relational  Classification  Customisation
# Only category information from the conference information is used.
score_rel (Doc, Class) :- relConf(Class, Doc);
```

Figure 5.6: Customisation of the Direct Link Exploitation.

approach, any relationship can be customised to be exploited. For instance, Figure 5.6 illustrates how to exploit only the venue information (i.e., only observing what conference each paper was published in) to perform the classification.

Content-based and relational information are complementary sources of information that are usually independently exploited. However, their combination could increase the classification capabilities by using all the available information. The main challenge is, as previously explained, the modelling of such knowledge and the integration of classifiers based on different foundations. We show how the abstraction based on Logics allows to seamlessly combine this two concepts.

```
## Hybrid approach to combine content-based and relational  classifiers
score_aux_hybrid (Class, Doc) :- score_rel (Class, Doc);
score_aux_hybrid (Class, Doc) :- score_text (Class, Doc);

score_hybrid (Class, Doc) :- score_aux_hybrid SUM(Class, Doc);
```

Figure 5.7: Hybrid Model Combining Relational and Content-Based Classification in Probabilistic Datalog.

Figure 5.7 shows a hybrid classifier that combines the scores of a generic content-based ($\texttt{score}_\texttt{t}\texttt{ext}$) and a relational classifier ($\texttt{score}_\texttt{r}\texttt{el}$). This example illustrates how relational and textual data can be seamlessly combined within a unique system. Such system can be easily adapted to new knowledge and information. As a result, much more advanced techniques can be applied, and what is equally important, knowledge engineers are able to create the classification strategies with minimum effort. The nature of this approach allows not only to combine a textual model with a relational one, but to use relational and content-based concepts in any method.

```
## Customisation for Hybrid approach
score_rel (Class, Doc) :- rel_conf (Class, Doc);
score_text (Class, Doc) :- p_c_d_knn_45(Class, Doc);

score_hybrid (Class, Doc) :- score_aux_hybrid SUM(Class, Doc) | (Doc);
```

Figure 5.8: Customisation of Hybrid Model.

5.4 Declarative Task Adaptation and Integration

Several strategies from different tasks in Information Retrieval (e.g., *ad-hoc* retrieval and text classification) share mathematical foundations or even similar objectives, and they can be applied together to solve complex information needs. The challenge is to provide a framework flexible enough to define models for different tasks with the ability to quickly combine them if needed. The main impact of defining different models using descriptive approaches is the capability to compose multi-task solutions and to adapt models from different tasks.

This section illustrates how our approach can be used to apply high level integration between different information retrieval tasks, and that such abstraction can be used by "knowledge engineers" to compose solutions tailored to their needs. The long term goal is to develop an information retrieval framework that supports high level integration for the main tasks in the field (*Ad hoc* retrieval, classification, summarisation, semantic search,...), while allowing the definition of multitask customisation.

As an example, and first step for such goal, the steps required to exploit text classification methods for document retrieval and vice versa are presented.

Information retrieval and text classification are two highly overlapped areas that share conceptual foundations [Belkin and Croft, 1992]. They have been applied together several times [Lam et al., 1999, Guo et al., 2004]. However, research is usually focused on each of them separately. In addition, there are techniques that have been used in both fields such as term weighting strategies.

All retrieval models rank documents with respect to a query, sorting them according to their potential relevance for the user's information need. This implies that, from a general point of view, the retrieval process can be seen as a binary classification of elements (documents) into a category (relevant or not with respect a query), if a threshold is defined. On the other hand, we can understand text classification as an *ad hoc* retrieval task where the query is the document to classify and the categories are the documents that will be retrieved. Using this strategy, we could

	Classification		***Ad-hoc* retrieval**	
	Number (docs)	Length (words)	Number (docs)	Length (words)
Input		Thousands		Few
Retrieve	Thousands	Hundreds	Hundreds of Thousands	Hundreds

Table 5.3: Dimensionality comparison between *Ad-hoc* Retrieval and Text Classification.

have, for each document to classify, a ranking of the best classes for it. Despite these similarities, viewing both tasks as a retrieval process, there are differences such as the length and nature of the queries (shorter for retrieval), the structure and representation of the documents (it needs to be defined in classification) and the number of documents (smaller in classification). Table 5.3 summarises this duality. Following this comparison, the system would rank the classes according to its relevance with respect to a given test document.

```
## TF−ICF retrieval of classes
# Term documents are modelled as queries
qterm(Term, Doc) :− test_term (Term, Doc);

# TermF based on term−class occurrences
tf_d (Term, Class) :− tf_class_sum (Term, Class);

# PIDF based on PIEF for class space
pidf (Term) :− pief (Term);

# Score normalisation
score (Class, Doc) :− tf_pidf_retrieve (Class, Doc) | (Doc);
```

Figure 5.9: TF-IDF Retrieval of Classes Modelling in probabilistic Datalog.

```
## LM−based class retrieval
# Test documents are modelled as queries
qterm(Term, Doc) :− test_term (Term, Doc);

# P(t|d) modelled as P(t|c)
p_t_d (Term, Class) :− tf_class_sum (Term, Class);

# P(t) computation
p_TF_t_c (Term) | TF() :− term_class (Term, Class); # Tuple−Frequency−based
coll_model (Term) :− p_TF_t_c (Term);

# Normalised score
score (Class, Doc) :− log_lm2_retrieve (Class, Doc) | (Doc);
```

Figure 5.10: LM Retrieval of Classes Modelling in probabilistic Datalog.

The application of retrieval models for text classification is presented as a test case to show the benefits of our approach for task adaptation. The following examples are based on already defined models of Language Modelling (LM) using Jelinek-Mercer smoothing and TF-IDF. Both models were represented using an abstraction language [Roelleke et al., 2008] that can be parameterised and used from our approach following the instructions shown in the predicate dictionary. Figure 5.9 shows the retrieval of classes based on TF-IDF (modified as TF-ICF, for inverse category frequency). To compute `tf_pidf_retrieve`, `tf_d` and `pidf` need to be represented. They model the normalised term frequency and a probabilistic version of IDF, respectively. All the changes that need to be done to adapt this strategy to category retrieval are to specify `tf_d` as the weight of a term for a class, and `pidf` as `pief` (probabilistic inverse element frequency), to compute `pidf` treating the classes as the documents. The final category-document score is normalised with respect to the test document. The customisation of Language Modelling (Figure 5.10) requires the specification of $P(t|d)$ and $P(t)$ given the collection. These values are mapped into the normalised importance of terms for classes and the probability of a term given the term-class occurrences.

5.5 Evaluation

This section illustrates the exploitation of relational information, either independently or in combination with content based classifiers, using probabilistic Datalog. The descriptive modelling for each one of the algorithms are presented in Section 5.3.

5.5.1 Set-up

The experiments use the `DBLP` collection. In all cases RCut has been used as the thresholding strategy [Yang, 2001], selecting the topic with the highest score for each element. The documents for which categorical information is available were randomly divided in two sets (50%-50%) for testing and training. The results show the average quality over five runs.

Three approaches are tested in the experiment: A relational classifier that follows the models in Figures 5.5 and 5.6; a content-based classifier based on k-NN; and a hybrid model combining their scores based on Figures 5.7 and 5.8. For the first and third cases, three variations are tested depending on the relational information to be exploited: Using the authorship links (author), the conference publication information (venue), or both of them (all).

Table 5.4: DBLP Micro and Macro-averaged F_1 quality values for different models, optimised using RCut.

Model	**MicroF1**	**MacroF1**
Relational (All)	86.63	84.57
Relational (Author)	63.92	58.61
Relational (Venue)	86.63	84.57
Hybrid (All)	88.66	86.60
Hybrid (Author)	76.30	75.17
Hybrid (Venue)	87.44	85.23
kNN 45	47.57	46.13

5.5.2 Quality Evaluation

Table 5.4 shows the quality levels achieved by different models for the classification of scientific papers. A relational and a content-based (k-NN) classifiers are compared, as well as their score combination. The results show how even a simple combination based on a score combination can outperform both the relational and the content-based strategies. The content-based classifier performs very poorly probably because of the very limited amount of training examples and content (only the title is present in the collection). The best source of information is the conference where each scientific article is published, rather than the authors of such paper. In addition, the low quality obtained by the content-based classifier is probably due to the limited textual content in the data, where only the title of the articles was used.

5.6 Summary

There are several approaches to augment the representation and/or the scoring for text classifiers, based on different types of knowledge. A structure and division of knowledge for text classification and a conceptual framework based on such structure have been presented to unify, integrate and combine different strategies for knowledge exploitation. It allows the flexible integration of multiple strategies for augmented representation and score modification and it encapsulates multiple methods that have been previously proposed in the literature. In addition, this approach also allows to exploit deeper knowledge connections between different entities.

Descriptive approaches have proven to be able to easily model such framework, as well as relational classification methods and its integration with content based classifiers within a unique system. Finally, we have shown how to combine document retrieval models to text classification with minimum changes as a test-case for flexible task customisation.

The experiments confirm the applicability of the descriptive modelling of hybrid methods exploiting both relational and content-based data, and they suggest that such models can improve the quality of any of the other approaches. Further experiments with more complex models should be address in future research. In addition, providing more textual data (i.e., including the abstracts and/or adding more training examples) might further improve the combination of relational and content-based classifiers.

This chapter tries to move traditional Information Retrieval towards an “engineering discipline” to work on improving the ability of teams to prototype and then efficiently deliver systems, and respond to customer requests.

Chapter 6

New Approaches of Knowledge Exploitation for the Classification Process

---

Knowledge has been applied several times to improve text classification, mainly to augment the document representation or to modify the classification scores. We argued that the potential of methods exploiting knowledge is beyond these two traditional goals. For this reason, we illustrate how knowledge and, more generally, different evidence generated by the classification process itself, can benefit different classification steps. This includes the introduction of two novel tasks: Semi-Automatic Text Classification (SATC) and Document Performance Prediction (DPP). The former focuses on rank documents and combine manual and automatic classification, maximising the quality, while minimising the cost. DPP addresses the challenge of predict the quality of a given document in a classification problem and how to exploit such information to increase the classification performance. Moreover, a new evaluation method that takes into account near misses using class dependencies, and an adaptive variation of *k*-NN are also presented. These approaches illustrate how knowledge can enhance not only the data representation, nor the scoring of the classification algorithms, but the classification process as a whole.

6.1 Semi-Automatic Text Classification

6.1.1 Preliminaries

Automatic text classification provide much faster and cheaper classification than human experts. However, even though there have been large improvements in the last decades, human experts achieve higher quality. Since the introduction of automatic classifiers, two alternative options can be applied. A full-automatic classification system, where every document is classified according to the decisions made by the classifier; or a completely manual classification if human experts classify all documents. The main drawbacks of the manual approach are its huge cost and potential infeasibility for large collections. Nonetheless, the quality achieved will be higher. A full automatic approach is preferred if large datasets are used (i.e., webpage classification) or when lower quality is not as important as the required human effort. In general, the time saved by using an automatic system is leveraged with the possible quality loss with respect to the manual classification.

This section focuses on an intermediate solution that combines the best of both approaches using Semi-Automatic Text Classification (SATC). This idea was independently, and almost simultaneously, introduced by ourselves [Martinez-Alvarez et al., 2012] and [Berardi et al., 2012] [1]. The main goal of SATC is to achieve high quality with minimum human effort or, more specifically, to use human experts only for the documents that the automatic system is more likely to misclassify. Therefore, maximising the quality, while minimising the cost. Given a set of documents to be classified, and a specific classifier, a SATC algorithm ranks the documents according to their likelihood of being correctly classified. The ranking allows experts to inspect documents iteratively, starting with the most uncertain ones, until a specific point, where the rest of the documents are automatically assigned. As a result, the quality levels can be kept high, while the human effort, and thus the cost, is minimised.

The **Document Difficulty Framework (DDF)**, a family of document certainty algorithms to address SATC, is also introduced. DDF exploits the document-category confidence scores computed by a classifier and the category thresholds given by a class-based thresholding strategy to calculate the certainty of each document. This implies that the category scores for all documents have to be computed. The framework defines an array of different metrics, depending on

[1] [Berardi et al., 2012] refers to it as Semi-Automated Text Classification, instead of Semi Automatic Text Classification

three different dimensions: how the document-class **evidence** is computed, which **classes** will be considered, and how to **aggregate** the document-based certainty.

6.1.2 Task Definition

SATC assumes that neither manual, nor automatic classification is the optimum solution. This situation appears when full automatic classification achieves lower than required quality, and manual classification is either too expensive or unfeasible due to lack of resources (e.g., millions of documents to be classified in 24 hours). The foundation of SATC is that if we are able to separate the documents with high probability to be correctly classified, and the ones that are probably wrong, the latter can be inspected by human experts while the former will be automatically classified. As a result, the resources (i.e., the human experts) are optimised, while the quality remains high. To address this task, SATC ranks the documents to be classified according to their uncertainty. SATC assumes that the documents with higher certainty are probably better classified, whereas the documents with higher uncertainty are more likely to be incorrectly classified. Therefore, the quality is maximised if the human annotators inspect the documents starting from the ones with higher uncertainty.

The possibility of combining human and automatic classification has been suggested before [Larkey and Croft, 1996] [Yang and Liu, 1999] [Sebastiani, 2002]. However, to the best of our knowledge, only one approach has been proposed in the literature: Utility-Theoretic Ranking [Berardi et al., 2012] optimises the global quality of the system, exploiting the potential benefit of manually inspecting each document, using the confidence scores of a classifier, and the quality gain that can be achieved if that label is actually correct. The main conceptual difference with our approach is that the utility-theoretic ranking exploits the collection information, trying to directly optimise the global quality, whereas DDF focuses on each document independently. Furthermore, DDF exploits threshold information, and class filtering for the aggregated document certainty.

Similar to SATC, active learning (explained in detail in Section 2.2) ranks documents according to their benefit in the learning process, selecting which unlabelled documents should be manually labelled and included as training examples. SATC focuses on the classification step, while active learning operates in the training phase, selecting the documents from which the classifier can learn the most.

6.1.3 Evaluation

Semi automatic text classification is evaluated using traditional classification quality measures once the human and the automatic decisions have been combined. This approach provides quality values for different proportions of the collection being automatically classified, where the most uncertain documents are manually classified [Berardi et al., 2012]. It is also important to analyse how quality varies depending on the ratio of documents being manually inspected. For this reason, quality variations with respect to the full automatic quality with the same classifier are also computed. The main challenge is that the relative quality increase depends on the base quality, when all the documents are automatically classified. For instance, in some cases, a 100% quality increase is impossible (i.e., full automatic classification achieving 95% quality), while more than 100% is possible for others, making comparisons over different classifiers unfeasible. To address this challenge, two alternatives based on the error reduction with respect to the full automatic system (instead of its quality increase), were introduced [Berardi et al., 2012]: Error Reduction at rank (ER) and Normalised Error Reduction at rank (NER).

ER measures the error reduction with a specific number of documents being automatically classified, where $E_p(n)$ models the error (defined as "1-quality") achieved by a classifier p with n documents being manually classified. Therefore, $E_p(0)$ represents the error for a fully automatic system. The definition of ER is as follows:

$$\mathrm{ER}_p(n) = \frac{\mathrm{E}_p(0) - \mathrm{E}_P(n)}{\mathrm{E}_p(0)} \tag{6.1}$$

NER subtracts the error reduction at rank n achieved by a random ranker ($\frac{n}{|Te|}$, where $|Te|$ is the size of the documents to be classified) from ER in order to obtain more meaningful quality values:

$$\mathrm{NER}_p(n) = \mathrm{ER}_p(n) - \frac{n}{|Te|} \tag{6.2}$$

A third metric includes the specific position of each document into the evaluation: Expected Normalised Error Reduction (ENER) exploits the probability of a human expert inspecting n documents ($P_s(n)$), where $P_s(n)$ can follow different probability distributions.

$$\mathrm{ENER}_p = \sum_{n=1}^{|Te|} P_s(n) \cdot \mathrm{NER}_p(n) \tag{6.3}$$

One possibly is to model $P_s(n)$ based on the general probability of inspecting one more document. This technique is shown below, where p models the probability of the next document to be inspected [Berardi et al., 2012]:

$$P_s(n) = \begin{cases} p^{n-1} \cdot (1-p) & \text{if } n \in \{1, ..., |Te-1|\} \\ p^{n-1} & \text{if } n = |Te| \end{cases} \tag{6.4}$$

The value of p can be defined as a function of the expected ratio (ξ) of documents being manually classified as follows:

$$p = \frac{1}{\xi \cdot |Te|} \tag{6.5}$$

Therefore, p is computed for different expected ratios of manually classified documents.

6.1.4 Document Difficulty Framework

The Document Difficulty Framework (DDF) is a family of document certainty metrics within the context of classification. DDF exploits the classification scores and the threshold values within a classifier-independent framework to compute the certainty of a document. This computation is divided into three different levels, inspired by the comparison of multi-label Active Learning metrics [Esuli and Sebastiani, 2009]: evidence, class and aggregation: The **evidence** level computes the confidence value for each document and category, using their classification score and the class threshold. The **class** level specifies which classes are to be considered in the final aggregation. The **aggregation** level combines the filtered confidence levels, producing a document-based certainty. Following these principles, DDF is defined as the composition of three transformation functions as shown below, one for each level, where ε represents the evidence, γ the class, and α the aggregation level:

$$\mathrm{certainty}(d) = \alpha\left(\{\gamma(\varepsilon(d, \cdot))\}\right) \tag{6.6}$$

Table 6.1: DDF Levels. c_i represents a class, d a document, and s the classifier's score for each document-class pair. $t(c_i)$ is the threshold for c_i, and $q(c_i)$ is the estimated quality for c_i. Finally, $\Gamma(\gamma,\varepsilon,d)$ represents the set of $\gamma(\varepsilon,d,c_i)$ values for each $c_i \in C$.

Evidence ε; given $d, c_i, t(c_i)$		**Class** γ; given $d, c_i, t(c_i), \varepsilon$		**Aggregation** α; given $\gamma(\varepsilon, d, \cdot)$	
S	$s(d,c_i)$	**A**	$\varepsilon(d,c_i)$	**M**	$\max\limits_{c_i\in C}(\gamma(\varepsilon,d,c_i))$
A	$\ln(1+\lvert s(d,c_i)-t(c_i)\rvert)$	**P**	$\begin{cases} \varepsilon(d,c_i) & \text{if } s(d,c_i)\geq t(c_i) \\ 0 & \text{otherwise} \end{cases}$	**A**	$\operatorname*{avg}\limits_{c_i\in C}(\gamma(\varepsilon,d,c_i))$
R	$\ln(1+\frac{\lvert s(d,c_i)-t(c_i)\rvert}{t(c_i)})$			**W**	$\sum\limits_{c_i\in C}(q(c_i)\cdot\gamma(\varepsilon,d,c_i))$

Table 6.1 summarises the different candidates analysed herein for each DDF levels. Each method is represented as the concatenation of three letters, representing the strategy followed in each level. For instance, `APA` computes the evidence by **A**bsolute difference, only for the **P**ositive classes **A**veraging over the classes:

$$\text{certainty}_{APA}(d) = \operatorname*{avg}_{c_i\in C: s(d,c_i)\geq t(c_i)} \ln(1+\lvert s(d,c_i)-t(c_i)\rvert) \tag{6.7}$$

The three levels considered in DDF and their respective candidates are explained below.

6.1.4.1 Evidence Level

The evidence level computes a confidence value for a document-topic pair. Its domain is defined as follows, where D denotes the set of documents, and C the classes:

$$\varepsilon \in \mathcal{E} : D \times C \to [0,\infty] \tag{6.8}$$

Three candidates are considered for this level: **score** (*S*), **absolute** difference (*A*) and **relative** difference (*R*). The first strategy (*S*) exploits the score obtained by the classifier, assuming that the higher the value, the more relevant the label is. Therefore, classes with higher scores are the ones with more certainty. This is a similar to relevance sampling [Lewis and Gale, 1994], The second method (*A*) exploits the score and the threshold. It assumes that larger distances imply lower uncertainty and higher chance that the document is correctly classified. A logarithmic function is applied to smooth the effect of large differences. The last method (*R*) applies the same principles as the absolute difference approach. However, it uses a relative difference, instead of the absolute value. The rationale is that the absolute distances can be misleading. For instance, a distance of 0.2 would be much more important if the threshold is 0.05 than if it is 0.6.

6.1.4.2 Class Level

The class level behaves as a filter, selecting whether to exploit the certainty of a specific label, and hence, if it will be available at the next aggregation step or not. It is defined as follows, where a composition with an element $\varepsilon \in \mathcal{E}$ would be applied:

$$\gamma \in \Gamma : \mathcal{E} \times D \times C \rightarrow [0, \infty] \tag{6.9}$$

This definition would require to follow the notation $\gamma(\varepsilon(d,c),d,c)$, which is quite cumbersome. Thus, when not ambiguous, we would simply denote it as $\gamma(\varepsilon(d,c))$ or just γ. Two candidates are considered for this level: **all** (A) and **positive** (P). The first strategy (A) considers all the confidence scores for a specific document. Therefore, no filtering is applied. The second method (P) selects the categories for which the classification score is higher or equal than the threshold. These are the classes that will be assigned to the document if automatic classification is applied. This strategy focuses on the positive labels, assuming that they are more representative that the negative ones due to the fact that, at least in text classification, the number of positive classes for a specific document is usually much smaller than the number of negative ones. For example, the average number of classes per document in `Reuters-21578` is 1.24, while the number of classes is 90 (See Section 2.8). This approach assumes that if all classes are observed, the document certainties are somehow diluted because most of the documents will obtain a high confidence that do not belong to a large subset of the classes.

6.1.4.3 Aggregation Level

When multi-label data is used, a certainty value per document, instead of per label, has to be provided, since the ranking of the labels cannot be used to select nor to rank documents. For example, even if 90% of the most certain labels are selected, it is impossible to decide which documents should be automatically classified (i.e., a document can have a very high certainty with respect to one category, while being very uncertain about all the rest). For this reason, the filtered evidence per class should be combined into a single certainty metric for each document. This level is defined as follows:

$$\alpha \in \mathcal{A} : \Gamma \times D \to [0, \infty] \tag{6.10}$$

Typically, it will be applied to the set of possible functions $\gamma \in \Gamma$, one for each class $c_i \in C$. This is equivalent, taken a document d and an evidence level function ε as inputs, to the set $\Gamma(\gamma, \varepsilon, d) = \{\gamma(\varepsilon, d, c_i) : c_i \in C\}$. It should be noted that a one-to-one relation exists between the set of classes used in the definition of Γ and γ itself, and thus, this notation could be simplified as in Table 6.1. Instead of applying the function to every element of Γ, we could simply apply the function γ to each class in C.

Three candidates are considered for this level: **maximum** (M), **average** (A) and **weighted** (W). The first method (M) selects the most certain class for each document. The goal is to rank higher documents with at least one class correctly classified. This strategy is defined for collections with a low number of classes per document. The second method (A) averages the confidence values for the filtered classes, providing a general estimation of how certain the class labels are. The third method (W) uses an averaged weighted linear combination (WLC), based on the estimated quality per class, where categories with low expected quality are weighted less. The main reason is that the implication between high certainty and high quality depends on the performance of the classifier. In other words, even if the certainty for a document-category is very high, the potential quality of such assignment depends on the quality of the classification itself.

6.1.5 Evaluation

Document Difficulty Framework (DDF) represents a family of methods to compute the certainty of a document from a text classification perspective (see Section 6.1.4). To achieve this goal, this approach relies on the classification scores and category thresholds. This section evaluates DDF for the task of Semi-Automatic Text Classification (SATC) , where neither manual, nor automatic classification is the optimum solution (explained in Section 6.1). SATC ranks the documents to be classified according to their uncertainty, and it assumes that the documents with higher certainty are probably better classified, whereas the documents with higher uncertainty are more likely to be incorrectly classified. Based on this ranking, the overall quality is maximised with respect to the available resources if the human annotators inspect a subset of the documents, starting from the ones with higher uncertainty.

6.1.5.1 Set-Up

`Reuters-21578`, `Reuters-21578-115` and `20-newsgroups` are used as datasets. Three different families of classifiers have been used, namely Naive Bayes (Weka [Hall et al., 2009]), k-NN (our own implementation) and SVM using LibSVM [Chang and Lin, 2011]. All documents have been weighted using `ltc` and feature selection based on χ^2 has been applied. Different parameters have been tested for each model and the configuration that achieves better micro-averaged F_1 quality has been selected. The number of features considered is 3000 for k-NN and NB and 10,000 for SVM; and the number of neighbours for k-NN is 60. In all cases the SCutFBR$_{.1}$ thresholding strategy has been used, applying a 5-fold cross-validation process. For SVM, the scored output is obtained by running LibSVM [Chang and Lin, 2011]. The DDF candidates are evaluated using the quality metrics ER and ENER (see Section 6.1.3).

6.1.5.2 Results

The SATC results compare the global quality of a system with respect to the amount of human intervention. Figure 6.1 shows the micro-averaged F_1 and ER evaluation for the best DDF metrics for each collection, depending on the percentage of manually classified documents (Similar graphs for other DDF variations are shown in Appendix A). It illustrates how the best DDF metrics achieve high quality levels, while manually assigning a small subset of the collections. For instance, for `Reuters-21578`, micro-average F$_1$ of more than 95% can be achieved with as few as 20% of the documents manually classified. Furthermore, it also shows that perfect quality is achieved with approximately 50% and 60% of documents manually classified for `Reuters-21578` and `Reuters-21578-115` respectively. `20-newsgroups` appears to be a more challenging collection for SATC, or at least for our approach, where perfect quality is only achieved after 80% of documents are inspected by experts. The reasons can be the uniformed distribution of documents over classes and the high similarity between some of the classes. The best performing model for full automatic classification is SVM. k-NN achieves almost the same quality as SVM for `Reuters-21578` and `Reuters-21578-115`, while being the worst classifier for `20-newsgroups`. The ER graphs suggest that SVM is also the best candidate using DDF for SATC. However, the ER curves are almost completely overlapped for all the different classifiers, especially using `Reuters-21578-115`. This result strongly supports the generalisation of DDF metrics, and it opens the possibility to predict the absolute quality achieved by a new classifier

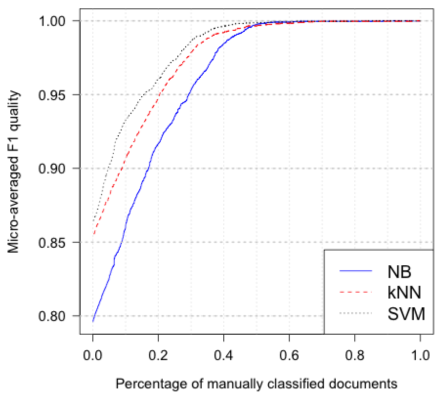
Table 6.2: `20-newsgroups` ENER with respect to the expected ratio of manually classified docs (ξ). Best results per model in bold, best overall is underlined. Improvement (%) with respect to RPA between brackets.

	NB			**kNN**			**SVM**		
ξ	0.05	0.1	0.2	0.05	0.1	0.2	0.05	0.1	0.2
RPA	.097	.164	.230	.073	.127	.185	.121	.194	.251
SAA	-.014 (-114)	-.021 (-113)	-.030 (-113)	.038 (-48)	.058 (-55)	.074 (-60)	.044 (-64)	.046 (-77)	.037 (-85)
SAM	.090 (-6)	.151 (-8)	.211 (-9)	.070 (-5)	.116 (-9)	.161 (-13)	.119 (-1)	.195 (1)	.265 (6)
SAW	.012 (-88)	.030 (-82)	.061 (-74)	.041 (-45)	.062 (-51)	.081 (-56)	.012 (-90)	.033 (-83)	.069 (-72)
SPA	.097 (0)	.164 (0)	.228 (-1)	.073 (-1)	.123 (-3)	.175 (-6)	.120 (-0)	.197 (2)	.267 (6)
SPM	.096 (-0)	.160 (-3)	.218 (-5)	.072 (-2)	.119 (-6)	.165 (-11)	.120 (-1)	.196 (1)	.266 (6)
SPW	.097 (1)	.166 (1)	.231 (0)	.073 (-1)	.125 (-2)	.180 (-3)	**.121** (1)	**.198** (2)	**.268** (7)
DAA	.094 (-3)	.158 (-4)	.219 (-5)	.069 (-6)	.116 (-9)	.170 (-8)	.119 (-2)	.194 (0)	.260 (4)
DAM	.050 (-48)	.077 (-53)	.112 (-51)	.048 (-34)	.081 (-37)	.119 (-35)	.079 (-35)	.136 (-30)	.200 (-20)
DAW	.091 (-6)	.155 (-6)	.216 (-6)	.064 (-12)	.107 (-16)	.156 (-16)	.117 (-3)	.193 (-1)	.259 (3)
DPA	.097 (1)	.166 (1)	.234 (2)	.073 (0)	.127 (0)	.186 (0)	.121 (0)	.197 (2)	.262 (4)
DPM	.096 (-0)	.163 (-1)	.226 (-2)	.073 (-1)	.124 (-3)	.174 (-6)	.121 (-0)	.196 (1)	.261 (4)
DPW	**.097** (1)	**.167** (2)	**.235** (2)	**.073** (0)	**.128** (1)	**.188** (1)	.121 (0)	.198 (2)	.265 (6)
RAA	.089 (-8)	.153 (-7)	.215 (-6)	.068 (-8)	.116 (-9)	.171 (-7)	.119 (-1)	.195 (1)	.260 (4)
RAM	.066 (-31)	.113 (-31)	.172 (-25)	.053 (-28)	.095 (-25)	.145 (-22)	.092 (-24)	.156 (-20)	.216 (-14)
RAW	.086 (-11)	.150 (-8)	.213 (-7)	.064 (-13)	.107 (-16)	.159 (-14)	.118 (-3)	.193 (-0)	.260 (3)
RPM	.096 (-1)	.160 (-3)	.221 (-4)	.073 (-0)	.124 (-2)	.176 (-5)	.120 (-0)	.193 (-0)	.250 (-0)
RPW	.097 (0)	.165 (1)	.232 (1)	.073 (0)	.128 (0)	.187 (1)	.121 (0)	.195 (1)	.256 (2)

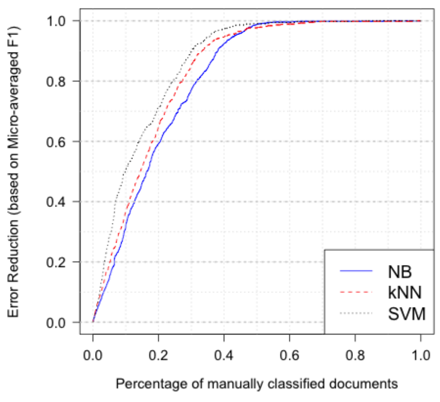
for a given ratio of manually classified documents.

Tables 6.2-6.4 show the ENER quality evaluation for all the DDF candidates, with different percentages of the documents expected to be manually classified (ξ). The results reported using the Utility-Theoretic method (UT) [Berardi et al., 2012] are used as baseline for `Reuters-21578-115`. For the other collections, to the best of our knowledge, no results have been reported with any model in the literature. For this reason, one of our candidates (RPA), is used as the baseline. RPA has been chosen because it is the first method that has ever been applied to address the SATC task [Martinez-Alvarez et al., 2012]. In all cases, the performance of DDF is higher when SVM is used, instead of NB or k-NN. For `Reuters-21578-115`, one DDF metric (SVM with a RAW configuration) outperforms both baselines, as well as any other DDF candidates. The improvements are as high as 14% and 50% with respect to UT and RPA, respectively. In addition, DPA and DPW also outperform UT when $\xi = 0.2$.

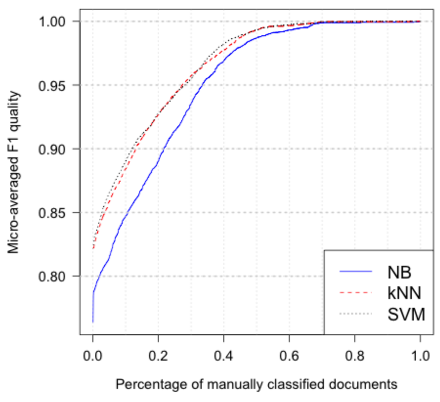
Tables 6.5-6.7 summarise the average quality for different candidates that share similar characteristics, depending on the percentage of the collection being inspected by human annotators (ξ). For instance, S** averages the error reduction for every variation using positive labels. It encapsulates information about SAA, SAM, SAW, SPA, SPM, and SPW. This analysis provides information about which strategies are better for each level in different conditions, and it helps to understand some the results from a general perspective.



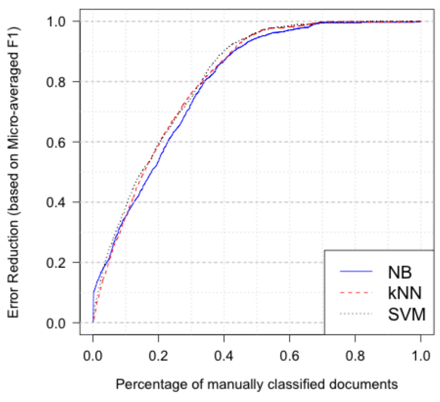
(a) `Reuters-21578` SPW



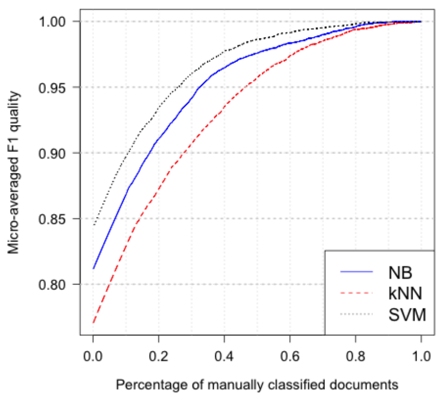
(b) `Reuters-21578` SPW



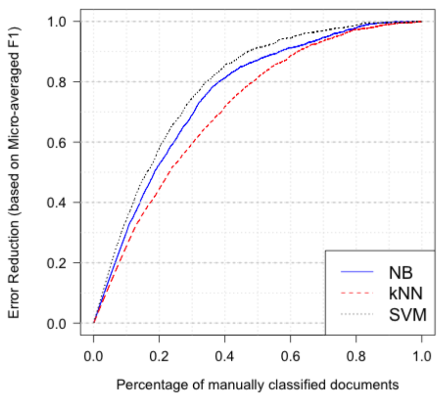
(c) `Reuters-21578-115` RAW



(d) `Reuters-21578-115` RAW



(e) `20-newsgroups` SPW



(f) `20-newsgroups` SPW

Figure 6.1: Micro-averaged F_1 (left) and ER (right) evaluation for different ratios of manually classified documents using the best Document Difficulty Framework candidate for `Reuters-21578`, `Reuters-21578-115`, and `20-newsgroups`.

Table 6.3: `Reuters-21578` ENER with respect to the expected ratio of manually classified docs (ξ). Best results per model in bold, best overall is underlined. Increment (%) with respect to RPA between brackets.

	NB			**kNN**			**SVM**		
ξ	0.05	0.1	0.2	0.05	0.1	0.2	0.05	0.1	0.2
RPA	.101	.178	.245	.139	.233	.317	.156	.250	.321
SAA	.027 (-73)	.032 (-82)	.026 (-89)	.091 (-35)	.155 (-33)	.226 (-29)	.050 (-68)	.052 (-79)	.042 (-87)
SAM	.089 (-12)	.162 (-9)	.244 (-0)	.102 (-27)	.176 (-25)	.251 (-21)	.171 (10)	.269 (8)	.349 (9)
SAW	**.140** (38)	**.210** (19)	.283 (16)	.095 (-32)	.162 (-30)	.235 (-26)	.157 (0)	.246 (-1)	.321 (0)
SPA	.103 (2)	.184 (4)	.267 (9)	.135 (-3)	.221 (-5)	.304 (-4)	.193 (24)	**.297** (19)	.376 (17)
SPM	.094 (-8)	.166 (-6)	.248 (1)	.120 (-14)	.189 (-19)	.260 (-18)	.171 (9)	.270 (8)	.351 (9)
SPW	.104 (2)	.185 (4)	.269 (10)	.137 (-2)	.225 (-3)	.309 (-2)	.192 (23)	.297 (19)	**.376** (17)
DAA	.105 (4)	.183 (3)	.265 (8)	.163 (17)	.234 (1)	.300 (-5)	.175 (12)	.262 (5)	.337 (5)
DAM	.055 (-45)	.093 (-48)	.130 (-47)	.064 (-54)	.122 (-48)	.196 (-38)	.041 (-74)	.074 (-71)	.135 (-58)
DAW	.081 (-20)	.157 (-11)	.242 (-1)	**.167** (20)	.240 (3)	.308 (-3)	.157 (0)	.248 (-1)	.326 (2)
DPA	.105 (4)	.192 (8)	.277 (13)	.143 (2)	.238 (2)	.323 (2)	.187 (19)	.293 (17)	.372 (16)
DPM	.096 (-5)	.174 (-2)	.257 (5)	.126 (-10)	.197 (-16)	.266 (-16)	.164 (5)	.262 (5)	.342 (7)
DPW	.105 (4)	.193 (8)	.280 (14)	.144 (3)	.241 (4)	**.329** (4)	.187 (20)	.294 (18)	.374 (16)
RAA	.115 (14)	.189 (6)	.267 (9)	.138 (-1)	.229 (-2)	.310 (-2)	.171 (9)	.260 (4)	.334 (4)
RAM	.082 (-20)	.135 (-24)	.178 (-27)	.064 (-54)	.131 (-44)	.209 (-34)	.119 (-24)	.192 (-23)	.250 (-22)
RAW	.126 (24)	.205 (16)	**.285** (17)	.158 (13)	**.245** (5)	.323 (2)	**.211** (35)	.295 (18)	.361 (13)
RPM	.093 (-8)	.150 (-16)	.193 (-21)	.126 (-10)	.195 (-16)	.258 (-18)	.136 (-13)	.203 (-19)	.254 (-21)
RPW	.106 (5)	.190 (7)	.267 (9)	.142 (2)	.239 (3)	.325 (3)	.170 (9)	.270 (8)	.344 (7)

Table 6.4: `Reuters-21578-115` ENER with respect to the expected ratio of manually classified docs (ξ). Best results per model in bold, best overall is underlined. Improvement (%) with respect to Utility-Theoretic (UT) Ranking between brackets.

	NB			**kNN**			**SVM**		
ξ	0.05	0.1	0.2	0.05	0.1	0.2	0.05	0.1	0.2
UT	.145	.221	.285	.145	.221	.285	.145	.221	.285
SAA	.015 (-90)	.019 (-91)	.014 (-95)	.060 (-58)	.109 (-51)	.172 (-40)	.032 (-78)	.035 (-84)	.032 (-89)
SAM	.111 (-23)	.165 (-25)	.230 (-19)	.058 (-60)	.113 (-49)	.186 (-35)	.127 (-12)	.206 (-7)	.282 (-1)
SAW	.133 (-9)	.187 (-16)	.248 (-13)	.065 (-55)	.116 (-48)	.180 (-37)	.127 (-12)	.200 (-10)	.267 (-6)
SPA	.068 (-53)	.142 (-36)	.224 (-21)	.063 (-56)	.132 (-40)	.216 (-24)	.103 (-29)	.196 (-11)	.283 (-1)
SPM	.054 (-63)	.119 (-46)	.200 (-30)	.056 (-61)	.112 (-49)	.185 (-35)	.092 (-37)	.179 (-19)	.265 (-7)
SPW	.067 (-53)	.142 (-36)	.224 (-21)	.064 (-56)	.134 (-39)	.220 (-23)	.102 (-30)	.196 (-11)	.283 (-1)
DAA	.097 (-33)	.153 (-31)	.221 (-23)	.139 (-4)	.199 (-10)	.260 (-9)	.114 (-21)	.194 (-12)	.274 (-4)
DAM	.032 (-78)	.047 (-79)	.063 (-78)	.070 (-52)	.121 (-45)	.184 (-35)	.055 (-62)	.100 (-55)	.170 (-40)
DAW	.039 (-73)	.101 (-54)	.181 (-36)	**.141** (-3)	.202 (-8)	.265 (-7)	.089 (-39)	.173 (-22)	.260 (-9)
DPA	.069 (-52)	.147 (-33)	.230 (-19)	.067 (-54)	.144 (-35)	.234 (-18)	.101 (-30)	.200 (-9)	.292 (2)
DPM	.055 (-62)	.121 (-45)	.201 (-29)	.059 (-59)	.120 (-46)	.195 (-31)	.091 (-37)	.182 (-18)	.271 (-5)
DPW	.069 (-52)	.148 (-33)	.232 (-19)	.067 (-54)	.146 (-34)	.237 (-17)	.102 (-30)	.201 (-9)	.293 (3)
RAA	.136 (-6)	.186 (-16)	.243 (-15)	.121 (-16)	.199 (-10)	.273 (-4)	.129 (-11)	.201 (-9)	.269 (-5)
RAM	.077 (-47)	.113 (-49)	.143 (-50)	.062 (-57)	.125 (-43)	.199 (-30)	.104 (-28)	.170 (-23)	.235 (-18)
RAW	**.147** (2)	**.204** (-8)	**.264** (-7)	.133 (-8)	**.209** (-6)	**.280** (-2)	**.167** (15)	**.240** (9)	**.306** (7)
RPA	.055 (-62)	.124 (-44)	.195 (-32)	.065 (-55)	.141 (-36)	.231 (-19)	.087 (-40)	.178 (-19)	.266 (-7)
RPM	.041 (-72)	.081 (-64)	.118 (-58)	.060 (-59)	.120 (-46)	.194 (-32)	.073 (-49)	.148 (-33)	.221 (-22)
RPW	.067 (-54)	.143 (-36)	.220 (-23)	.066 (-54)	.144 (-35)	.236 (-17)	.097 (-33)	.193 (-13)	.282 (-1)

6.1.5.3 Analysis

For `20-newsgroups`, there is almost no difference between the performance of candidates applying average aggregation and those applying weighted aggregation (e.g., DPA vs DPW). The main reason for this is that the quality achieved for different classes is very similar. Furthermore, although no one of the best candidates includes the aggregation based on the maximum confidence, this strategy achieves high quality (i.e. SPM is virtually as good as the best candi-

Table 6.5: Average ENER evaluation for DDF patterns and $\xi = 0.05$

Collection	Model	S**	D**	R**	*A*	*P*	**A	**M	**W
20newsgroups	NB	.063	.088	**.088**	.063	**.097**	.077	**.082**	.080
	kNN	.061	.067	**.067**	.057	**.073**	**.066**	.065	.065
	SVM	.090	.113	**.115**	.091	**.121**	.107	**.108**	.102
Reuters21578_115	NB	.075	.060	**.087**	**.087**	.061	.073	.062	**.087**
	kNN	.061	**.090**	.085	**.094**	.063	.086	.061	**.089**
	SVM	.097	.092	**.109**	**.105**	.094	.094	.090	**.114**
Reuters21578	NB	.093	.091	**.104**	.091	**.101**	.093	.085	**.110**
	kNN	.113	**.134**	.128	.116	**.135**	.135	.100	**.140**
	SVM	.156	.152	**.160**	.139	**.173**	.155	.134	**.179**

Table 6.6: Average ENER evaluation for DDF patterns and $\xi = 0.1$

Collection	Model	S**	D**	R**	*A*	*P*	**A	**M	**W
20newsgroups	NB	.108	.148	**.151**	.107	**.164**	.131	.137	**.139**
	kNN	.100	.114	**.116**	.095	**.125**	**.111**	.110	.109
	SVM	.144	.186	**.188**	.149	**.196**	.170	**.179**	.168
Reuters21578_115	NB	.129	.119	**.142**	**.131**	.130	.129	.108	**.154**
	kNN	.119	.155	**.156**	**.155**	.132	.154	.118	**.158**
	SVM	.169	.175	**.188**	.169	**.186**	.167	.164	**.201**
Reuters21578	NB	.157	.165	**.174**	.152	**.179**	.160	.147	**.190**
	kNN	.188	.212	**.212**	.188	**.220**	.218	.168	**.225**
	SVM	.238	.239	**.245**	.211	**.271**	.236	.211	**.275**

Table 6.7: Average ENER evaluation for DDF patterns and $\xi = 0.2$

Collection	Model	S**	D**	R**	*A*	*P*	**A	**M	**W
20newsgroups	NB	.153	.207	**.214**	.154	**.228**	.183	.193	**.198**
	kNN	.139	.165	**.170**	.137	**.179**	**.160**	.157	.158
	SVM	.195	**.251**	.249	.203	**.261**	.223	**.243**	.230
Reuters21578_115	NB	.190	.188	**.197**	.179	**.205**	.188	.159	**.228**
	kNN	.193	.229	**.235**	**.222**	.216	.231	.191	**.236**
	SVM	.235	.260	**.263**	.233	**.273**	.236	.241	**.282**
Reuters21578	NB	.223	**.242**	.239	.213	**.256**	.224	.208	**.271**
	kNN	.264	.287	**.290**	.262	**.299**	.297	.240	**.305**
	SVM	.302	**.315**	.311	.273	**.346**	.297	.280	**.350**

date for SVM). On the other hand, maximum aggregations perform poorly with the multi-label collections, as expected. The models that consider positive classes lose their competitiveness against selecting all classes, for `Reuters-21578-115`. The reason is that this strategy was conceived for collections where test documents have at least one correct class, as documents with no classes are assigned a large uncertainty. Another difference is that, while there is a clear winner for `Reuters-21578-115` (RAW), there is none for `Reuters-21578`. Furthermore, the qualities achieved by the best model in `Reuters-21578` are significantly higher than those for `Reuters-21578-115`. This means that the addition of documents without correct classes makes the SATC problem more complex to solve, or at least that DDF metrics are less suited for this type of datasets. Results also show that SAA (and SAW for `20-newsgroups` because of the similar qualities per class) is only suited for classifiers that do not normalise the scores per document. If the classification scores are normalised per document ($\sum_{c_i \in C} s(d, c_i) = 1$), SAA produces the same difficulty, independently of the document. As a result, it performs as a random ranker for this type of classifiers, which include the versions of NB and SVM used in this thesis. Other poor metric is DAM, because the highest confidence based on difference is usually based on a very low (or even zero) score. Therefore, the certainty computation will be uniquely based on this information. Any method following the pattern $\{$A,R$\}$AM is likely to perform very poorly because any class with a zero score can be considered as the label with higher confidence, and therefore, the document confidence will be based on it. For the evidence level, the best strategy is the relative difference. The class level illustrates that the selection of positive classes achieves good quality, as long as the assumption that all the documents have at least one correct class is correct. Otherwise (i.e., like `Reuters-21578-115`), all classes should be considered. The aggregation level shows that the exploitation of category quality estimation outperforms the other strategies for `Reuters-21578` and `Reuters-21578-115`.

6.1.6 Discussion

Semi Automatic Text Classification represents a largely unexplored task within text classification that is critical in environments where high quality classification is needed, but resources are limited. Its main goal is to achieve high quality with minimum human effort. This research has introduced SATC and the document difficulty framework (DDF) to address it. DDF generalises several methods by abstracting three different levels, specifying how to manipulate the scores and thresholds to obtain a document certainty measure. Results show that DDF metrics achieve virtually perfect classification with as low as 50% of documents being classified with some collections. One of the main limitations of SATC is that, even with this high error reduction, the amount of documents to be manually classified would be too large in some cases (e.g., where the collection has millions of documents).

SVM is the best classifier for DDF and RAW is its best overall variation, with the exception of

`Reuters-21578-115`, where NB with the SAW strategy is the best alternative. DDF outperforms all the previously proposed methods in the literature for SATC. The strategy analysis shows that the best models should include a relative difference of scores, and the exploitation of estimated class quality. In addition, observing only the positive classes for a document achieves better average quality, but only if all documents belong to at least one class. Nonetheless, the results from the pattern analysis can be biased towards the extreme cases and be less reliable. For instance, given the fact that SAA performs as a random ranker for Naive Bayes and SVM significantly decreases the average quality of the patterns S**, *A* and **A.

6.2 Document Performance Prediction

6.2.1 Preliminaries

The main goal of performance prediction is to estimate the effectiveness of a specific approach to solve a task. In Information Retrieval, Query Performance Prediction (QPP) estimates the performance of a search engine for a query. QPP allows to apply specific processing techniques to difficult queries, such as query expansion or knowledge augmentation [Amati et al., 2004], and the aggregation of search results [Aslam and Pavlu, 2007]. In classification, performance metrics have been defined for a specific collection or classifier. We argue that a quality estimation at the document level provides a much deeper analysis and understanding of the classification process. It enables classification ensembles in a document level, as well as specialised processing of the most difficult documents. Furthermore, the observation of difficult documents with respect to the models will provide insights about what type of documents each algorithm classifies the best. This section introduces and defines the concept of document performance prediction (DPP) for classification.

Document performance prediction shares the principles of QPP (explained in Section 3.3), but it is focused on documents instead of queries. A comparison of the duality between DPP and QPP is presented, underlying their similarities and differences. Furthermore, this section also introduces the Threshold-based Document Performance (TDP) prediction framework, a family of document performance prediction algorithms that exploits the document-class confidence scores computed by a classifier and the class thresholds given by any class-based thresholding strategy to estimate the performance of each document. Besides, this framework defines an array of different metrics that estimate the quality of documents depending on three dimensions: how the document-class **evidence** is computed, what **normalisation** is applied, and the estimated **quality** for a given label. This approach is conceptually similar to the Document Difficulty Framework that was introduced in Section 6.1.4.

6.2.2 Task Definition

Before explaining the rationale behind document performance prediction, each of the components involved in such process are mathematically described. Table 6.8 illustrates the common notation for all equations presented in this section.

Symbol	**Explanation**
C	the set of classes
c	class
T_r	the set of training documents
T_e	the set of testing documents
T_v	the set of validation documents
σ	classification scoring function
τ	threshold function
f	decision function which determines if a document will be labelled in a class
q	quality function for a collection
q_d	quality function for a document
$\hat{q}_d$	estimated document quality
α	performance predictor
r	correlation function

Table 6.8: Notation Summary for Document Performance Prediction.

Given a set of training documents to learn from, and a test document d, a classifier provides one score per each class c:

$$\sigma_c(T_r, d) \in \mathbb{R} \tag{6.11}$$

Given a set of scores for each document (from a validation set) and class, the thresholding strategy provides the optimum threshold with respect to a quality measure q. This step is usually applied as a cross-validation process:

$$\tau_q(\{\sigma_c(T_r, d) : d \in T_v\}) \in \mathbb{R} \tag{6.12}$$

From now on, the set of training documents is assumed to be the same, therefore, for the sake of readability, it is omitted in the equations. As a result, $\sigma_c(d)$ and $\tau(c)$ represent the score and the threshold for the class c, using document d. Given a score for a document and a class, and the class threshold, the decision function (f) determines if the document belongs to the class or not:

$$f_c(d, \sigma_c(d), \tau(c)) \in \{0,1\} \tag{6.13}$$

Given a set of decisions and a quality metric based on the test set, is computed as follows:

$$q(\{f(d, \sigma_c(d), \tau(c)) : d \in T_e \land c \in C\}) \in \mathbb{R} \tag{6.14}$$

The performance prediction functions produce a score, which depends on the class thresholds, and classification scores for the methods exploiting post-classification information. Methods that do not use it (i.e pre-classification metrics) depend only on the specific document:

$$\alpha(\{\tau(c), \sigma_c(d), d : c \in C\}) \in \mathbb{R} \tag{6.15}$$

For simplicity, $\alpha(d)$ denotes the performance of document d. Finally, a predictor quality function r assesses the predictive quality of the certainty function compared to the real quality value. The standard methodology to measure the effectiveness of performance prediction techniques consists of comparing the rankings of several elements (documents in the case of DPP) based on their performance prediction and quality values. This is measured by correlation functions such as Pearson's, Spearman's, or Kendall's coefficients, implying that $r \in [-1,1]$.

$$r(\{\alpha(d)\}, \{q_d(d)\}) \in \mathbb{R} \tag{6.16}$$

Quality evaluation in text classification is traditionally computed globally, based on a set of documents. However, the correlation metrics require a document-based quality evaluation (q_d). Therefore, such evaluation should be defined similarly to the global quality, but limiting the space to a specific document as shown below:

$$q_d(\{f(\sigma_c(d), \tau(c) : c \in C\}) \in \mathbb{R} \tag{6.17}$$

We propose a document-level F_1 quality evaluation measure that follows the same principles of traditional F_1, but it focuses only on one document:

$$\mathrm{Pr}_D(d) = \frac{TP_D(d)}{TP_D(d) + FP_D(d)} \tag{6.18}$$

$$\mathrm{Re}_D(d) = \frac{TP_D(d)}{TP_D(d) + FN_D(d)} \tag{6.19}$$

$$\mathrm{F}_{1_D}(d) = \frac{2 \cdot \mathrm{Pr}_D(d) \cdot \mathrm{Re}_D(d)}{\mathrm{Pr}_D(d) + \mathrm{Re}_D(d)} \tag{6.20}$$

These document-specific quality metrics can now be compared to any performance predictor for evaluation purposes.

6.2.3 Applications

The capability of predict the quality (or performance) of a document can be exploited in several situations in the context of classification. For instance, it allows a document-level classifier combination, where the best classifier for each document is applied. Other examples are the adaptive document augmentation methods that can use different strategies depending on the expected quality of a document (e.g., apply augmented representation just for the documents with low performance prediction). In addition, the ranking based on performance prediction for a set of documents can be applied to solve the Semi Automatic Text Classification task (see Section 6.1). The main opportunities and challenges for each one of this applications are illustrated in the next sections.

6.2.3.1 Document-Level Classifier Combination

The combination of classifiers is usually applied on a collection level. For example, the results or rankings from different classifiers are aggregated using quality estimations for each classifier [Larkey and Croft, 1996]. This approach has the drawback that it does not consider the situation that one classifier performs badly for the collection, while it achieves high quality for specific documents. A document-based classifier combination should be able to address this problem.

Document performance prediction is a well suited candidate for this task and it has the main benefit that it incorporates a new granularity dimension (i.e., the document context). This is similar to the use of reliability indicators to combine classifier and document-based features in a meta-classifier [Bennett et al., 2005].

One of the challenges of using performance prediction is that the performance values are not sufficient to be applied for combination. Quality estimation should be computed, instead of performance prediction. The reason is that, even though the performance values are suited to rank different documents given the same classifier, they are not applicable over different classifiers: Given two documents (d_i and d_j), and a classifier (ω), higher prediction value implies higher expected quality ($\hat{q}_\omega$) for a document. On the other hand, for a given document d, there is no implication between its prediction values over different classifiers (ω and ω'). These concepts are mathematically formulated as follows:

$$\alpha_\omega(d_i) \geq \alpha_\omega(d_j) \rightarrow \hat{q}_\omega(d_i) \geq \hat{q}_\omega(d_j) \tag{6.21}$$

$$\alpha_\omega(d) \geq \alpha_{\omega'}(d) \nrightarrow \hat{q}_\omega(d) \geq \hat{q}_{\omega'}(d) \tag{6.22}$$

The conceptual difference between predicted values and quality estimations has been pointed out in the context of QPP [Hauff et al., 2009]. QPP algorithms are useful to estimate the query performance among a predefined set of target queries, where the query with the highest score is considered to perform the best in that set. In that context, such algorithms can produce a ranking of queries, but do not directly estimate their quality (i.e., the Average Precision in information retrieval or F_1 in text classification), which is required for the combination of classifiers and, as mentioned in [Hauff et al., 2009], to compare the performance of queries among different collections. The authors propose to apply linear regression to obtain normalised versions of the predicted scores.

Regression is not suited for document performance prediction because the quality for any given document is limited to a finite (and small) set of values. The reason for this limited range is that the F_1 quality (among other measures) depends on the number of true positives (TP), false positives (FP) and false negatives (FN). Their range, for a given document, is based on the total number of classes and the number of labelled documents, as shown below, where $R(d)$ is the set of correctly labelled classes for d and C is the set of classes in the collection.

$$\begin{aligned} TP &\in \{0...|R(d)|\} \\ FP &\in \{0...|C|-|R(d)|\} \\ FN &\in \{0...|R(d)|\} \end{aligned} \tag{6.23}$$

For any given document, $FN = |R(d)| - TP$. Therefore, the maximum number of quality values is defined as $|R(d)| \cdot (|C| - |R(d)|)$. In addition, a maximum boundary for the overall number of quality values for a set of documents can be computed using the maximum number of correct classes for any document, and the maximum number of classes assigned by the system. These ranges are shown below, where A models the assigned categories by the system to a specific document:

$$\begin{aligned} TP &\in \{0\ldots\max(|R(d_i)|)\} \\ FP &\in \{0\ldots\max(|A(d_i))|\} \\ FN &\in \{0\ldots\max(|R(d_i)|)\} \end{aligned} \tag{6.24}$$

This implies that the maximum number of quality values, for any given document is defined as: $\max(|R(d_i)|) \cdot \max(|A(d_j)|)$. For instance, for `Reuters-21578`, the maximum number of correct classes is 14. As a result, if the maximum number of assigned classes by the system is, for example, 10, there are 140 possible quality values to be computed. However, the average number of correct classes per document is 1.24, and most documents are assigned to one or two classes by the system. Therefore, the majority of the documents only have two to four different quality values (always including zero and one).

6.2.3.2 Document Augmentation

Performance prediction allows to apply additional, or more advanced, processing to the most uncertain documents. This technique is referred to as document augmentation or reformulation (similarly to query reformulation in Query Performance Prediction). Alternatively, multiple representations can be computed, selecting the one the highest performance prediction. However, as previously explained for classifier combination, a performance prediction cannot be directly applied, instead, a quality estimation should be obtained. For instance, if the number of tokens is used as a predictor, and two different representations, with 1000 and 5000 features are compared, the latter will always have more tokens, and thus a higher performance value. Given a classifier ω, a quality estimation $\hat{q}$, and a representation r, these concepts can be formalised as follows:

$$\alpha_{\omega,r}(d_i) \geq \alpha_{\omega,r}(d_j) \nrightarrow \hat{q}_{\omega,r}(d_i) \geq \hat{q}_{\omega,r}(d_j) \tag{6.25}$$

If the performance prediction values are successfully mapped to quality estimations, the best representation, for a given classifier ω and document d, can be selected as follows:

$$\arg\max_{r} \hat{q}_{\omega,r}(d) \tag{6.26}$$

Any document augmentation technique can be applied to provide an alternative representation of the document.

6.2.3.3 Semi-Automatic Text Classification (SATC)

Semi-automatic text classification (explained in detail in Section 6.1) assumes that neither manual, nor full automatic classification is the optimum solution. This situation appears when full automatic classification achieves lower than required quality, and a full manual classification is either too expensive or unfeasible due to lack of resources. The foundation of SATC is that if we are able to separate the documents with high probability to be correctly classified, and the ones that are probably wrong, the latter can be inspected by human experts while the former will be automatically classified. To solve this task, SATC methods rank the documents to be classified by their uncertainty.

Document performance prediction seems to be well suited to solve SATC problems, as it can be used to rank documents according to their expected quality. However, SATC ranking is optimum when the documents are sorted according to their capacity to achieve maximum quality improvement. This characteristic implies that, even though a DPP-based and a collection-based ranking are related, the best SATC ranking is not necessary the ranking obtained by an optimum DPP method. The reason for this is that, from a SATC perspective, the number of correct labels is irrelevant, while the number of incorrect ones (either false positive or false negative) are critical for the selection of the best documents to be inspected. For instance, given two documents (d_1 and d_2) and their assigned classes $A(d_1) = \{c_1, c_2, c_3, c_4, c_5\}$, $A(d_2) = \{c_1, c_2\}$. If all classes but one are correctly assigned and there is one class missing for each document, the first case will have one false positive (FP), 4 true positives (TP) and one false negative (FN), whereas d_2 has 1 FP, 1 TP, and 1 FN. The increase in quality achieved if a human inspects either document is the same, as the number of TP should have no impact on this increment. In both cases, a manual inspection will change each false positive into a true positive, and each false negative into a true negative, while keeping the rest of assignments because they were correct. Therefore, both documents should have the same position in the ranking, even though d_1 is "better" from a DPP perspective.

6.2.4 Duality between DPP and QPP

The basis of the prediction for QPP and DPP are essentially different; whereas in QPP the element upon which the predictions are made is a query, in DPP the basis of predictions is a document, for which more and richer information is usually available. In addition to the fact that documents are longer than queries, they can present structural data or other complex information (e.g., authorship, citations, ...). Furthermore, there are also more documents than queries in the collections used for evaluation, and thus, the predictor has to be efficient with respect to a larger number of elements. This aspect also motivates further ways of estimating the predictive power of a document performance predictor when only a subset of the documents are considered (e.g., only the top most difficult). In addition, an important difference of DPP with respect to QPP is that in classification, a document could be assigned to multiple classes, and hence, relevance becomes a multidimensional variable, in contrast to relevance in classical information retrieval, where a document is either relevant or not for a query. Both tasks consider a quality metric based on the comparison of human assessments and the system predictions. In QPP, this metric is usually the average precision. In classification, quality metrics are not usually based on a specific document but a set of them, and thus we have to define document-based evaluation metrics for classification. Another significant difference is that the elements of the prediction (i.e., queries or documents) are issued to different components of the system. In QPP a retrieval engine receives the query and return a list of likely relevant documents for that query, whereas in DPP documents are classified into the more relevant classes. This differences lead to different definitions of performance predictors, along with alternative information sources available in each situation. While query performance predictors may use the output of a retrieval engine to produce predictions, DPP metrics can be based on classification scores and category thresholds as

Table 6.9: Performance prediction concepts in QPP vs DPP.

	QPP	**DPP**
Elements	Query	Document
Quality	Average precision	Document-based F_1
Relevance levels	Binary	Multidimensional
Elements are issued to	Retrieval engine	Document classifier

Table 6.10: Performance prediction applications for QPP vs DPP.

QPP	**DPP**
Query selection	SATC
Query expansion	Document augmentation
Rank aggregation	Ensemble classifier

their main input. We have to emphasise that, despite their differences, summarised in Table 6.9, QPP and DPP are inherently and intrinsically the same. Both aim to estimate the performance of an element (either a query or a document) using a quality metric. Furthermore, they can also use the output of the system where the elements have been issued to for the prediction.

Table 6.10 shows a comparison between them from the task perspective. It underlines the similarity of their application in two different contexts (i.e., queries vs documents). There is an almost perfect parallelism for knowledge expansion. The queries that are expected to perform badly are expanded or reformulated, while some type of knowledge augmentation or a different document representation will be applied for under performing documents. The possibility of using performance predictions for model combinations is also very similar, where rank aggregation and (document-level) ensemble classification share the same philosophy and steps. SATC (see Section 6.2.3.3) is similar to the expansion of the available relevance judgements by properly selecting queries [Hosseini et al., 2011]. Providing feedback to the system administrator was considered from the beginning as a potential advantage of using QPP [Yom-Tov et al., 2005]. In this way, future queries (or documents, in our case) are better answered by taking into account the amount of estimated poorly performing queries.

6.2.5 Threshold-based Document Performance Prediction

The Threshold-based Document Performance (TDP) prediction framework models different DPP metrics, within the context of text classification. The two main foundations of this framework are the uncertainty of any document-class assignment, and the estimation of this assignment to be correct. In other words, the former computes how confident the system is in the labelling of a document to a class, while the latter computes the estimated quality that such assignation would achieve. In addition to these, a third dimension is included, whose main goal is to normalise the label confidence to obtain a value within the range $[0,1]$. This formulation represents the core of this framework. Different estimations for uncertainty labelling and quality will define the different metrics to be applied. Therefore, TDP is divided in three different levels: quality, evidence, and normalisation.

The **quality** level estimates the expected performance of an assignment to the class. The **evidence** level computes the confidence value for each document and category, using their classification score and the class threshold. The **normalisation** level maps the evidence values into a $[0,1]$ range. TDP is based on the composition of three transformation functions, one for each level, where θ measures the quality, ε the evidence, and ω the normalisation level.

Exploiting these three dimensions, the framework estimates the F_1 quality for a document, based on the estimation of TP, FP, TN and FN

$$\hat{\mathrm{TP}}(d) = \sum_{c\in C;s(d,c)\geq\tau(c)} \theta(d,c)\cdot\omega(\varepsilon(d,c)) \tag{6.27}$$

$$\hat{\mathrm{FP}}(d) = \sum_{c\in C;s(d,c)\geq\tau(c)} 1-\theta(d,c)\cdot\omega(\varepsilon(d,c)) \tag{6.28}$$

$$\hat{\mathrm{TN}}(d) = \sum_{c\in C;s(d,c)<\tau(c)} \theta(d,c)\cdot\omega(\varepsilon(d,c)) \tag{6.29}$$

$$\hat{\mathrm{FN}}(d) = \sum_{c\in C;s(d,c)<\tau(c)} 1-\theta(d,c)\cdot\omega(\varepsilon(d,c)) \tag{6.30}$$

$$\alpha(d) = \hat{\mathrm{F}}_1(d) = \frac{2\cdot\hat{\mathrm{Pr}}(d)\cdot\hat{\mathrm{Re}}(d)}{\hat{\mathrm{Pr}}(d)+\hat{\mathrm{Re}}(d)} \tag{6.31}$$

A predictor based on the TDP framework consists in the combination (or composition) of a specific strategy for each level. Table 6.11 summarises the different candidate strategies analysed herein. Each DPP metric is then referred to as the concatenation of three letters, representing the

Table 6.11: TDP Levels. c_i represents a class, d a document, and s the classifier's score for each document-class pair. $t(c_i)$ is the threshold for c_i, and $q(c_i)$ is the estimated quality for c_i.

Quality		**Evidence**		**Normalisation**	
θ; given d, c_i		ε; given $d, c_i, t(c_i)$		ω; given $d, c_i, t(c_i), \varepsilon$	
B	1.0	**B**	1.0	**L**	$\frac{e^{\varepsilon}}{e^{\varepsilon}+1}$
F	$\hat{F}_1(c_i)$	**A**	$\lvert s(d,c_i) - t(c_i)\rvert$	**F**	$\frac{\varepsilon}{\varepsilon+1}$
		R	$\frac{\lvert s(d,c_i)-t(c_i)\rvert}{t(c_i)}$	**S**	$\frac{\varepsilon}{\sqrt{\varepsilon^2+1}}$

strategy followed in each one of the levels.

6.2.5.1 *Quality Level*

Two candidates are considered to compute the quality of a specific assignation: **binary** (*B*), and **F$_1$-based** (*F*). The first method (*B*) does not exploit any quality estimation. Therefore, for this level, the value computed for each label is always one, assuming that the class quality is not relevant. As a result, the evidence and normalisation dimensions will have all the prediction power. The second method (*F*) assumes that the quality of every label for a given class is the same as the estimated F$_1$ quality for the class, obtained via cross-validation.

6.2.5.2 *Evidence Level*

Three candidates are considered for this level: **binary** (*B*), **absolute** difference (*A*) and **relative** difference (*R*). The first method(*B*) assumes that the certainty of all the decisions is the same (1.0), independently of the thresholds and scores. The second method (*A*) exploits the score and the threshold, assuming that larger distances imply lower uncertainty and higher chance that the document is correctly classified. This assumption is similar to uncertainty sampling in the case of active learning [Lewis and Gale, 1994]. The last method (*R*) applies the same principles as the previous approach. However, it uses a relative difference instead of the absolute value. The rationale is that the absolute distances can be misleading. For instance, a distance of 0.2 would be more important if the threshold is 0.05 than if it is 0.6. This strategy is based on the SATC method presented in Section 6.1.4

6.2.5.3 *Normalisation Level*

The normalisation level is responsible for mapping the evidence score to a range between zero and one. Three candidates are considered for this level: **logit** (*L*), **fraction** (*F*) and **squared** fraction (*S*). The first strategy (*L*) applies a logit normalisation. The second method (*F*) applies a fractional normalisation. The third method (S) uses a normalisation based on the squared root of the squared value to be normalised. Other methods can be considered for this strategy, as long as they normalise the evidence within the range [0,1].

6.2.6 Evaluation

Document Performance Prediction (DPP) estimates the quality performance of a classifier on a document basis. Several applications for DPP were illustrated in Section 6.2. However, all of them rely on high levels of correlation being achieved between the performance predictors values and the achieved quality for each document. The main goal of this section is to evaluate such correlation.

Table 6.12: Pre-classification DPP metrics. Weight refers to any feature weighting strategy (e.g., idf, χ^2, `ltc`, ...).

Name	**Definition**
No. words	$\alpha(d) = \|t \in d\|$
Avg. Weight	$\alpha(d) = avg_{t \in d} weight(t)$
Max. Weight	$\alpha(d) = \max_{t \in d} weight(t)$
Std. Weight	$\alpha(d) = std_{t \in d} weight(t)$
Nouns	$\alpha(d) = \|t \in d : \text{is\_noun}(t)\|$
Prepositions	$\alpha(d) = \|t \in d : \text{is\_prep}(t)\|$
Pronouns	$\alpha(d) = \|t \in d : \text{is\_pronoun}(t)\|$

Table 6.12 summarises several pre-retrieval query performance predictors that are directly applied in order to analyse their effectiveness. The linguistic features have been obtained using `nltk` [Bird et al., 2009]. A trivial pre-classification predictor would be the document length, analogous to those predictors defined for query performance prediction based on the query length. Other statistics based on term frequency (TF) and inverse document frequency (IDF) are also computed. More importantly, other metrics, specific to the classification can be exploited. This includes observing the weights for different strategies such as `ltc` or `tfc` [Buckley et al., 1994] and category information (e.g., χ^2). Besides, linguistic features are also computed for each document, such as those defined in [Mothe and Tanguy, 2005], (e.g., number of nouns or pronouns). In addition to the pre-classification metrics, the threshold-based document performance prediction (TDP) metrics, presented in detail in Section 6.2, are also evaluated. TDP exploits the uncertainty of a document-class assignment, and the estimation of this assignment to be correct.

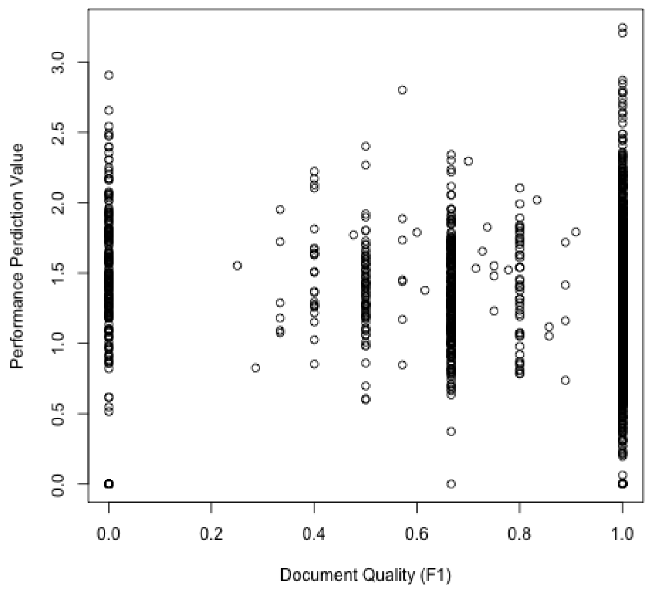


Figure 6.2: Quality vs Prediction using `std.idf` for `Reuters-21578` with a k-NN classifier.

6.2.6.1 Set-Up

The datasets used for this experiment are `Reuters-21578` and `20-newsgroups`. Three different families of classifiers have been used, namely Naive Bayes (Weka [Hall et al., 2009]), k-NN (our own implementation) and SVM using LibSVM [Chang and Lin, 2011]. All documents have been weighted using `ltc` and feature selection based on χ^2 has been applied. Different parameters have been tested for each model and the configuration that achieves better micro-averaged F_1 quality (for traditional text classification) has been selected. The number of features considered is 3000 for k-NN and NB and 10,000 for SVM; and the number of neighbours for k-NN is 60. In all cases the SCutFBR$_{.1}$ thresholding strategy has been used, applying a 5-fold cross-validation process. In the case of SVM the scored output is obtained, instead of the binary decision.

6.2.6.2 Results

Before showing the quality results for the different predictors, Figure 6.2 illustrates the comparison between a predictor based on the standard deviation of IDF (`std.idf`), and the documents quality achieved by a k-NN classifier for `Reuters-21578`. This example empirically supports the claim made in Section 6.2, that a small number of discrete quality values are attainable for each document. In this case, such values are mainly restricted to the set $\{0, 0.4, 0.5, 0.66, 0.8, 1\}$, in contrast with the continuous range of the predictor.

Table 6.13: Pearson correlation: Pre-Classification Predictors.

	20newsgroups			**Reuters21578**		
	NB	**kNN**	**SVM**	**NB**	**kNN**	**SVM**
vocab.size	14.26	15.54	9.42	-25.22	-20.22	-25.13
pronouns	3.99	5.46	3.67	-15.69	-11.19	-15.51
prepos.	5.84	6.79	5.65	-27.48	-21.53	-24.78
nouns	1.41	1.98	1.38	-21.49	-17.61	-21.14
max.χ^2	21.00	21.36	14.08	-34.22	-28.65	-25.01
avg.χ^2	23.59	23.20	19.40	-34.31	-28.55	-27.53
std.χ^2	16.72	16.61	11.34	-33.19	-28.27	-19.68
max.tf	5.55	6.70	5.32	-7.32	-2.96	-8.36
avg.tf	8.66	9.62	7.09	5.82	8.32	6.49
std.tf	7.72	8.49	7.15	-0.03	2.65	-0.71
max.tfc	-10.62	-9.17	-7.26	2.83	7.78	8.42
avg.tfc	-22.07	-19.95	-16.89	17.69	19.38	19.44
std.tfc	-18.32	-16.85	-15.95	6.83	8.74	21.52
max.ltc	-20.06	-18.21	-15.34	6.86	9.65	16.59
avg.ltc	-22.23	-20.05	-17.20	17.90	20.15	19.98
std.ltc	-19.97	-18.31	-16.89	3.60	5.07	20.91
max.idf	**28.12**	**26.21**	15.86	-31.00	-26.73	-21.31
avg.idf	26.56	25.23	**22.16**	**-34.78**	**-29.24**	**-28.20**
std.idf	14.13	12.66	6.23	-24.60	-21.95	-11.43

Table 6.14: Spearman correlation: Pre-Classification Predictors.

	20newsgroups			**Reuters21578**		
	NB	**kNN**	**SVM**	**NB**	**kNN**	**SVM**
vocab.size	20.86	21.30	16.36	-23.42	-24.68	-27.46
pronouns	9.04	9.72	8.12	-20.05	-19.32	-20.18
prepos.	13.72	14.19	12.66	-32.63	-31.23	-30.01
nouns	15.17	15.61	12.83	-26.09	-26.68	-25.18
max.χ^2	20.59	19.65	14.60	-38.77	-34.87	-29.32
avg.χ^2	23.01	20.95	20.62	-44.77	-39.45	-36.77
std.χ^2	16.78	14.92	11.89	-38.80	-33.70	-24.83
max.tf	14.82	14.26	12.75	-6.82	-6.70	-10.28
avg.tf	11.35	10.69	8.69	1.54	2.02	2.97
std.tf	11.95	11.48	10.51	-2.56	-1.82	-1.58
max.tfc	-9.12	-8.95	-6.89	-0.01	4.67	9.15
avg.tfc	-19.79	-20.15	-15.95	29.52	30.14	28.54
std.tfc	-15.06	-16.06	-14.12	6.50	10.54	25.41
max.ltc	-17.06	-16.64	-13.88	5.42	10.10	19.78
avg.ltc	-20.30	-20.92	-16.24	29.72	30.74	29.04
std.ltc	-16.59	-17.43	-14.99	3.89	8.38	25.91
max.idf	**26.36**	**25.34**	15.54	-37.14	-33.41	-28.69
avg.idf	24.66	22.69	**22.17**	**-47.04**	**-41.35**	**-37.50**
std.idf	15.16	12.72	6.91	-32.43	-28.30	-16.32

Table 6.15: Pearson correlation: TDP predictors.

	20newsgroups			**Reuters21578**		
	NB	**kNN**	**SVM**	**NB**	**kNN**	**SVM**
FBL	47.92	47.01	55.36	26.72	51.35	1.11
FBF	44.91	40.07	53.95	26.22	47.18	0.59
FBS	47.51	46.20	55.17	26.66	50.85	1.05
FAL	54.85	59.19	61.33	38.33	58.82	15.33
FAF	58.71	51.89	61.35	**66.87**	54.56	56.21
FAS	57.03	59.10	60.26	65.88	58.68	54.69
FAM	10.87	38.27	7.06	-0.33	2.25	19.77
FRL	61.46	48.57	**63.08**	38.22	51.70	24.12
FRF	61.44	42.75	60.59	50.52	47.95	42.82
FRS	62.23	48.57	61.65	54.66	51.93	44.08
FRM	10.08	36.40	7.10	-0.26	0.33	14.14
BBL	46.66	52.58	54.02	10.33	55.05	-12.40
BBF	41.41	42.19	51.92	5.88	48.20	-13.68
BBS	45.89	51.31	53.73	9.66	54.09	-12.63
BAL	52.89	**70.64**	60.35	14.08	**70.47**	-2.34
BAF	59.62	59.04	61.51	65.92	59.45	**56.63**
BAS	57.94	69.61	60.46	65.14	68.54	55.27
BAM	10.87	39.53	6.93	-0.32	2.48	19.77
BRL	61.52	54.56	62.88	24.43	55.30	12.69
BRF	61.52	45.68	60.71	38.32	49.40	35.56
BRS	**62.42**	53.36	61.55	48.36	54.83	40.91
BRM	9.87	38.26	7.03	-0.23	4.23	14.13

Tables 6.13 and 6.14 show the Pearson's and Spearman's correlation values (where the ratio has been scaled to $[-100,+100]$ for clarity reasons) for the pre-classification performance prediction methods. The best predictors are those based on `idf`, being `avg.idf` the best overall metric. The results illustrate that pre-classification predictors achieve correlations values up to 34% and 47%, applying Pearson and Spearman measures respectively in `Reuters-21578`, and up to 28% and 26% for `20-newsgroups`.

Tables 6.15 and 6.16 show the correlation scores for the TDP metrics, achieving consistently higher correlation than the pre-classification metrics, with values as high as 70% and 60% for Pearson and Spearman respectively.

6.2.6.3 Analysis

For both collections, the highest correlations based on pre-classification predictors were obtained using the Naive-Bayes classifier. A relevant insight is that, although the predictors based on linguistic features, and those based on classification specific information perform relatively well,

Table 6.16: Spearman correlation: TDP predictors.

	20newsgroups			**Reuters21578**		
	NB	**kNN**	**SVM**	**NB**	**kNN**	**SVM**
FBL	22.64	34.29	33.35	20.61	47.85	-06.49
FBF	22.64	33.92	33.35	20.61	47.15	-6.46
FBS	22.64	34.24	33.35	20.61	47.78	-6.46
FAL	29.08	44.02	42.93	24.41	51.65	-1.14
FAF	56.28	46.46	51.85	62.04	52.45	52.52
FAS	**56.96**	46.76	**51.85**	**62.13**	52.50	**53.07**
FAM	11.98	50.45	7.17	0.08	46.53	23.95
FRL	35.10	42.57	42.42	22.17	50.53	0.07
FRF	51.00	49.18	49.08	37.17	53.32	26.79
FRS	51.25	50.78	49.75	39.07	54.11	26.09
FRM	11.98	51.09	7.17	-0.56	48.58	19.35
BBL	19.66	31.70	43.81	0.21	44.49	-35.94
BBF	19.66	31.61	43.81	0.21	44.36	-35.94
BBS	19.66	31.61	43.81	0.21	44.34	-35.94
BAL	29.97	54.27	42.64	20.77	53.56	-7.69
BAF	55.80	47.61	50.77	59.58	52.34	51.16
BAS	56.60	**55.26**	50.83	60.16	**54.30**	52.16
BAM	11.98	49.90	7.17	0.08	43.43	23.95
BRL	32.40	42.33	40.20	17.09	49.33	-8.60
BRF	49.07	48.72	47.32	28.51	52.65	18.60
BRS	50.11	51.30	48.40	33.02	53.78	22.89
BRM	11.98	50.61	7.17	-0.56	47.63	19.35

they are inferior to `idf`-based predictors. This is similar to the results reported for QPP, where idf-based metrics are among the best pre-retrieval predictors. Both collections show similar absolute correlation values for different predictors, which is the main evaluation approach for performance prediction. However, the sign of the correlation changes in several cases and the underlying reasons for this difference should be investigated in future research. The different characteristics and structure of the collections could be causing this effect, specially the class distribution and the range of document length. The very poor performance of linguistic features in `20-newsgroups` is probably caused by the large differences in the number of tokens per document. *k*-NN is the best performer applying Pearson correlation, while NB shows larger values for Spearman.

Within the TDP framework, FAS with NB and SVM, and BAS for *k*-NN are consistently the best candidates for both collections, according to Spearman correlation, and some of the best performers according to Pearson. An unexpected results is that several metrics that do not consider the quality estimation perform well (i.e., BRS, BDS and BDL). This result suggests that the evidence and normalisation levels are the most representative factors to estimate the performance of the classifiers. The reason for the very low correlation for some predictors (i.e., BDM with k-NN for `Reuters-21578`) is that the normalisation process should be tuned for each classifier in order to optimise the process. For instance, tuning $\frac{x}{x+y}$ per model instead of using $\frac{x}{x+1}$. .

6.2.7 Discussion

Performance prediction has been applied to information retrieval, mainly for query performance prediction (QPP). We have presented and defined the concept of document performance prediction (DPP), and its challenges and opportunities have been analysed. Moreover, we have explored the differences and similarities between QPP and DPP, being the multidimensionality of the relevance, and the discrete number of possible quality values for a given document the most notable differences. Furthermore, the Threshold-based Document Performance (TDP) prediction framework has been introduced.

Document Performance Prediction has the potential to be applied to several applications, in order to improve different classification-related tasks. However, it presents two main challenges: Firstly, DPP should obtain high correlation with respect to the document quality. Otherwise, it might not be suitable for any of its potential applications. Secondly, performance prediction values are not applicable when multiple classifiers or representations are exploited. As a result, these values should be mapped to quality estimation values. Furthermore, any approach addressing this challenge should take into account the fact that each document has a finite (and usually small) number of possible quality values. Therefore, value discretisation should be applied as part of the mapping process from performance to quality values.

The experiments presented in this section show how traditional query performance prediction techniques can be applied for document performance prediction. The results show that relatively high levels of correlation (i.e., up to almost 50%) can be achieved. This supports the claim that document performance prediction is applicable. For the case of TDP metrics, the correlation significantly increases, to 70%. This result is critical, as the applicability of DPP require good predictive capabilities. The experiments also confirm that, similarly to QPP, post-classification methods are superior to pre-classification ones, and that for the latter, `idf`-based are the best performing methods, even outperforming the algorithms that rely on classification specific information

6.3 Near-Misses Evaluation

6.3.1 Preliminaries

The quality evaluation of classification results has been traditionally based on the ratio of misclassifications. This practice is based on the binary judgement that either a topic is relevant for a document or not. This strategy penalises misclassifications regardless of the closeness between the classified and the labelled topics. However, from a user perspective, some misclassifications are plausible (or at least understandable) mistakes, while others are extremely wrong. In other words, some misclassifications are more harmful, because the selected labels are very *distant* from the correct topics. For instance, the classification of a `Banking` document related to *"Santander Bank Policies"* as `Finance` is more desirable than assigning it to `Tourism`. However, both categories are incorrect and the traditional classification metrics penalise both decisions equally. This example is representative due to the fact that Santander is also a touristic town in northern Spain. Therefore, even if the semantic content of the document is clear, an automatic system might classify it in `Tourism`.

The final objective of an evaluation metric in classification is to measure the human satisfaction regarding the results produced by the system. Therefore, it is reasonable to penalise more the misclassifications that are less desirable. This implies that if a misclassification is perceived by the user to be further from the truth compared to another incorrect decision, an evaluation metric should take into account this difference. A similar argument was defended by [Deloo and Hauff, 2013]. Misclassification degrees has another benefit: consider a case where a document is correctly classified under a class by an automatic system, but it has not been correctly labelled by the human judges. A method that considers relationships between categories and their closeness can tackle the issues arisen from the inaccurate assessments. Of course, this situation is only possible if the assumption that human judgements are always correct is ignored. Nonetheless, research have shown that this assumption is not true, as the disagreement between assessors for both text classification and document retrieval obtain relatively high values [Voorhees, 2000] [Webber and Pickens, 2013]. Traditional evaluation measures are based on the inaccurate assumption that all mistakes are equally undesirable. This approach shares some similarities with cost sensitive learning (see Section 2.1.5), where the cost matrix would be based on multiple classes and their dependencies.

This section proposes two families of evaluation metrics that consider the degrees of misclassification. It also investigates how different classifiers perform according to such metrics, compared to traditional evaluation measures.

6.3.2 Degrees of Misclassification

The notion of misclassification degrees rely on the dependency or closeness between the wrongly assigned classes by the system and the correct one/s. The main principle is to decrease the penalisation for those mistakes with higher dependency with respect to the correct class/es. Figure 6.3 illustrates an example of document overlap for the set of classes {`Banking`, `Finance`, `Tourism`}. It shows the high correlation between `Finance` and `Banking` and that they are almost disjoint sets with respect to `Tourism`. In addition, it provides an understandable example to the fact that the dependency does not have to follow the symmetric property. Almost all documents belonging to `Banking` are labelled as Finance, while only around half of the `Finance` documents are in `Banking`. As a result, to classify a `Banking` document in `Finance` is perceived to be a less important mistake than to misclassify a `Finance` one in `Banking`. Similar challenges are present in several collections. For instance, `20-newsgroups` (see Section 2.8) has groups that would be difficult to differentiate even for a human judge (e.g. `talk.religion.misc` *vs* `soc.religion.christian`). Furthermore, the fact that this collection is single-labelled implies that if a document is classified in `talk.religion.misc` when it was labelled as `soc.religion.christian`, it would be penalised as a false positive for one class, and as a false negative for the other. Other example is the classification of a document which talks about guns in the middle east, because there are two very relevant labels for the document: `talk.politics.guns` or `talk.politics.mideast`.

Degrees of misclassification evaluation is more suited when the chaotic and the "no-perfect-judgements" nature of collections are taken into account, and when single-label collections are considered. The approaches presented in this section exploit a dependency matrix D, which measures the dependency between the classes assigned by the system and the correct labels. The main evaluation goal is that the higher similarity between them, the lower penalisation in terms of quality should be applied for the misclassification. If the class dependencies are computed based on the document overlapped shown in Figure 6.3, it will generate the matrix shown below, where $D_{i,j}$ represents the dependency between the classification of class j by the system with

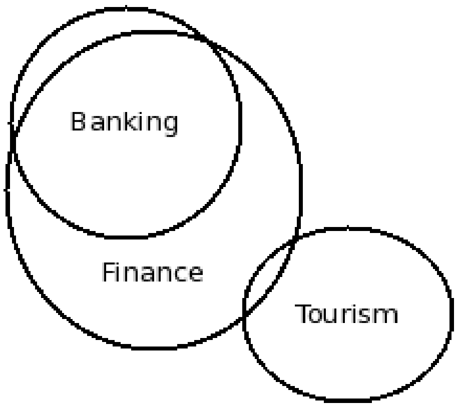


Figure 6.3: Example of Documents Overlap between different Classes.

respect to the correct labelled class i.

$$D = \begin{pmatrix} 1 & 0.5 & 0 \\ 0.8 & 1 & 0.05 \\ 0 & 0.01 & 1 \end{pmatrix} \tag{6.32}$$

For instance, given the set of categories $C = \{\texttt{Banking}, \texttt{Finance}, \texttt{Tourism}\}$, $D_{2,1}$ represents the probability that Finance is a near-miss assignation for a document assigned to Banking. This can be seen as a conditional probability between categories where $D_{i,j} = P(i|j)$. This setting assumes that all similarity values in D are in the range [0, 1]. Different strategies to compute category similarities based on document sets, textual content and semantic relations, are discussed in the next sections. The correct labels, as assessed by human experts, are represented by matrix H, while the automatic classification decisions are specified by M. Both are $m \times n$ matrices (m being he number of classes and n the number of documents to be classified). In H, each position i, j has a value of one, if the i-th class is assigned to the j-th document, or a zero otherwise. If degrees of misclassifications are included, H is transformed into H^D. For the correct labels, H^D is defined as the original correctness value, and for the incorrect labels, it assigns the maximum dependency of the assigned class with respect to a correct label.

$$H^D_{i,j} = \begin{cases} 1 & \text{if } i = j \\ \max_x(\{D_{x,j} \cdot H_{x,i}) & \text{otherwise} \end{cases} \tag{6.33}$$

One of the main priorities is that any correct assignment obtains the same quality as it would without considering near-misses. All the strategies focus on minimising the penalisation for

Symbol	Explanation
C	Set of classes
$D_{i,j}$	Dependency between the classification of class j with respect to the class i
$M_{i,j}$	System assignment for the i-th class and j-th document
$H_{i,j}$	Human assessment for the i-th class and j-th document
$L_{i,j}$	Correct system assignment for the i-th class and j-th document
$H^D_{i,j}$	Near-misses assessment for the i-th class and j-th document
$L^D_{i,j}$	Near-misses degree of correct assignment for the i-th class and j-th document
τ	Threshold to assume a near misses is a correct assignation for F_{1_U}

Table 6.17: Notation Summary for Near Misses Evaluation.

wrong assignations, while the correct classifications are evaluated normally. As a result, it can be asserted that any metric following this principles will compute a quality value which is higher or equal to its equivalent without considering dependencies. The next sections focus on the evaluation metrics and the class similarity estimation. Table 6.17 summarises the common notation.

6.3.3 Dependency-Aware Evaluation Measures

Two algorithms are designed to analyse degrees or misclassification and incorporate category dependencies in the evaluation. The first method is a variation of F_1 that applies a threshold to "interpret" that a classification is correct if a certain similarity with a labelled (assessed) topic is achieved. The second algorithm is based on a probabilistic interpretation of F_1 and it incorporates the probability of a mistake being correct in the calculations. The design of the methods implies that, for the same set of classified documents, they achieve equal or higher absolute quality than traditional metrics. The reason for this is the fact that they are based on the minimisation of "punishment" of misclassified documents.

6.3.3.1 Class-Dependency Aware F_1 (F_{1_U})

F_{1_U} modifies the definition of false positives in F_1 to include class similarity information. Traditionally, false positives count the number of labels incorrectly classified in a class (see Section 2.7). In our approach, FP is reformulated [2] using an array formulation as Equation 6.35 shows, where L models the specific pairs class-document that were correctly labelled.

$$L_{i,j} = M_{i,j} \cdot H_{i,j} \tag{6.34}$$

$$\mathrm{FP} = \sum_{i,j} (M - L)_{i,j} \tag{6.35}$$

The same principle can be applied with the near-misses assessments, providing a measure of partial correctness (L^D) as presented in Equation 6.38. Then, a variation of FP, coined FP_U, is presented in Equation 6.37. FP_U subtracts the assignations made by the system by the partial correct assignations L^D, computing the aggregation of partial penalisations

$$L^D_{i,j} = M_{i,j} \cdot H^D_{i,j} \tag{6.36}$$

$$\mathrm{FP}_U = \sum_{i,j} (M - L^D)_{i,j} \tag{6.37}$$

This solution changes the nature of the contingency table by using real values instead of number of documents. As a result, it can not be used while maintaining the theoretical foundations of the method. This issue is addressed by assuming that the topics with high dependency with respect to correct classes are also correct. A threshold τ is used to filter the dependency level needed to be considered an“indirect” correct class. Therefore, the foundations and mechanisms applied in the definition of F_1 are met. F_{1_U} modifies F_1 by assuming that the assignments that have a dependency with a correct class greater than a threshold value τ are correct.

$$L^D_{i,j} = \begin{cases} 1 & \text{if } M_{i,j} \cdot H^D_{i,j} > \tau \\ 0 & \text{otherwise} \end{cases} \tag{6.38}$$

[2]The computation remains the same as in traditional FP

$$\mathrm{TP}_U = \sum_{i,j} (L^D)_{i,j} \tag{6.39}$$

These variations lead to new versions of Precision, Recall and F_1, (Pr_U, Re_U and F_{1_U}) which include misclassification degrees:

$$\mathrm{Pr}_U = \frac{\mathrm{TP}_U}{\mathrm{TP}_U + \mathrm{FP}_U} \tag{6.40}$$

$$\mathrm{Re}_U = \frac{\mathrm{TP}_U}{\mathrm{TP} + \mathrm{FN}} \tag{6.41}$$

$$\mathrm{F}_{1_U} = \frac{2 \cdot \mathrm{Pr}_U \cdot \mathrm{Re}_U}{\mathrm{Pr}_U + \mathrm{Re}_U} \tag{6.42}$$

6.3.3.2 *Exhaustivity-Specificity (ES)*

Exhaustivity and Specificity measures are based on probabilistic interpretations of F_1. Exhaustivity measures, given a class, the probability of a document being correctly classified if it has been labelled in that class by a human. This directly relates to Recall. On the other hand, Specificity measures the probability of a document being correctly classified, given that it has been assigned by the system. This measures are then defined as follows, using the matrix notations previously introduced.

$$\begin{aligned} E_i &= P(\mathrm{correct}(c_i)|\mathrm{assessed}(c_i)) \\ &= \frac{\sum_j M_{i,j} \cdot H^D_{i,j}}{\sum_j H_{i,j}} = \frac{\sum_j L^D_j}{\sum_j H_{i,j}} \end{aligned} \tag{6.43}$$

$$\begin{aligned} S_i &= P(\mathrm{correct}(c_i)|\mathrm{classified}(c_i)) \\ &= \frac{\sum_j M_{i,j} \cdot H^D_{i,j}}{\sum_j M_{i,j}} = \frac{\sum_j L^D_j}{\sum_j M_{i,j}} \end{aligned} \tag{6.44}$$

These metrics incorporate the near-misses information by using the matrix H^D to report partially correct classified topics. After this, following the same strategy as F_1, the harmonic mean is computed to give an overall and unique quality measure per class, referred to as ES,

$$ES_i = \frac{2 \cdot E_i \cdot S_i}{E_i + S_i} \tag{6.45}$$

Both micro and macro-averaged strategies can be applied to any of the measures obtain a general quality value for a set of classes.

6.3.4 Class Dependency Estimation

F_{1_U} and ES rely in the concept of dependency between the classes. This section summarises different families of metrics that can be applied for this task. The main goal of this section is to show the flexibility of the evaluation methods and to introduce the metrics that are used in the experimental section.

6.3.4.1 Set-Based

The similarity between classes can be computed by measuring their documents overlap. The more documents labelled in both classes, the more similar the classes are. Equation 6.46 provides one example of this type of method, where the dependency is based on the ratio between the intersection of documents in both classes and the number of documents in classes. This equation represents $P(C_j|C_i)$ within the document space and it is a non-symmetric measure.

$$D_{overlap_{i,j}} = \frac{|C_i \cap C_j|}{|C_i|} \tag{6.46}$$

Any other method based on overlap between sets such as Dice or Jaccard coefficients (illustrated in Equations 6.47 and 6.48 respectively) can be used.

$$D_{dice_{i,j}} = \frac{2 \cdot |C_i \cap C_j|}{|C_i| + |C_j|} \tag{6.47}$$

$$D_{jaccard_{i,j}} = \frac{|C_i \cap C_j|}{|C_i \cup C_j|} \tag{6.48}$$

The main issue with set-based dependency measures is that the dependency calculations depend on the taxonomy and indexing policy of a given collection. Therefore, the overlap ratio can be very small even for similar classes. In addition, they are only applicable for multi-label collections because there is no document overlap in single-label or multi-class environments. As a result, a set-based class similarity applies complete independence between classes if applied to single-label collections. Therefore, it computes the same quality value as the traditional F_1.

6.3.4.2 Content-Based

Content-Based dependency refers to any method that exploits the textual content of documents within each class. One strategy is to compute the cosine similarity between their "representative" documents. For this reason, the centroid for each category is obtained, following Equation 6.49, where $w(t,d)$ represents the weight for the term t in document d and D_{c_i} is the set of documents belonging to the i-th class.

$$\text{centroid}(c) = \frac{1}{|D_{c_i}|} \sum_{[t \in d; d \in c_i]} w(t,d) \tag{6.49}$$

$$D_{centroid_{i,j}} = \text{sim}(\text{centroid}(c_i), \text{centroid}(c_j)) \tag{6.50}$$

The main benefit of this family of measures is that the similarity between classes is independent of indexing policies and types of collections and it more accurately represents semantic closeness.

6.3.4.3 Structural or Relational

Structural or relational information between classes provides a helpful source of information that can be exploited to analyse their dependencies. For instance, given a hierarchical collection, specific values of dependency can be specified for sub-topics and "sibling" classes.

$$D_{structural_{i,j}} = \begin{cases} 1 & \text{if } i = j \\ 0.75 & \text{if } c_j \subset c_i \\ 0.5 & \text{if } c_i \subset c_j \\ 0.25 & \text{if } c_j \subset c_x \wedge c_i \subset c_x \\ 0 & \text{otherwise} \end{cases} \tag{6.51}$$

Equation 6.51 defines a practical example based on hierarchical information. Similarity between classes is defined as 1.0 for the same class, 0.75 for the parent class given a child, 0.5 in the opposite case and 0.25 for sibling topics. The obvious limitation of this method is that it requires a hierarchical collection or some type of relational information between different topics. Different techniques have been presented to evaluate hierarchical classification [Sun and Lim, 2001]. The main focus of our research is to use degrees of classification in non-hierarchical traditional classification environments. Moreover, the capability to model hierarchical dependencies is shown to underline the flexibility of the approach.

6.3.4.4 Human/Expert-Based

The usage of a dependency matrix allows a straight-forward manual specification of these weights based on domain-specific and/or expert knowledge. Therefore, the matrix can be used to explicitly represent the satisfaction degree for some misclassification. From a practical perspective, any dependency method can be used to obtain a starting dependency matrix which will then be modified by an expert or a final user, thanks to the understandable definition. Furthermore, a human-based strategy to modify the dependency matrix allows, even if the values are initially based on other strategies, different dependency matrix for different users. As a result, a personalised near-misses evaluation can be achieved.

6.3.5 Evaluation

The final objective of an evaluation metric in classification is to measure the human satisfaction based on the decisions made by the system. Therefore, it is reasonable to penalise more the misclassifications that are less desirable. In other words, if a misclassification is perceived by the user to be further from the truth, compared to another incorrect decision, an evaluation metric should take this difference into account. The evaluation of degrees of misclassification was addressed in Section 6.3. This section analyses how the relative ranking of classifiers changes when degrees of misclassification are included, with respect to traditional evaluation measures.

6.3.5.1 Set-Up

Two collections have been used for the experiments: `20-newsgroups` and `Reuters-21578`. Three different families of classifiers have been used: Naive Bayes (Weka [Hall et al., 2009]), k-NN (our own implementation) and SVM using LibSVM [Chang and Lin, 2011]. All documents have been weighted using `ltc` and feature selection based on χ^2 has been applied. Different parameters have been tested for each model and the configuration that achieves better micro-averaged F_1 quality has been selected. The number of features considered is 3000 for

k-NN and NB and 10,000 for SVM; and the number of neighbours for k-NN is 60. In all cases the SCutFBR$_{.1}$ thresholding strategy has been used, applying a 5-fold cross-validation process. In the case of SVM the scored output is obtained by running LibSVM [Chang and Lin, 2011] with the option "-b 1". For the centroid metric, the similarity computation is based on the representation of documents with 3000 features based on χ^2, independently of the specific classifier. In all the experiments, the value of τ has been manually set to 0.4, based on the dependency matrices. Those dependencies with higher value than τ are presented in bold in Tables 6.18-6.21.

Table 6.18: Dependency matrix (%) between the top 10 most common classes of `Reuters-21578` using dependency as similarity metric. Values higher than 0.4 are shown in bold.

	acq	crude	earn	grain	interest	moneyfx	ship	trade	corn	wheat
acq	**100.0**	3.6	0.56	0.46	0.0	0.19	2.54	0.82	1.1	0.0
crude	0.85	**100.0**	0.45	0.46	0.29	0.19	20.3	1.9	0.0	0.94
earn	0.97	3.34	**100.0**	0.0	0.0	0.0	0.51	0.0	0.0	0.0
grain	0.12	0.51	0.0	**100.0**	0.0	0.19	14.21	2.17	**98.9**	**98.58**
interest	0.0	0.26	0.0	0.0	**100.0**	26.21	0.0	1.63	0.0	0.0
moneyfx	0.06	0.26	0.0	0.23	**40.63**	**100.0**	0.0	8.42	0.55	0.0
ship	0.3	10.28	0.03	6.47	0.0	0.0	**100.0**	0.54	2.21	3.3
trade	0.18	1.8	0.0	1.85	1.73	5.76	1.02	**100.0**	0.55	1.89
corn	0.12	0.0	0.0	**41.34**	0.0	0.19	2.03	0.27	**100.0**	27.83
wheat	0.0	0.51	0.0	**48.27**	0.0	0.0	3.55	1.09	32.6	**100.0**

Table 6.19: Dependency matrix (%) between the top 10 most common classes of `Reuters-21578` using centroid as similarity metric. Values higher than 0.4 are shown in bold.

	acq	crude	earn	grain	interest	moneyfx	ship	trade	corn	wheat
acq	**100.0**	35.09	28.39	25.58	28.53	29.68	28.22	32.12	21.75	21.73
crude	35.09	**100.0**	17.79	28.34	26.91	28.21	39.88	38.84	24.38	23.3
earn	28.39	17.79	**100.0**	11.48	13.35	13.21	11.17	15.1	9.42	9.48
grain	25.58	28.34	11.48	**100.0**	21.32	24.24	38.69	37.87	**89.83**	**93.3**
interest	28.53	26.91	13.35	21.32	**100.0**	**80.27**	19.56	38.02	17.81	17.9
moneyfx	29.68	28.21	13.21	24.24	**80.27**	**100.0**	22.48	**51.42**	20.39	20.24
ship	28.22	39.88	11.17	38.69	19.56	22.48	**100.0**	32.22	30.53	31.28
trade	32.12	38.84	15.1	37.87	38.02	**51.42**	32.22	**100.0**	32.64	31.05
corn	21.75	24.38	9.42	**89.83**	17.81	20.39	30.53	32.64	**100.0**	**74.46**
wheat	21.73	23.3	9.48	**93.3**	17.9	20.24	31.28	31.05	**74.46**	**100.0**

6.3.5.2 Results

The evaluation methods considering degrees of misclassification that were introduced in Section 6.3 rely on the calculation of category dependencies. One set-based and one content-based dependency methods are applied in the experiments: overlap and centroid (see Equations 6.46 and 6.50). The utility matrices for different models and collections are shown in Tables 6.18-6.21. The table for the overlap-based dependency in `20-newsgroups` shows a 0% dependency between each pair of different topics. The reason for this is that a set-based metric can not be

Table 6.20: Dependency matrix (%) for a sample of 10 classes in `20-newsgroups` using dependency as dependency metric. Values higher than 0.4 are shown in bold.

	motorcycles	x	mideast	guns	misc	autos	misc	graphics	christian	misc
rec.motorcycles	**100.0**	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
comp.windows.x	0.0	**100.0**	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
talk.politics.mideast	0.0	0.0	**100.0**	0.0	0.0	0.0	0.0	0.0	0.0	0.0
talk.politics.guns	0.0	0.0	0.0	**100.0**	0.0	0.0	0.0	0.0	0.0	0.0
talk.religion.misc	0.0	0.0	0.0	0.0	**100.0**	0.0	0.0	0.0	0.0	0.0
rec.autos	0.0	0.0	0.0	0.0	0.0	**100.0**	0.0	0.0	0.0	0.0
talk.politics.misc	0.0	0.0	0.0	0.0	0.0	0.0	**100.0**	0.0	0.0	0.0
comp.graphics	0.0	0.0	0.0	0.0	0.0	0.0	0.0	**100.0**	0.0	0.0
soc.religion.christian	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	**100.0**	0.0
comp.os.ms-windows.misc	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	**100.0**

Table 6.21: Dependency matrix (%) for a sample of 10 classes in `20-newsgroups` using centroid as similarity metric. Values higher than 0.4 are shown in bold.

	motorcycles	x	mideast	guns	misc	autos	misc	graphics	christian	misc
rec.motorcycles	**100.0**	29.11	24.88	33.15	30.06	**47.41**	34.8	31.71	27.15	29.54
comp.windows.x	29.11	**100.0**	23.1	26.52	28.59	31.5	30.47	**55.69**	25.84	**56.7**
talk.politics.mideast	24.88	23.1	**100.0**	39.47	38.25	28.47	**43.31**	24.72	35.6	21.65
talk.politics.guns	33.15	26.52	39.47	**100.0**	**43.64**	38.68	**52.84**	28.93	36.31	27.24
talk.religion.misc	30.06	28.59	38.25	**43.64**	**100.0**	33.37	**45.63**	30.8	**66.84**	27.25
rec.autos	**47.41**	31.5	28.47	38.68	33.37	**100.0**	39.81	35.24	30.51	34.18
talk.politics.misc	34.8	30.47	**43.31**	**52.84**	**45.63**	39.81	**100.0**	32.17	**42.52**	29.54
comp.graphics	31.71	**55.69**	24.72	28.93	30.8	35.24	32.17	**100.0**	28.21	**56.86**
soc.religion.christian	27.15	25.84	35.6	36.31	**66.84**	30.51	**42.52**	28.21	**100.0**	24.65
comp.os.ms-windows.misc	29.54	**56.7**	21.65	27.24	27.25	34.18	29.54	**56.86**	24.65	**100.0**

applied to single-label collections such as `20-newsgroups`. Tables 6.22 and 6.23 illustrate the micro and macro-averaged quality measure for all the different methods and collections using $\mathrm{F1}_U$ and ES respectively. A third dependency model, referred to as "Traditional", is also introduced. It represents a similarity matrix with zero in every position except the diagonal. As a result, no dependencies between different classes are considered and both evaluation metrics $F1_U$ and ES compute the same quality as traditional F_1. The tables show the improvement of each model with respect to the baseline, which is the worst performing classifier using the same evaluation model in the same collection. For instance, for a k-NN classifier in `Reuters-21578` evaluated using ES with centroid similarity, the baseline is Naive-Bayes, because it is the model with lower ES quality in that collection. The improvement calculation is shown below, where q represents a specific evaluation measure, b is the baseline model, and m is the model being evaluated:

$$\mathrm{improv}(q,m) = \frac{q(m) - q(b)}{q(b)} \tag{6.52}$$

The main objective of this section is to investigate how the dependency-aware evaluation models modify the relative quality and ranking of classifiers. However, the different nature of traditional and dependency-aware evaluation measures makes it impossible to apply a direct comparison between their absolute values. For this reason, the strategy used in this research is to analyse the changes in the relative improvement with respect to the baseline, for each of the evaluation metrics. Therefore, the variation between the improvement of each model with respect to the baseline and the improvement of the same model using a traditional evaluation are used. Variations greater than zero imply that that model improves more when using dependency-aware models. Therefore, its misclassifications are less damaging than those done by the baseline model. The relative quality results are presented in Tables 6.24 and 6.25, and they are also graphically illustrated in Figures 6.4 and 6.5. The relative improvement measure is defined below, where q represents a evaluation metric, e is the dependency metric, t models the traditional dependency matrix and m specifies the classification model.

$$\text{improv}_{\text{rel}}(q,m,e) = \frac{\text{improv}(q_e,m) - \text{improv}(q_t,m)}{\text{improv}(q_t,m)} \tag{6.53}$$

For instance, NB and k-NN achieve a quality of 0.81 and 0.76 in `20-newsgroups` respectively when using traditional F_1 evaluation. This represents an improvement of 4.6% for NB with respect to the baseline (k-NN). On the other hand, if a centroid dependency evaluation is applied, the quality of NB and k-NN are 0.90 and 0.87 respectively, and the improvement of NB is reduced to 2.39%. As a result, the relative difference of improvement for NB when changing from traditional to centroid-based evaluation is -48.09% (from 4.6% to 2.39%) . This implies that, in case of a misclassification, k-NN mistakes are less damaging than those made by NB. Nonetheless, it also shows that NB still performs better than k-NN with both metrics.

6.3.5.3 Analysis

The comparison between the overlap and the cosine dependency matrices for `Reuters-21578` illustrates the fact that different dependency measures compute very different dependency matrices. Overlap values range from 0 to around 98%. However, if the overlap between grain, corn, and wheat are ignored, the maximum dependency drops to 48%. In addition, most of the dependencies have very low values. The similarity matrix ranges from 9% to 93%, with most classes having dependencies close to 20%. These suspected differences on the dependency values imply that any parameter based on them should be tuned specifically for that configuration. This supports the idea that ES is better suited than F_{1_U}, due to the fact that ES is parameter-less, while F_{1_U} requires the tuning of the τ parameter. Another important factor for the dependency

Table 6.22: Quality using F_{1_U} measure. Improvements with respect to the baseline (%) are shown between brackets.

		micro			**macro**		
		centroid	overlap	traditional	centroid	overlap	traditional
Reuters21578	NB	83.77	81.15	79.94	34.64	33.16	32.38
	kNN	89.27 (5.5)	87.26 (6.11)	85.81 (5.87)	**63.27** (28.64)	**62.28** (29.12)	**59.63** (27.26)
	SVM	**89.41** (5.63)	**88.06** (6.91)	**86.27** (6.33)	62.81 (28.17)	61.88 (28.72)	58.75 (26.38)
20newsgroups	NB	87.3 (3.73)	80.92 (4.6)	80.92 (4.6)	86.54 (3.8)	80.3 (4.62)	80.3 (4.62)
	kNN	83.57	76.32	76.32	82.74	75.69	75.69
	SVM	**88.87** (5.31)	**84.33** (8.01)	**84.33** (8.01)	**88.25** (5.51)	**83.76** (8.07)	**83.76** (8.07)

Table 6.23: Quality using ES measure. Improvements with respect to the baseline (%) are shown between brackets.

		micro			**macro**		
		centroid	overlap	traditional	centroid	overlap	traditional
Reuters21578	NB	87.93	83.15	79.94	34.16	33.16	32.38
	kNN	**92.79** (4.87)	89.13 (5.98)	85.81 (5.87)	**62.61** (28.45)	**62.07** (28.91)	**59.63** (27.26)
	SVM	92.36 (4.44)	**89.89** (6.74)	**86.27** (6.33)	61.63 (27.47)	61.12 (27.96)	58.75 (26.38)
20newsgroups	NB	89.77 (2.39)	80.92 (4.6)	80.92 (4.6)	83.89 (3.92)	80.3 (4.62)	80.3 (4.62)
	kNN	87.39	76.32	76.32	79.96	75.69	75.69
	SVM	**90.56** (3.17)	**84.33** (8.01)	**84.33** (8.01)	**86.39** (6.43)	**83.76** (8.07)	**83.76** (8.07)

measures is to observe how the values of a specific algorithm change depending on the collection. The tables show a similar distribution of values for both collections when using a cosine based similarity. `20-newsgroups` shows less sparsity in the values with a range from 20-66%, whereas `Reuters-21578` values range from 9% to 93%. The reason for this difference probably is that this collection's distribution of classes is almost uniform, whereas `Reuters-21578` has a very skewed distribution. The fact that `20-newsgroups` is a single-label collection makes it impossible to compare it with the overlap metric.

The class-dependency evaluation metrics achieve higher absolute quality values than the traditional version, as they introduce less penalisation for the mistakes. Tables 6.22 and 6.23 show how the relative improvements change considerably. Furthermore, in one case the ranking of classifiers changes. In the `Reuters-21578` collection, SVM is the best performing model with micro-averaged traditional evaluation, while k-NN outperforms it when centroid-based dependency is applied with the ES metric.

Tables 6.24 and 6.25 illustrate how the relative difference of improvements of some classifiers are consistently better than others when near-misses are taken into account. In `Reuters-21578`, NB is the worst algorithm in all conditions, with the exception of micro-averaged quality using centroid dependency. The improvement of NB, if micro-average is used, can be caused by the low quality achieved by the traditional (without considering near-misses) evaluation. SVM and k-NN have a similar behaviour for this collection. `20-newsgroups` presents a much clearer

Table 6.24: Relative difference (%) of the improvements using F_{1_U} and the improvements using traditional F_1.

		micro		macro	
		centroid	overlap	centroid	overlap
Reuters21578	NB	0.0	0.0	0.0	0.0
	kNN	-6.24	4.14	5.06	6.84
	SVM	-10.98	9.15	6.8	8.88
20newsgroups	NB	-18.87	0.0	-17.71	0.0
	kNN	0.0	0.0	0.0	0.0
	SVM	-33.75	0.0	-31.81	0.0

Table 6.25: Relative difference (%) of the improvements using ES and the improvements using traditional F_1.

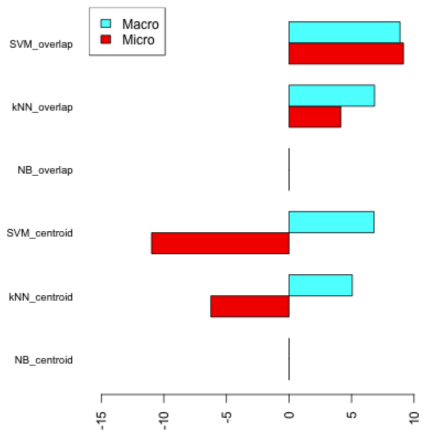
		micro		macro	
		centroid	overlap	centroid	overlap
Reuters21578	NB	0.0	0.0	0.0	0.0
	kNN	-17.05	1.99	4.38	6.06
	SVM	-29.86	6.51	4.13	6.0
20newsgroups	NB	-48.06	0.0	-15.0	0.0
	kNN	0.0	0.0	0.0	0.0
	SVM	-60.4	0.0	-20.39	0.0

message. k-NN significantly outperforms all the other models when near-misses are considered. This means that if a mistake is made by k-NN it is much closer to the real class than it would be by NB or SVM. Furthermore, this result is consistent with both evaluation methods.

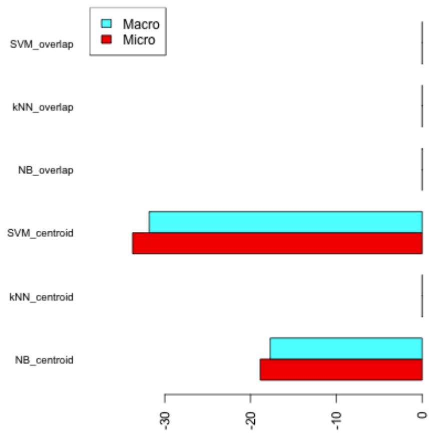
6.3.6 Discussion

The importance of dependencies in text classification is a known fact. In addition, some mistakes are worse than others from a human perspective, and the final goal of the evaluation process in classification is to measure the user satisfaction. Therefore, degrees of misclassification is a factor that should be considered. This section has discussed and proposed complementary evaluation measures based on class dependency metrics and the notion of degrees of misclassification. Two evaluation measures (ES and F_{1_U}) have been introduced. Their general behaviour is similar. However, ES is considered the best metric, as it is parameter-less, it does not involve any tuning process, and it uses the dependency degree in the final quality computation.

This research has investigated the concept of degrees of misclassification to produce an evaluation that represents the user satisfaction better than current approaches. Different methods to incorporate such knowledge into an evaluation function have been analysed and the quality ranking of models varies in one case. k-NN outperforms the traditional best model (SVM) in

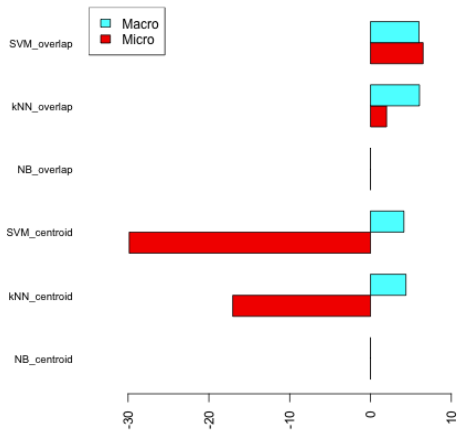


(a) Reuters-21578

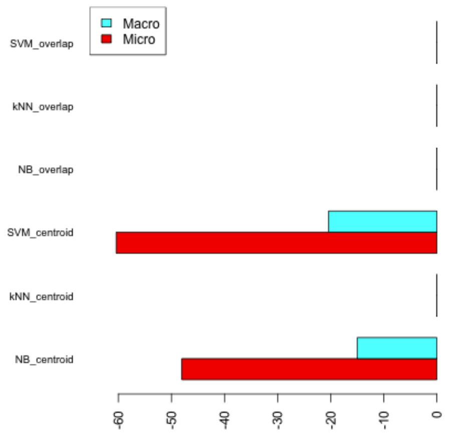


(b) 20newsgroups

Figure 6.4: Relative difference (%) of the improvements using micro-average F_{1_U} and the improvements using traditional F_1.



(a) Reuters-21578



(b) 20newsgroups

Figure 6.5: Relative difference (%) of the improvements using micro-average ES and the improvements using traditional F_1.

`Reuters-21578`, if *ES* is applied using the centroid similarity between classes as the dependency metric. Moreover, the relative improvement of classifiers change significantly when degrees of misclassification are incorporated into the evaluation measures. NB is the model which decreases its relative improvement the most. On the other hand, *k*-NN is the most stable one, as it performs as well as SVM in `Reuters-21578`, while it improves by up to 20% and 60% its relative macro and micro-averaged quality for `20-newsgroups`. These findings support the intuition that the evaluation of classification should consider the notion of near-misses to represent the human satisfaction more closely, in order to provide a more realistic measurement. Both evaluation measures, ES and F_{1_U}, behave similarly. However, ES is considered the best metric, as it is parameter-less, and it uses the dependency degree in the final quality computation. The suitability of these methods for collections with noisy and/or missed judgement assessments represents a very interesting path, as well as the analysis of individual misclassifications.

6.4 Dynamic k-NN

6.4.1 Preliminaries

k-Nearest Neighbours (k-NN) is a lazy learning method that has been used for classification for the last 40 years [Sebastiani, 2002] (explained in detail in Section 2.5.2). It exploits the similarity of a document with respect to each training document to select the k "nearest" examples to learn from. The number of neighbours considered in the classification process (k) is its most important parameter. It is traditionally optimised, using cross-validation, in order to maximise its performance.

Multi-label classification problems can be addressed by generating n binary classifiers, where n is the number of classes. Strategies such as "one-vs-rest" have been applied. This is a common strategy for some classifiers such as SVM, which has been reported to achieve better results than k-NN for text classification [Joachims, 1998]. We argued that a global k optimisation, applied in traditional k-NN, unfairly weakens k-NN comparison with methods like SVM that optimise each two-class problem separately. The creation of n binary classifiers has been rarely (if ever) used with k-NN probably because of its computational cost in the classification phase. This research is driven by the hypothesis that optimising k globally represents a sub-optimal solution, while a class-based optimisation achieves better quality. The methods that dynamically modify the number of neighbours based on different factors are known as adaptive k-NN (see Section 3.2.4).

This section analyses the global optimisation of the number of neighbours in k-NN, and introduces Dynamic k-NN (δk-NN), a variation of k-NN that applies class-based k optimisation. δk-NN selects the best k for each class, based on the quality estimations for traditional k-NN. Furthermore, a maximum quality boundary, if all the quality estimations are perfect, is illustrated and analysed. δk-NN does not involve any added computational cost because it is based on the steps normally applied for k-NN.

From a theoretical perspective, k-NN leads to a decision function that approximates the optimal Maximum a Posteriori (MAP) decision rule [Bishop, 1995]. Density estimation is implicitly carried out by looking for the closest neighbours, and the parameter k controls the amount of regularisation involved in the estimate. A global k optimisation ignores the fact that different classes may come from different densities and with variable sample sizes, which may pose conflicting requirements on the regularisation parameter. For instance, classes with few training documents and unimodal distributions may benefit from a smoother estimate (larger k), while a well-sampled multimodal distribution may be better approximated by a smaller k.

Before explaining the process followed by δk-NN, the tuning steps when k-NN is applied to a new dataset are explained below. The main reason is to illustrate that δk-NN does not involve any additional computational cost because it exploits the knowledge generated in the tuning phase for the traditional k-NN.

1. The collection is divided into training, validation and testing subsets.
2. A set of potential k values is selected to be tested (e.g., $\{5, 10, 15, ... 50\}$).
3. For each preselected k value, the best thresholds (based on the validation set) and quality values (using the testing documents) per category are computed.

After the general tuning of k-NN, the information about the best threshold and quality per class is available, for each one of the considered k values.

6.4.2 δk-NN Scoring

To optimise the number of neighbours on a category level can provide better quality for the k-NN classifier. However, this would imply the creation of n binary k-NN classifiers, which significantly increases the computational cost in classification time. δk-NN optimises k for each category without the increased computational cost required if multiple binary classifiers are used. It selects a different number of neighbours for each class, based on the quality estimation for each pair $(k, class)$. How this estimation is calculated is the focus of Section 6.4.3. Equation 6.54 shows the definition of δk-NN, where the use of a class c as a parameter for the set of nearest neighbours (*NN*) is the only difference compared to traditional k-NN. Equation 6.55 defines the set of nearest neighbours, for a specific document and class.

$$\text{score}_{\delta\text{-}k\text{-NN}(c,d)} = \frac{1}{|\text{NN}_k(d,c)|} \sum_{[d_i \in \text{NN}_k(d,c)]} \text{sim}(d,d_i) \cdot y(d_i,c) \tag{6.54}$$

$$\begin{aligned} \text{NN}_k(d,c) &= \{d' \in D' : \text{sim}(d,d') \geq \text{sim}(d,d_i), \\ &\quad \forall d_i \in D \backslash D' \wedge |D'| = k(c)\} \end{aligned} \tag{6.55}$$

NN depends on a document and a specific category. This strategy can be seen as implicitly dividing a multi-label k-NN algorithm into n binary k-NN classifiers that will be aggregated using the "one-vs-rest" approach. If a k-NN classifier is implemented for each class, each binary classifier will be computed independently and the optimal threshold would be calculated for the specific category. Furthermore, each one of them will produce a score for any document measuring its belonging to the class. δk-NN indirectly follows the same principle, without the drawbacks of having to implement n different classifiers.

As explained before, k-NN implicitly relies on an adaptive density estimation step for class-conditional densities, given by the following formulation, where n_c is the number of documents of class c contained in the volume element ΔV_k of feature space, N_c is the total number of training examples for class c and the discretisation volume ΔV_k is chosen to contain exactly k of the training samples closest to d.

$$P(d|c) = \frac{n_c}{N_c \Delta V_k} \tag{6.56}$$

This can be seen as an adaptive "binning" of feature space, where the size of the bins instead of being constant (as is the case of histogram estimation) is allowed to vary with sample density. The bin size cancels out when the class-conditional densities are substituted in the maximum a posteriori rule to give the k-NN classification decisions. With this in mind, it should be evident that an adaptive choice of the parameter $k = k(c)$ to suit the different underlying distribution and sampling sizes of the various classes should lead to better density approximation and therefore better results.

6.4.3 Estimating the Optimum k per Class

The estimation of the best number of neighbours for each class is a critical part of δk-NN. After the tuning of k-NN, the estimated quality and best threshold for each class and k value ($q_k(c)$, $\tau_k(c)$) are available, as shown in Algorithm 1. Given this information, k-NN and δk-NN select the k values following different strategies (see Algorithms 2 and 3 respectively).

Algorithm 1 Tuning steps for k-NN.

```
Input: $\vec{k}$ {set of considered k values}
Input: $\vec{c}$ {set of classes}
1: for all $k$ in $\vec{k}$ do
2:     $\vec{\tau}_k, \vec{q}_k \leftarrow crossValidation(k)$
3: end for
```

Algorithm 2 Optimisation of k parameter in k-NN.

```
Input: $\vec{k}$ {set of considered k values}
Input: $\vec{c}$ {set of classes}
Input: $\vec{\tau}_k$ {thresholds for each $k$}
Input: $\vec{q}_k$ {quality estimation for each $k$}
1: $k \leftarrow \{k \in \vec{k} : \mathrm{argmax}_k q_k\}$
2: $\tau_k \leftarrow \{\tau \in \vec{\tau}_k, k \in \vec{k} : \mathrm{argmax}_k q_k\}$
```

Algorithm 3 Optimisation of k parameters in δk-NN.

```
Input: $\vec{k}$ {set of considered k values}
Input: $\vec{c}$ {set of classes}
Input: $\vec{\tau}_k$ {thresholds for each $k$}
Input: $\vec{q}_k$ {quality estimation for each $k$}
1: for all $c$ in $\vec{c}$ do
2:     $k(c) \leftarrow \{k \in \vec{k} : \mathrm{argmax}_k q_k(c)\}$
3:     $\tau_k(c) \leftarrow \{\tau \in \vec{\tau}_k, k \in \vec{k} : \mathrm{argmax}_k q_k(c)\}$
4: end for
```

The main difference between k-NN and δk-NN is how to exploit cross-validation information. k-NN adopts the k value with higher global quality, whereas δk-NN exploits the best k per class and the specific thresholds with that configuration.

6.4.4 Maximum Quality Boundary

The δk-NN performance relies on the accuracy of the quality estimations. This section focuses on the situation where a perfect quality estimation is achieved. This is, if the estimated best number of neighbours for each topic yield the best performance when applied to the test set. Such model is referred to as Oracle (in reference to being able to “predict” the future). A modification of our approach (δk-NN$_{Oracle}$) that uses the best k values based on the test set is introduced. Its definition implies that its macro-averaged quality always increases or remains the same with the number of neighbours, independently of the quality metric. The reason being that if the quality estimations are exact (as they are with an Oracle), the best number of neighbours per class is always selected. Therefore, for any two number of neighbours, the δk-NN$_{Oracle}$ macro-averaged quality of the highest one is always the best. This concept is explained below, where k and k' represent number of neighbours and $q(k,c)$ models the quality achieved for the class c by δk-NN$_{Oracle}$.

$$\forall k,k' : k \geq k' \rightarrow q(k,c) \geq q(k',c) \tag{6.57}$$

The monotonicity characteristic does not apply to δk-NN, nor to δk-NN$_{Oracle}$ with micro-averaged quality metrics. In these cases, the quality could decrease with larger numbers of neighbours.

6.4.5 Evaluation

Dynamic k-NN (δk-NN) is an adaptive version of k-NN where the number of neighbours are automatically optimised per category, based on a cross validation process (see Section 6.4). This section analyses the sub-optimality of traditional k-NN, where k is globally set, and it illustrates the improvement in quality by using δk-NN. Two different experiments are reported: the analysis on the optimality of different k values, and a quality comparison between k-NN and δk-NN.

6.4.5.1 Set-Up

`Reuters-21578` and `20-newsgroups` are used for the experiments. Documents are represented using `ltc` and χ^2 as feature selection measure, with 3000 and 5000 features for `Reuters-21578` and `20-newsgroups` respectively because it is the configuration that achieves better micro-averaged F_1 quality. The k values that have been tested are $\{5, 10, 15, ...200\}$. Experiments are carried out using our own implementation of both k-NN and δk-NN. A 5-fold validation is applied, maximising micro-averaged F_1, using FBRScut.1 as the thresholding strategy.

6.4.5.2 Results

Figures 6.6 and 6.7 illustrate the number of topics that share the same optimal k (i.e., it maximises the micro-average F_1). For instance, eleven different classes have an optimum k value of five in `Reuters-21578`. This data shows that there is a large variation on the value of the best k per class in both collections. However, the specific histogram distributions are very different. `20-newsgroups` shows that almost every class has a different optimum k, while `Reuters-21578` has three specific values of k (i.e., 5, 70 and 200) that are optimum for several categories. A preliminary analysis showed no correlation between the classes size and the optimum k for `Reuters-21578`. Nonetheless, deeper analysis on the histogram characteristics might provide useful information. The broad distribution of best k values per class shows that most of the classes have an optimum k different than the global. In addition, only two class for `Reuters-21578`, and none for `20-newsgroups` are optimised by using the best global k (60 and 55, respectively). These results support that a global optimisation is a sub-optimal solution for k-NN.

Tables 6.26-6.29 show the quality achieved by k-NN, δk-NN and $\delta k\text{-NN}_{Oracle}$ for both collections, applying micro and macro-averaged F_1. Significance testing is based on paired t-test. Improvements with respect to the best performing k-NN (with a maximum k number of neighbours) are shown. Figures 6.8 and 6.9 show similar information in a graphical format. δk-NN (with a maximum of 200 neighbours) outperforms the best k-NN in all the configurations, with the exception of `Reuters-21578` being evaluated with macro-averaged F_1, where both models achieved almost the same quality. However, none of them are statistically significant. The improvements in `20-newsgroups` are much larger than the ones in `Reuters-21578`.

6.4.5.3 Analysis

δk-NN consistently outperforms k-NN. The main reason why such improvement is much larger in `20-newsgroups` than in `Reuters-21578` is that the estimations of k for classes with very few examples, as it happens in `Reuters-21578`, can be inaccurate. This is supported by the fact that the improvements of $\delta k\text{-NN}_{Oracle}$ are always statistically significant. The estimation of the best k value for each category is a critical part of δk-NN and better estimators will increase its performance. δk-NN has another benefit, it optimises micro and macro-averaged quality

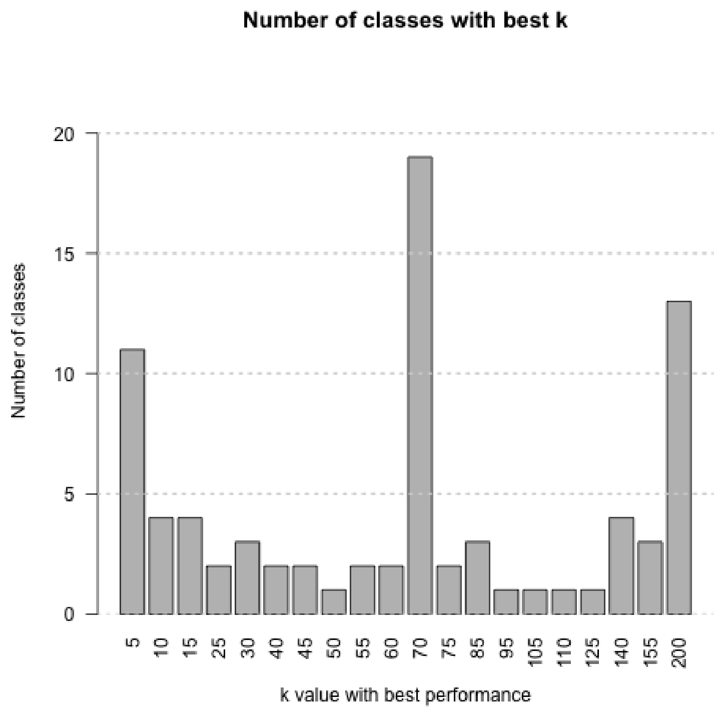
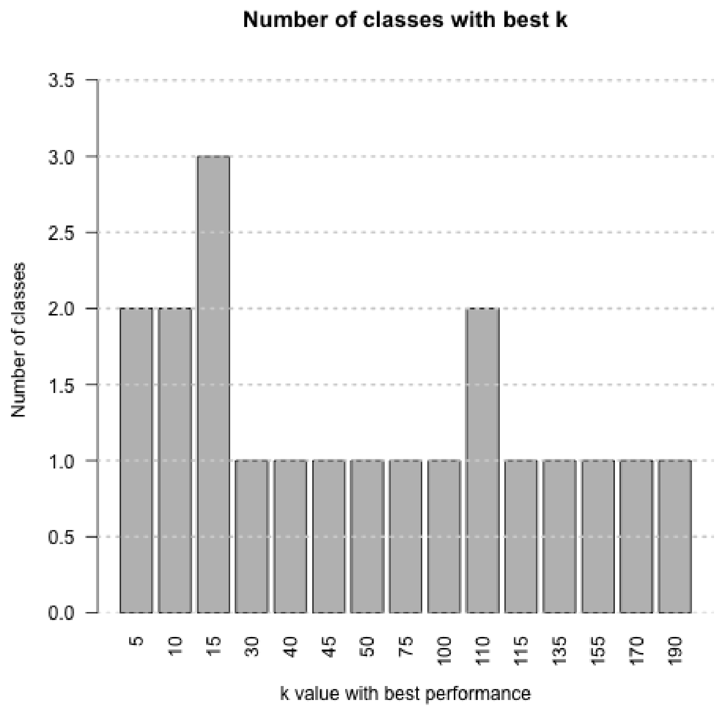


Figure 6.6: Histogram for the best k values per class for the `Reuters-21578` collection in terms of micro-averaged F_1, with 3000 features selected using χ^2.



(a) 20-newsgroups

Figure 6.7: Histogram for the best k values per class for the `20-newsgroups` collection in terms of micro-averaged F_1, with 5000 features selected using χ^2.

Table 6.26: `Reuters-21578` micro-averaged F_1 quality with different values of k using 3000 features selected using χ^2. Improvements (%) with respect to the best k-NN in brackets. Statistically significance improvements with $p < 0.01$, and $p < 0.02$ represented with † and ‡ respectively.

K	**kNN**	**DynamicKnn**	**DynamicKnnOracle**
15	84.73	84.98 (0.31)	85.57 †(1.00)
30	85.06	85.31 (0.29)	86.41 †(1.59)
50	85.57	85.76 (0.11)	87.20 †(1.78)
100	85.66	86.21 (0.47)	87.77 †(2.29)
150	85.48	86.12 (0.36)	87.96 †(2.51)
200	85.25	86.14 (0.38)	88.02 †(2.58)

Table 6.27: `Reuters-21578` macro-averaged F_1 quality with different values of k using 3000 features selected using χ^2. Improvements (%) with respect to the best k-NN in brackets. Statistically significance improvements with $p < 0.01$, and $p < 0.02$ represented with † and ‡ respectively.

K	**kNN**	**DynamicKnn**	**DynamicKnnOracle**
15	56.90	58.24 (2.35)	60.51 †(6.33)
30	59.62	58.77 (-1.43)	64.56 †(8.29)
50	59.49	60.29 (-0.16)	66.03 †(9.34)
100	60.06	60.97 (0.96)	67.68 †(12.08)
150	61.01	60.92 (-0.97)	69.17 †(12.45)
200	60.99	61.23 (-0.46)	69.41 †(12.84)

Table 6.28: `20-newsgroups` micro-averaged F_1 quality with different values of k using 5000 features selected using χ^2. Improvements (%) with respect to the best k-NN in brackets. Statistically significance improvements with $p < 0.01$, and $p < 0.02$ represented with † and ‡ respectively.

K	**kNN**	**DynamicKnn**	**DynamicKnnOracle**
15	76.51	76.20 (-0.61)	77.26 ‡(0.77)
30	76.88	77.26 (0.34)	78.05 †(1.37)
50	77.01	77.57 (0.74)	78.78 †(2.30)
100	76.90	77.81 (0.85)	79.22 †(2.68)
150	76.15	77.79 (0.83)	79.45 †(2.98)
200	75.87	77.85 (0.91)	79.46 †(3.00)

Table 6.29: `20-newsgroups` macro-averaged F_1 quality with different values of k using 5000 features selected using χ^2. Improvements (%) with respect to the best k-NN in brackets. Statistically significance improvements with $p < 0.01$, and $p < 0.02$ represented with † and ‡ respectively.

K	**kNN**	**DynamicKnn**	**DynamicKnnOracle**
15	76.21	75.86 (-0.48)	76.93 †(0.92)
30	76.37	76.79 (0.39)	77.66 †(1.52)
50	76.40	77.06 (0.74)	78.35 †(2.42)
100	76.34	77.27 (0.92)	78.76 †(2.86)
150	75.72	77.25 (0.90)	78.98 †(3.16)
200	75.47	77.30 (0.96)	78.99 †(3.17)

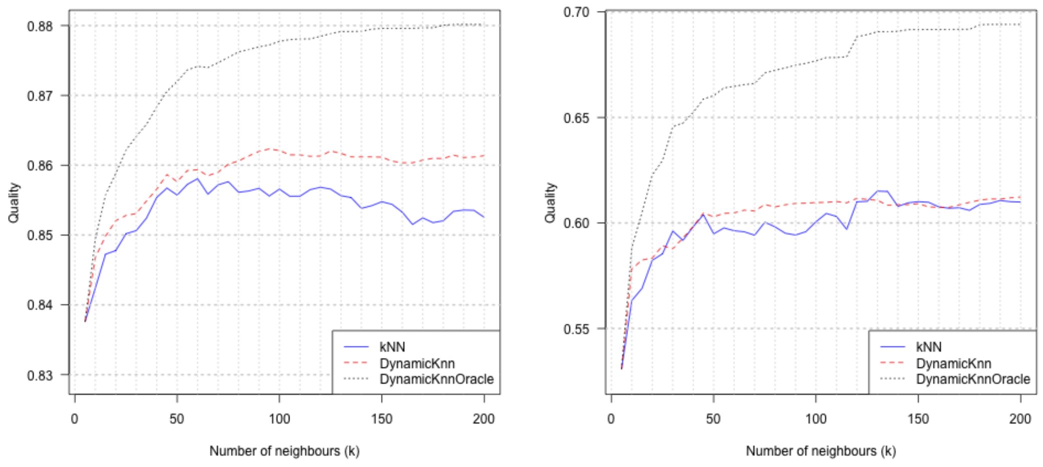


Figure 6.8: Micro (left) and Macro (right)-average F_1 quality for `Reuters-21578` for different values of k, using 3000 features based on χ^2.

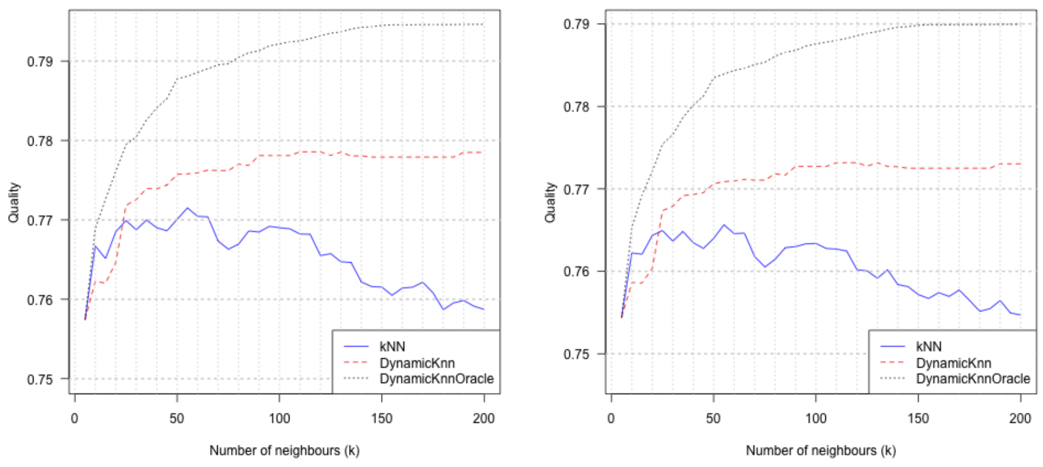


Figure 6.9: Micro (left) and Macro (right)-average F_1 quality for `20-newsgroups` for different values of k, using 5000 features based on χ^2.

simultaneously with the highest value of k, whereas k-NN requires to tune them independently.

On the other hand, for `Reuters-21578`, k-NN maximises micro-averaged F_1 with 55 neighbours, while the macro-averaged is the best when k is 135.

6.4.6 Discussion

The global tuning of the number of neighbours for k-NN leads to sub-optimal results. Adaptive methods should be apply in order to increase the potential of k-NN. This research presents a category-based k optimisation (δk-NN) that is based on quality estimation, and the tuning process that is applied with traditional k-NN.

Experiments show that δk-NN should be preferred over k-NN, as it outperforms traditional k-NN in almost all cases, without having any drawback. Furthermore, an Oracle variation of the algorithm illustrates its potential, if perfect quality estimations were available. δk-NN$_{Oracle}$ statistically significantly outperforms all the other models with up to 3% micro and 12% macro-averaged increase with respect to the best k-NN. Results also suggest that δk-NN is more suited for non-skewed collections. One relevant finding is that even uniformly distributed collections show very diverse optimum k values per category. This result collides with one of the main implicit assumptions in related research, which is that the number of neighbours should be optimised based on the size of a given category [Baoli et al., 2004].

6.5 Summary

Knowledge exploitation have been largely investigated for augmented representation and score augmentation, while other steps of the classification process have been overlooked. We have proposed novel tasks and methods for some of those steps. These examples underline how knowledge should be more present in the general classification flow.

Chapter 7

Conclusions and Research Outlook

7.1 Summary

This thesis has investigated the seamless modelling of text classification using a descriptive approach to improve flexibility and maintainability. We have shown how this approach leads to a high-level abstraction that provides a flexible and adaptable environment for prototyping and knowledge exploitation with minimum engineering effort. However, this work has also shown some of the limitations of the approach in terms of expressiveness and scalability. We have also illustrated how to model and intelligently combine textual and relational data in heterogeneous networks. Furthermore, this research have also shown the automatic translation from descriptive modelling to a mathematical formulation for verification and validation purposes.

The second goal of this research is to investigate how to use knowledge and evidence exploitation in steps of the classification process where they have not traditionally been applied. Firstly, the task of Semi-Automatic Text Classification has been proposed to intelligently combine manual and automatic classification. In addition, different approaches to address this task have also being proposed. Secondly, Document Performance Prediction has been introduced to estimate the document performance for a classification system. Thirdly, the possibility of exploiting category dependencies to improve the classification evaluation mechanisms has been investigated. Finally, a strategy to locally optimise the number of neighbours for *k*-NN has been proposed.

7.2 Conclusions

The main research questions from which this thesis originated are asked in Section 1.2. The conclusions of this research are outlined based on the answers to such questions.

To what degree is the expressiveness of descriptive approaches, represented by probabilistic Datalog, enough to model and customise traditional classifiers?

This thesis shows the modelling of different variations of Naive-Bayes and *k*-NN classification families. This illustrates that they can be expressed using a descriptive approach. The quality evaluation also provides empirical evidence that they achieve, as expected, the same quality as other approaches. The compact high-level definitions and the use of a predicate dictionary lead to a flexible framework where experts can model specific strategies with minimum engineering effort. As a result, there is virtually total customisation for any task. The experiments also illustrate that collections can be indexed in a reasonable amount of time. For instance, `20-newsgroups` requires 33 minutes to index all the information required for a Naive-Bayes classifier, while it requires 160 ms per document in classification time. This results support the claim that descriptive approaches can be applied in prototypical environments. Furthermore, the efficiency of this approach relies heavily on engine implementation and optimisations (which are out of the scope of this thesis). Therefore, these experiments should be considered as a maximum boundary of the required time. Any improvement made to the underlying system will potentially increase its efficiency.

To what degree can models expressed in a descriptive approach be automatically translated to a mathematical formulation to observe and verify its semantics?

The nature of probabilistic Datalog implies that a program can be automatically translated to a mathematical definition if each operator can be independently translated. Such definition must specify two aspects: what tuples to keep or generate, and what probability should be computed for the new tuples. We have shown the translation steps for the main operators of the language.

Can a formal framework generalise and integrate the main knowledge exploitation techniques for data augmentation and score modification in text classification?

We have shown a conceptual framework in which several techniques for knowledge exploitation can be included. Although the specific algorithms use different techniques, all of them rely on some weight for specific entities and/or the relationships between them as their core mechanism. This is considered in our approach and, therefore, any method that follows the same principle can be included within the framework.

To what degree descriptive approaches can seamlessly integrate textual and relational classifiers with heterogeneous data?

Combining textual and relational classification is a challenge due to the complexity of combining diverse data. Probabilistic Datalog has proven to be able to model and combine, using a unique framework, textual and relational classifiers within a heterogeneous network. As a result, the prototyping capabilities in such complex environments is increased.

Is a descriptive approach abstract enough to apply flexible task integration and adaptation?

The combination of different information retrieval tasks is becoming more common in order to improve specific models or to address complex information needs. However, such integration involves a complex engineering process using the current systems. Abstraction based on Logics seamless allows to apply multi-task adaptation.

Given a set of documents to be classified and a limited amount of human resources. What quality can be achieved if those resources are optimised to focus only on the documents that the automatic system is more likely to misclassify?, and, are the category thresholds and classification scores useful for this task?

This thesis has proposed the novel task of Semi-Automatic Text Classification to maximise the overall classification quality in environments with limited human resources. The scores and category thresholds obtained in a classification process are a very helpful source of information that can be exploited in different forms. This work has proposed a family of metrics that uses this information to rank documents. The results show that micro-averaged F_1 values higher than 0.95 can be achieved with only 30-50% of documents being inspected. The main implication of this is that a large cost reduction can be applied in those cases and that the resources (i.e., human annotators) can be optimised by relying on the certainty ranking.

With what confidence can a set of documents be ranked according to their expected classification quality?, and, are the category thresholds and classification scores useful to solve this task?

This thesis has introduced the Document Performance Prediction task in the context of classification. Results using approaches from the query performance prediction field indicate that the pre-classification performance predictors correlate relatively well with the ranking of documents, with Spearman coefficient values up to 0.25-0.47 depending on the collection. The best performers are the `idf` based predictors, similarly to what has been reported for query performance prediction. In addition, a Threshold based Document Performance prediction framework that exploits the classification scores and the category thresholds has been proposed. The results show that this approach significantly outperform all the pre-classification methods with correlation values of up to 0.7.

Given a set of classifiers and their quality evaluation, will their relative quality or their ranking change using an evaluation metric that includes degrees of misclassification?

The importance of dependencies in text classification is a known fact that has traditionally been neglected in the evaluation process. Some mistakes are worse than others from a human perspective, and the final goal of an evaluation process in classification is to measure the user satisfaction. Therefore, it is a factor that should be considered. For this reason, we have presented two different methods that incorporate this information into the evaluation process (see Section 6.3), and compare them with traditional evaluation metrics.

This research shows that the quality ranking varies in one case, using these approaches and three classifiers. However, their relative improvement change significantly in most cases. This supports the claim that different classifiers tend to misclassified "worse" than others. In particular, the results suggest that *k*-NN is the classifier that tends to miss "closer" to the real categories. The fact that there are almost no changes in the ranking of classifiers can be due to the large difference in performances.

How can the number of neighbours in a k-NN classifier be automatically optimised per category without dividing it into multiple binary classifiers, and, what quality improvement will be achieved compared to traditional k-NN?

The number of neighbours can be dynamically optimised per category, based on the estimated quality values that are obtained in the tuning phase of any traditional k-NN classifier. A model that applies this approach, referred to as dynamic k-NN (δk-NN) was presented in Section 6.4. δk-NN only considers examples from the top $k(c)$ most similar examples with respect to the category c, using the number of neighbours that produced the best estimated quality value in the tuning phase.

The global optimisation of k leads to sub-optimal results. Results show that δk-NN constantly outperforms the traditional version, without having any disadvantage. These improvements are not statistically significant. Nonetheless, if the quality estimation would be done perfectly, the same approach will obtain statistically significant improvements of up to 12% and 3% for macro-average F_1, using `Reuters-21578` and `20-newsgroups` respectively. This shows the potential of the approach and the need for better quality estimations.

7.3 Discussion, Limitations and Research Outlook

The descriptive modelling of classification provides a prototypical environment that can easily incorporate new types of data and tasks for unforeseen information needs with minimum engineering effort. This also includes the possibility of exploit and combine textual and relational information. The main limitation is the current impossibility of model iteration within the framework. As a result, some models are not feasible to be represented without externalising some of their steps (e.g., SVM). In addition, the modelling of some strategies such as Laplace correction result on counterintuitive and somehow confusing representations. One of the main options to address these challenges is to allow internal functions within the language to support these operations. Also, although the system can be applied to reasonably sized collections, its scalability and efficiency is worse than other systems. Further research should be done to address both challenges in the future.

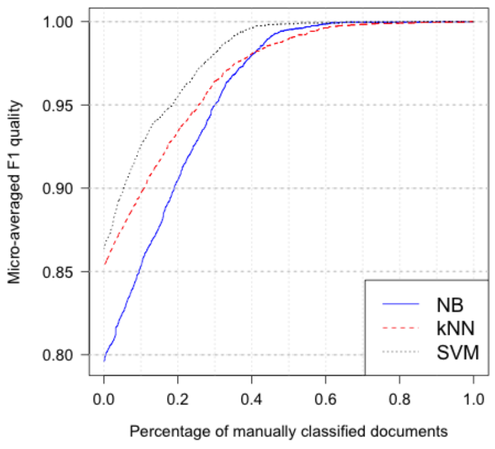
On the other hand, this thesis has proven that knowledge can and should be applied to all the different steps in the classification process. There are multiple possibilities to extend this research in the future. In particular, the two novel tasks of Semi-Automatic Text Classification (SATC) and Document Performance Prediction (DPP) can be seen as new research lines for the community. This thesis has shown the cost reduction that can be achieved, without a loss in quality, by focusing the human resources on the documents most likely to be misclassified by a classification system. However, the amount of work done in this field is very limited and other algorithms to measure the certainty of each document should be proposed. Also, SATC should focus on very low ratios of documents being manually classified if bigger collections are used due to the scalability challenges. For DPP, although its potential has been shown in the correlation experiments, further research should be done to test its real applicability. For instance, using it to dynamically select the best classifier for a given document. Finally, the discussion of near-misses evaluation could be used to revisit how quality is evaluated in classification.

This research has contributed to the long goal of a high-level abstraction system for information retrieval that can integrate knowledge and different tasks to solve complex information needs represented by "knowledge engineers". In addition, we have shown the potential of some steps of the classification process that research has traditionally neglected.

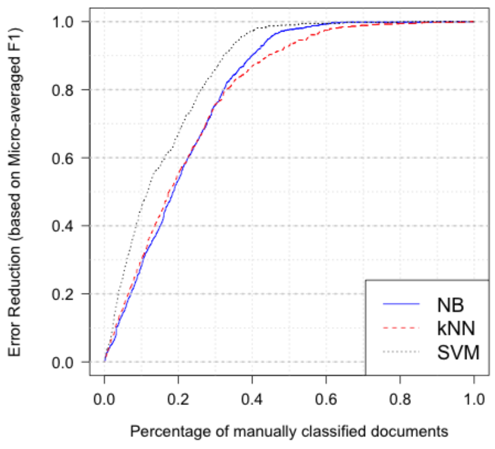
Appendix A

Document Difficulty Framework quality graphs for SATC

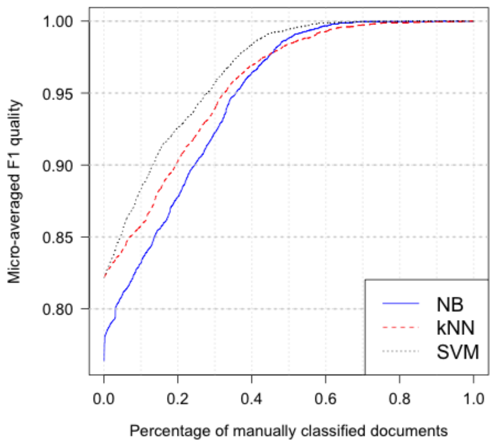
This section shows the classification quality graphs for the different models presented in the Document Difficulty Framework (see Section 6.1.4) with different percentages of the collection being manually classified.



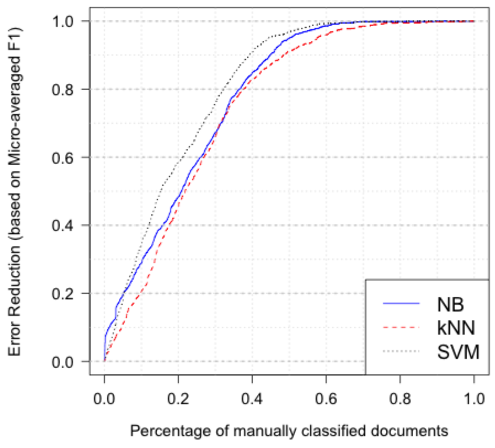
(a) Reuters-21578 SAM



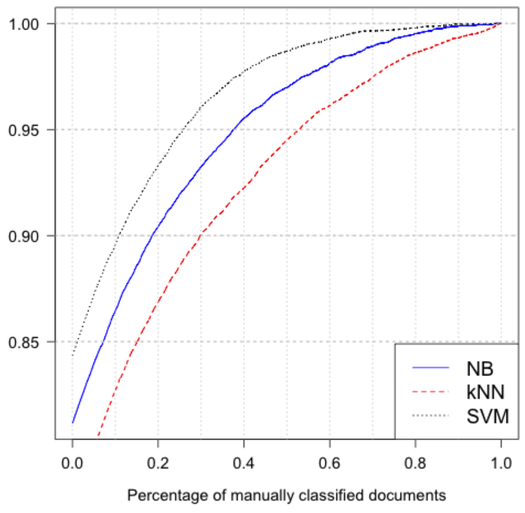
(b) Reuters-21578 SAM



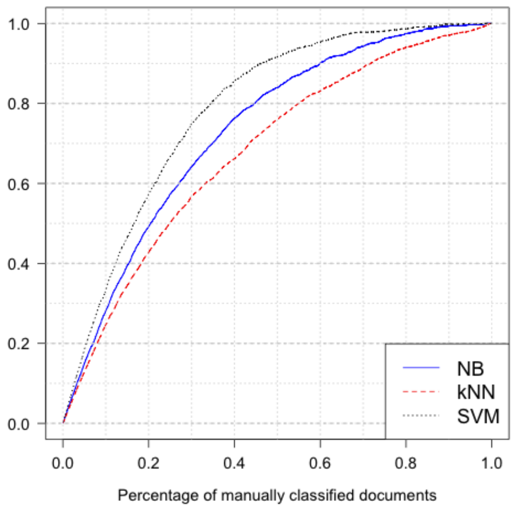
(c) Reuters-21578-115 SAM



(d) Reuters-21578-115 SAM

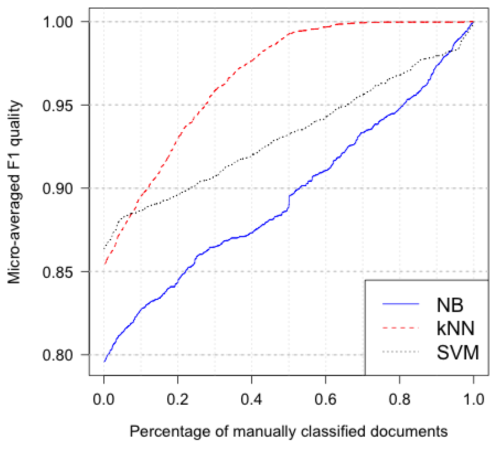


(e) 20-newsgroups SAM

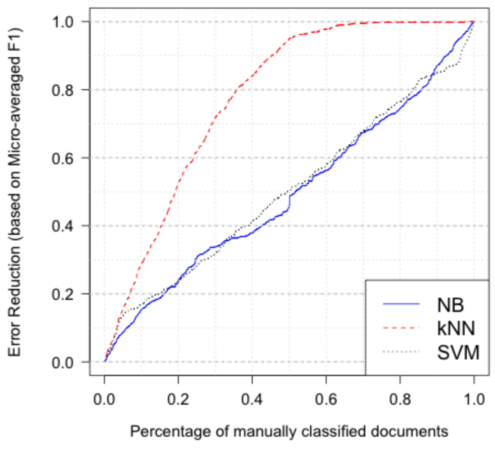


(f) 20-newsgroups SAM

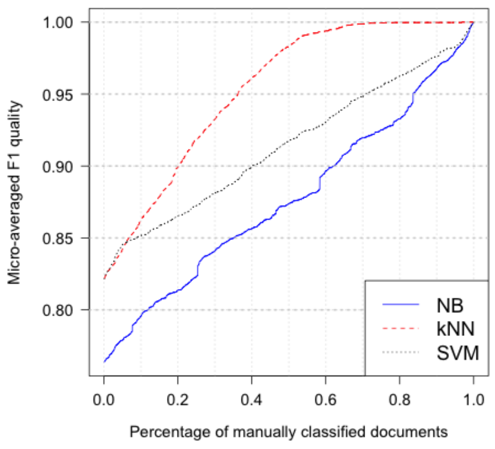
Figure A.1: Micro-averaged F_1 (left) and ER (right) evaluation for different ratios of manually classified documents using SAM for `Reuters-21578`, `Reuters-21578-115`, and `20-newsgroups`.



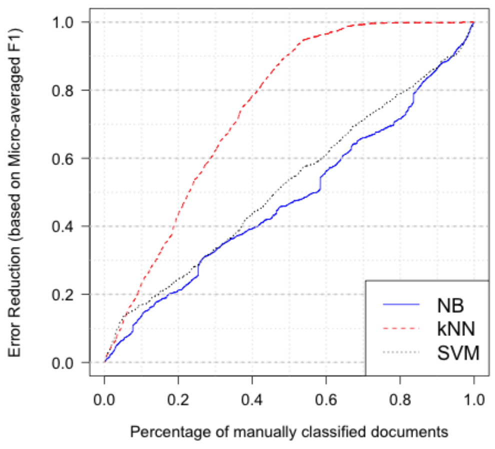
(a) Reuters-21578 SAA



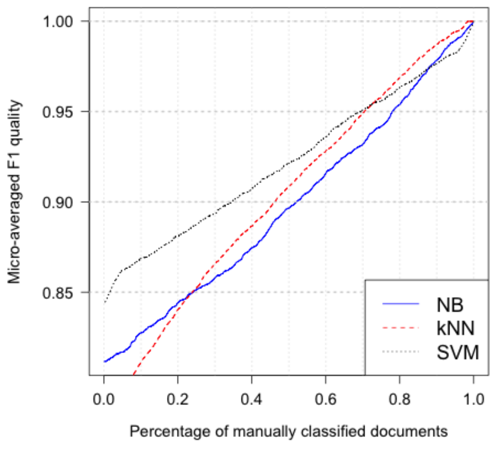
(b) Reuters-21578 SAA



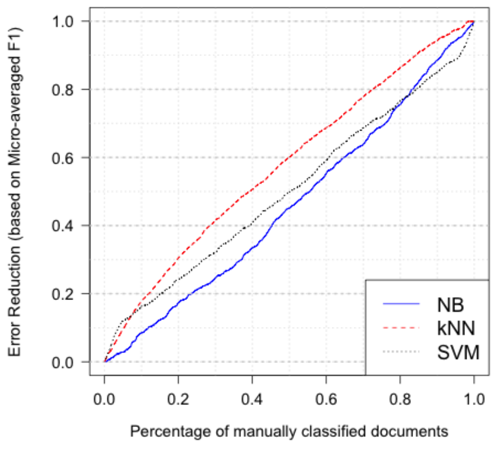
(c) Reuters-21578-115 SAA



(d) Reuters-21578-115 SAA

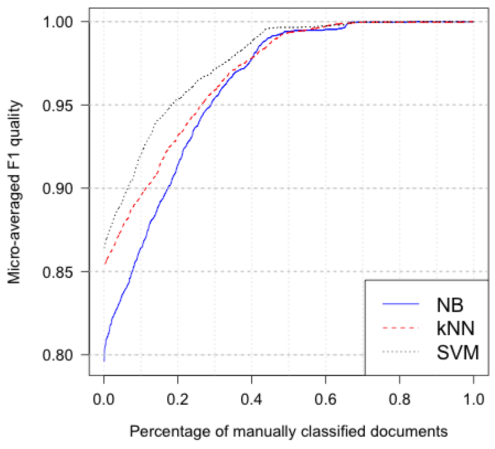


(e) 20-newsgroups SAA

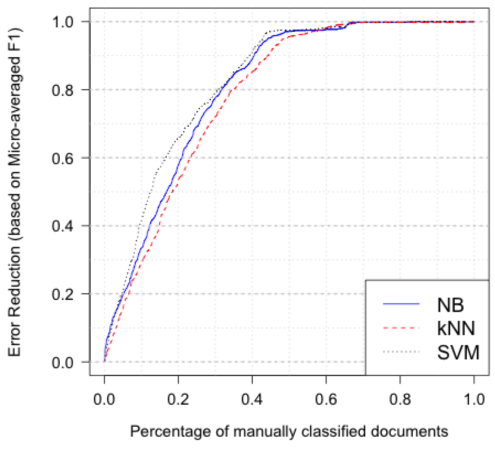


(f) 20-newsgroups SAA

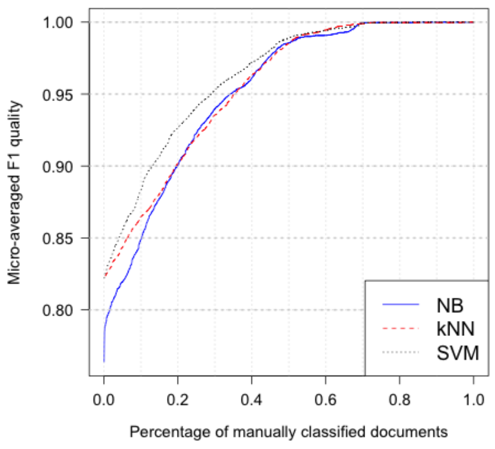
Figure A.2: Micro-averaged F_1 (left) and ER (right) evaluation for different ratios of manually classified documents using SAA for Reuters-21578, Reuters-21578-115, and 20-newsgroups.



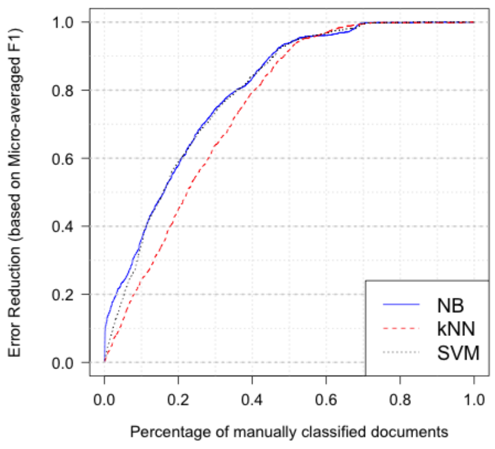
(a) `Reuters-21578` SAW



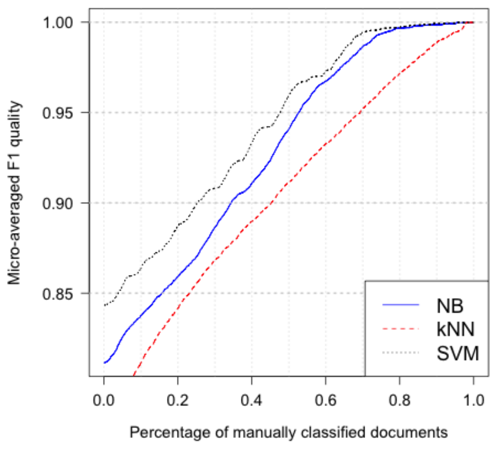
(b) `Reuters-21578` SAW



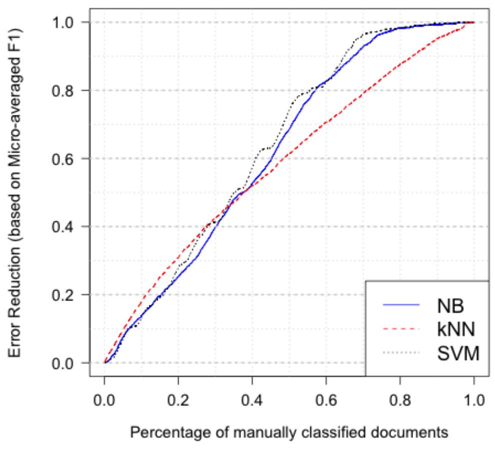
(c) `Reuters-21578-115` SAW



(d) `Reuters-21578-115` SAW

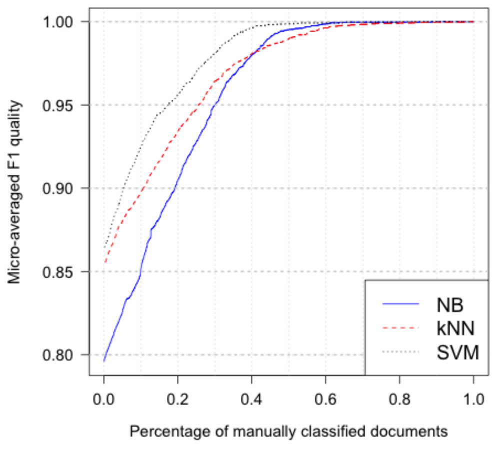


(e) `20-newsgroups` SAW

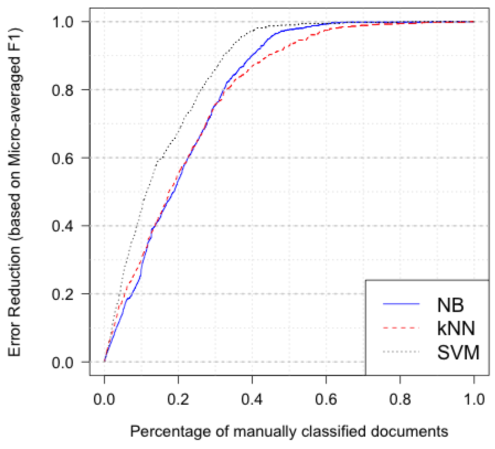


(f) `20-newsgroups` SAW

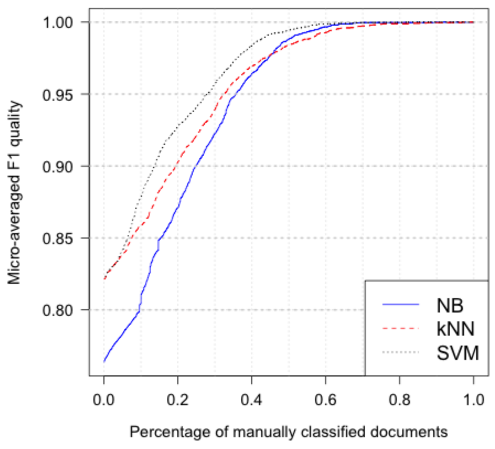
Figure A.3: Micro-averaged F_1 (left) and ER (right) evaluation for different ratios of manually classified documents using SAW for `Reuters-21578`, `Reuters-21578-115`, and `20-newsgroups`.



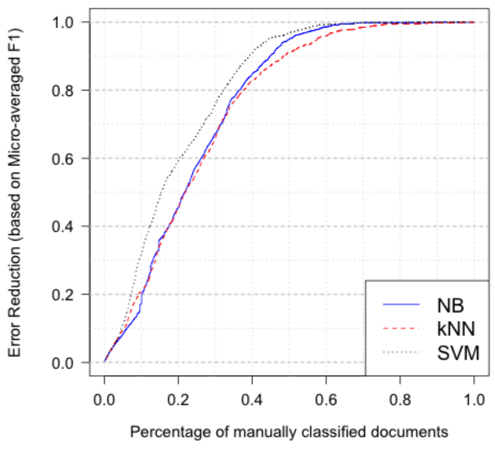
(a) `Reuters-21578` SPM



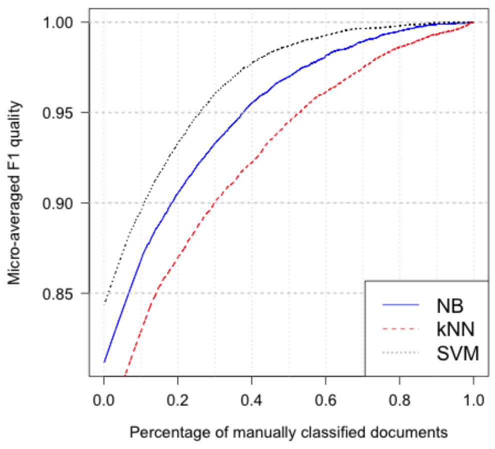
(b) `Reuters-21578` SPM



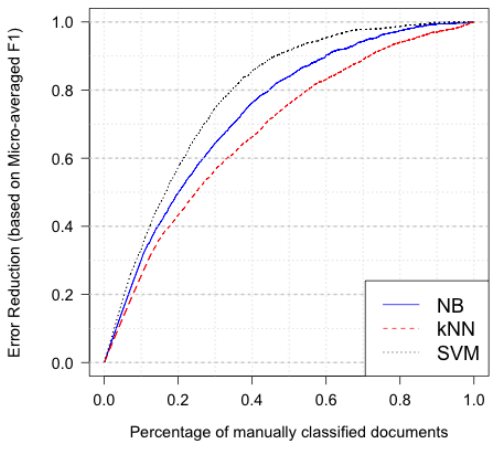
(c) `Reuters-21578-115` SPM



(d) `Reuters-21578-115` SPM

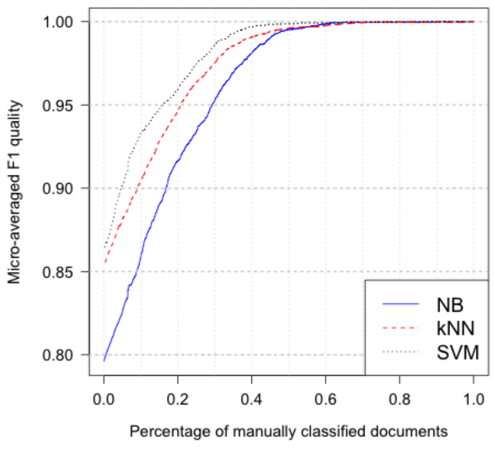


(e) `20-newsgroups` SPM

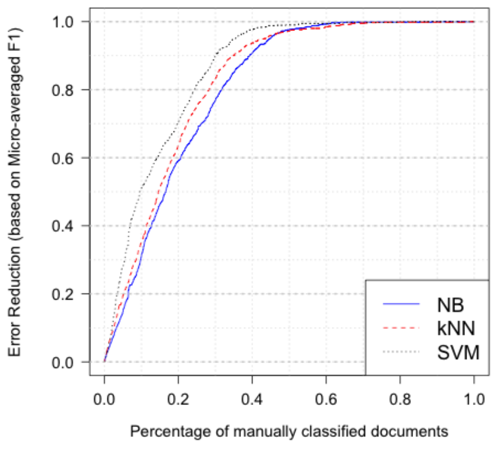


(f) `20-newsgroups` SPM

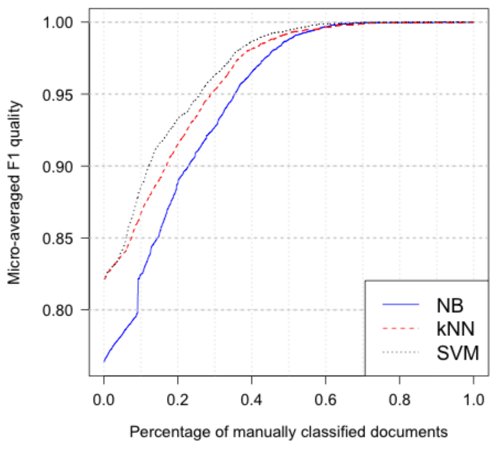
Figure A.4: Micro-averaged F_1 (left) and ER (right) evaluation for different ratios of manually classified documents using SPM for `Reuters-21578`, `Reuters-21578-115`, and `20-newsgroups`.



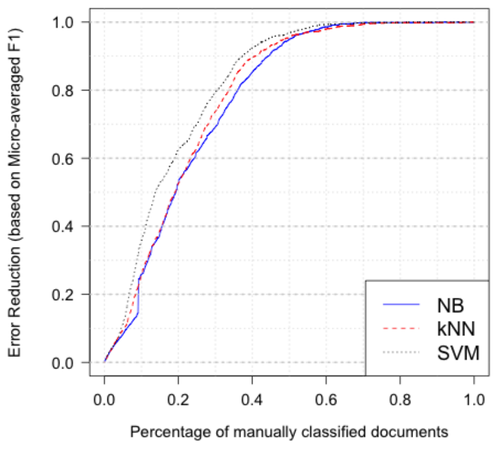
(a) `Reuters-21578` SPA



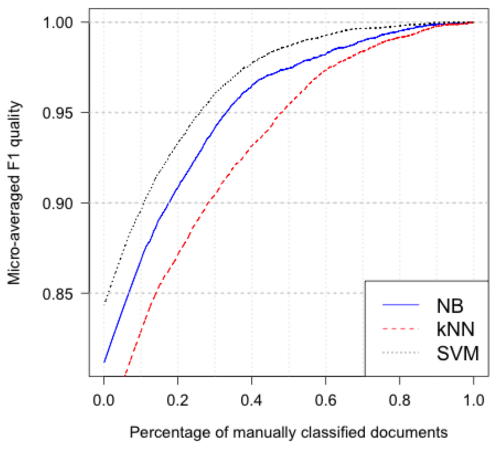
(b) `Reuters-21578` SPA



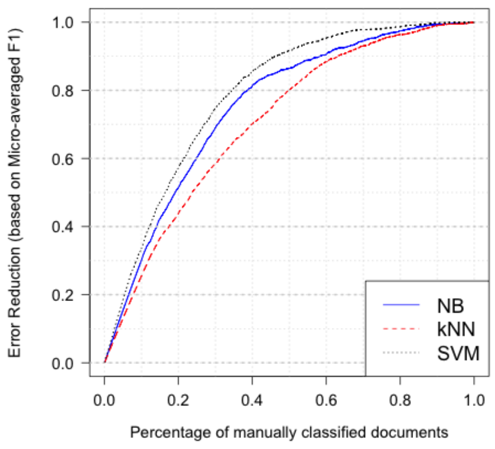
(c) `Reuters-21578-115` SPA



(d) `Reuters-21578-115` SPA

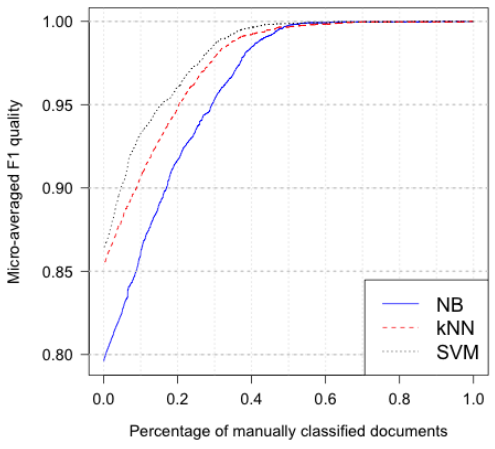


(e) `20-newsgroups` SPA

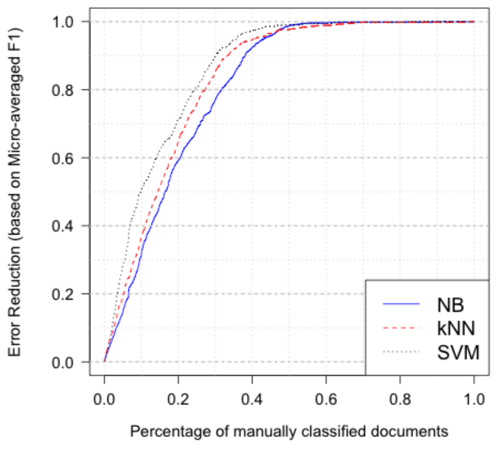


(f) `20-newsgroups` SPA

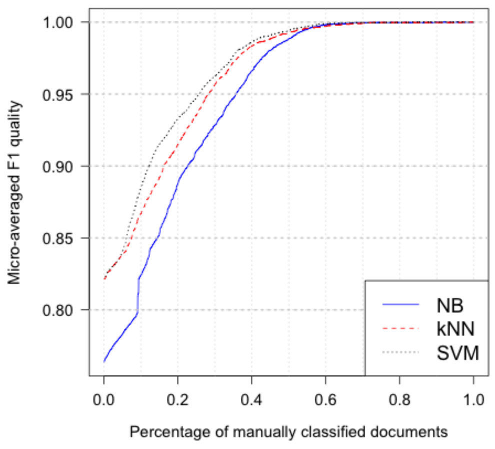
Figure A.5: Micro-averaged F_1 (left) and ER (right) evaluation for different ratios of manually classified documents using SPA for `Reuters-21578`, `Reuters-21578-115`, and `20-newsgroups`.



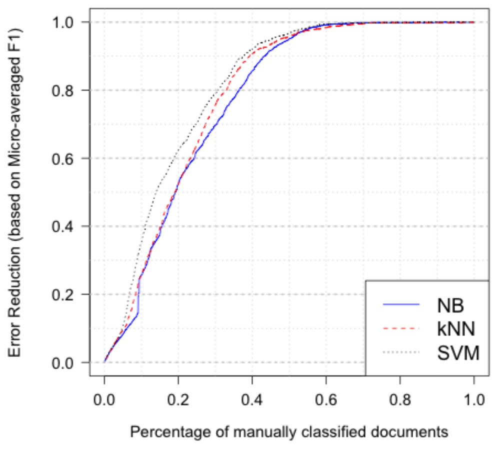
(a) `Reuters-21578` SPW



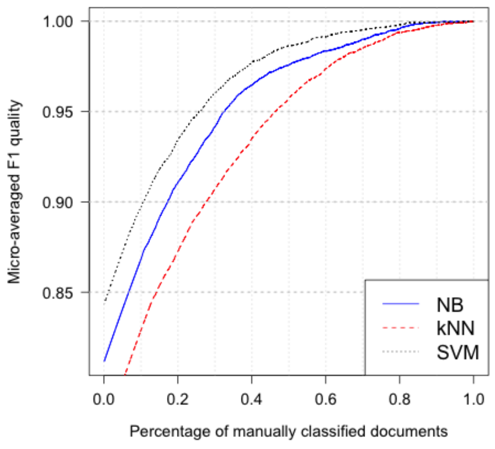
(b) `Reuters-21578` SPW



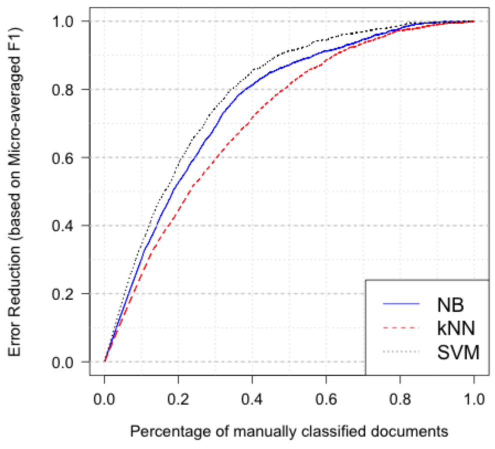
(c) `Reuters-21578-115` SPW



(d) `Reuters-21578-115` SPW

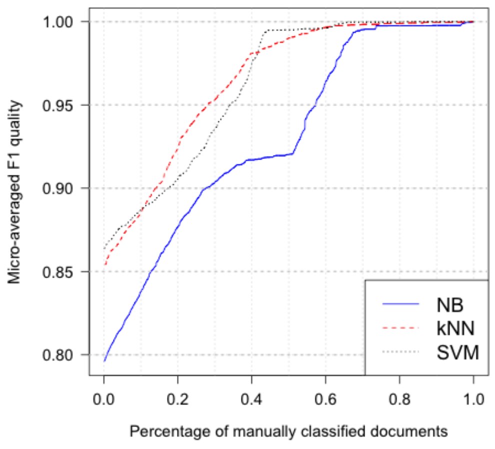


(e) `20-newsgroups` SPW

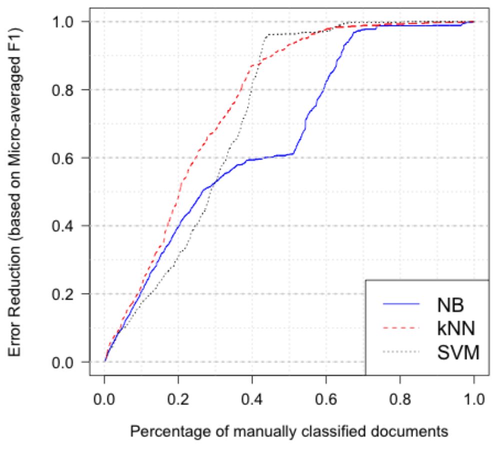


(f) `20-newsgroups` SPW

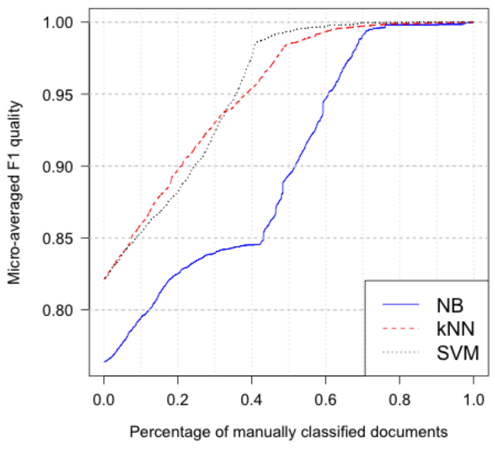
Figure A.6: Micro-averaged F_1 (left) and ER (right) evaluation for different ratios of manually classified documents using SPW for `Reuters-21578`, `Reuters-21578-115`, and `20-newsgroups`.



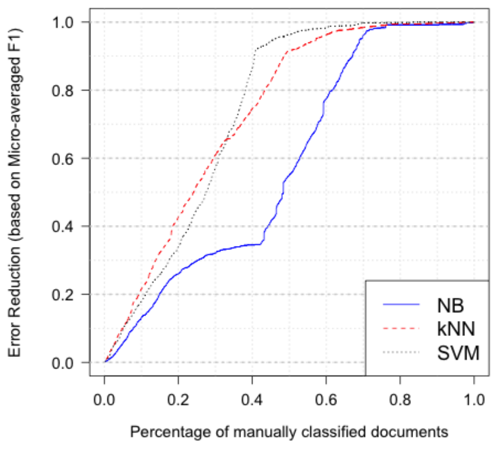
(a) Reuters-21578 AAM



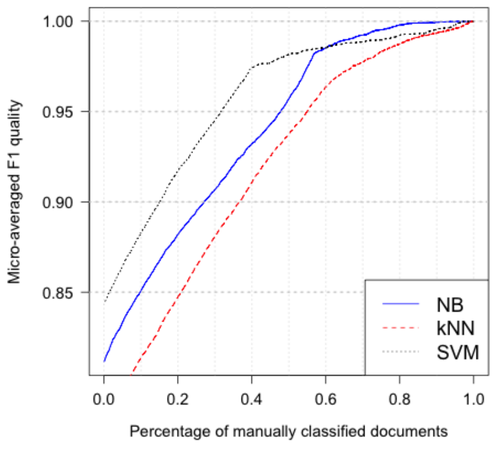
(b) Reuters-21578 AAM



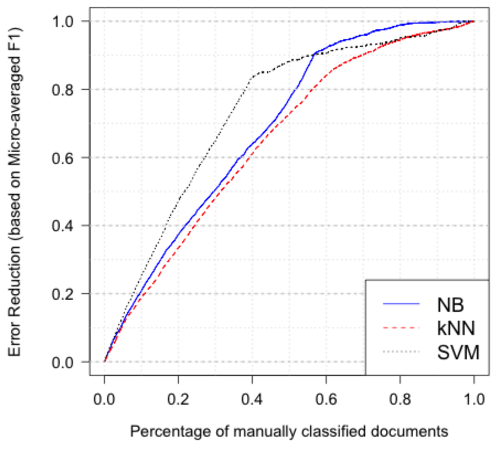
(c) Reuters-21578-115 AAM



(d) Reuters-21578-115 AAM

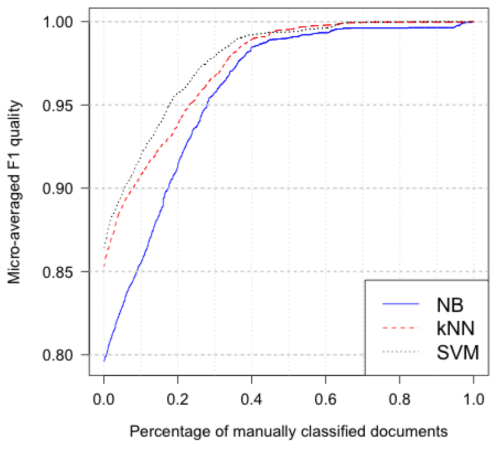


(e) 20-newsgroups AAM

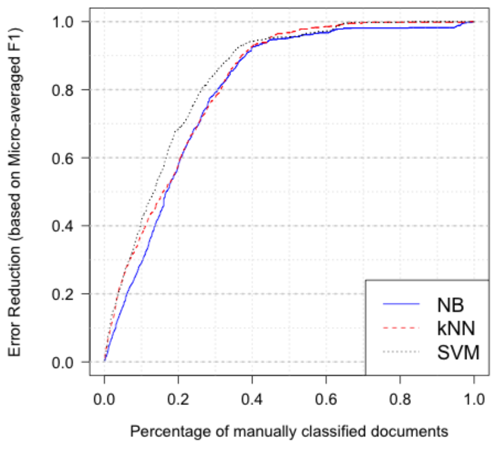


(f) 20-newsgroups AAM

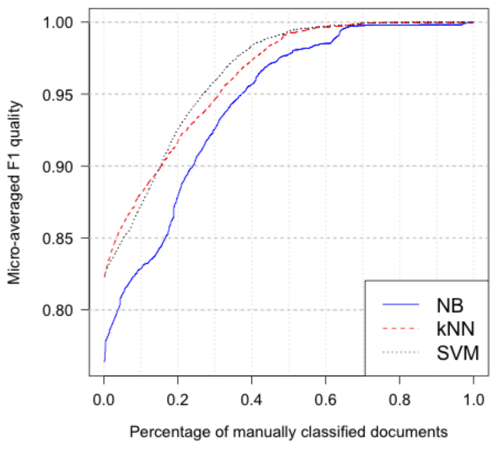
Figure A.7: Micro-averaged F_1 (left) and ER (right) evaluation for different ratios of manually classified documents using AAM for Reuters-21578, Reuters-21578-115, and 20-newsgroups.



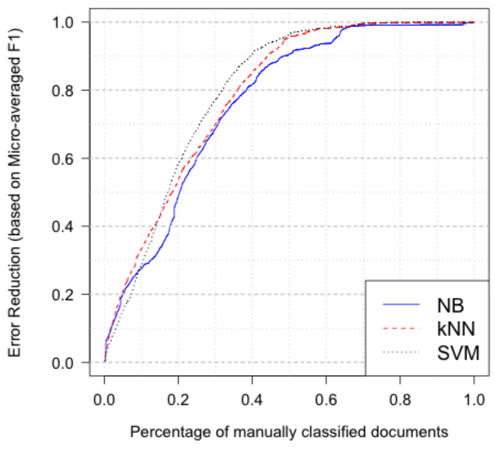
(a) `Reuters-21578` AAA



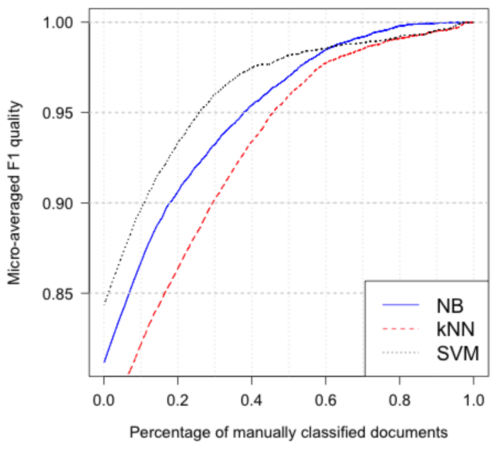
(b) `Reuters-21578` AAA



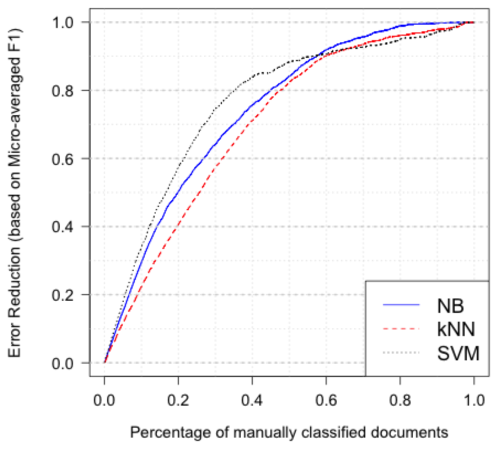
(c) `Reuters-21578-115` AAA



(d) `Reuters-21578-115` AAA

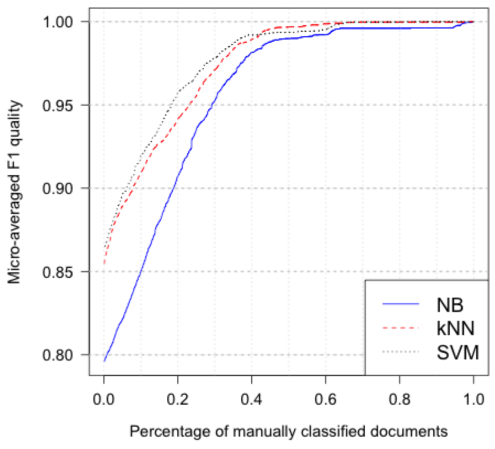


(e) `20-newsgroups` AAA

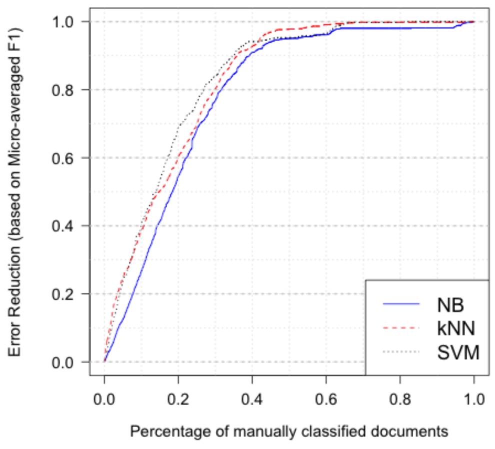


(f) `20-newsgroups` AAA

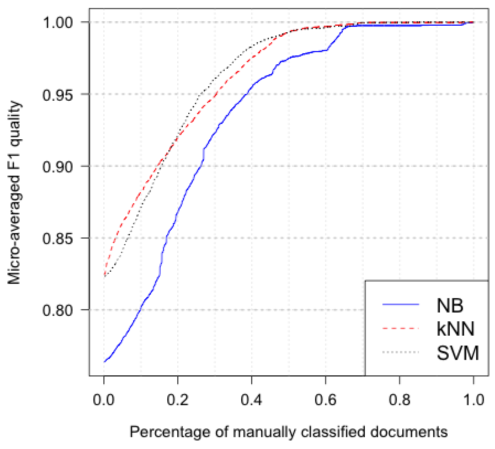
Figure A.8: Micro-averaged F_1 (left) and ER (right) evaluation for different ratios of manually classified documents using AAA for `Reuters-21578`, `Reuters-21578-115`, and `20-newsgroups`.



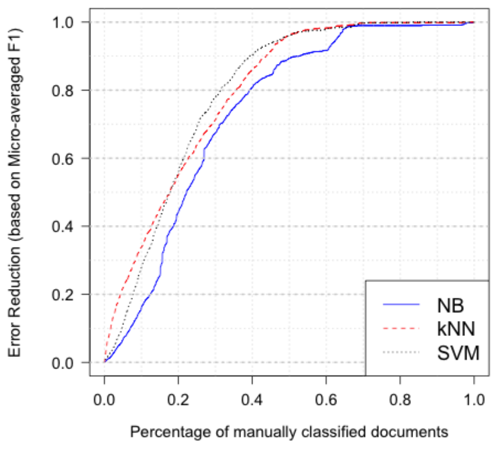
(a) `Reuters-21578` AAW



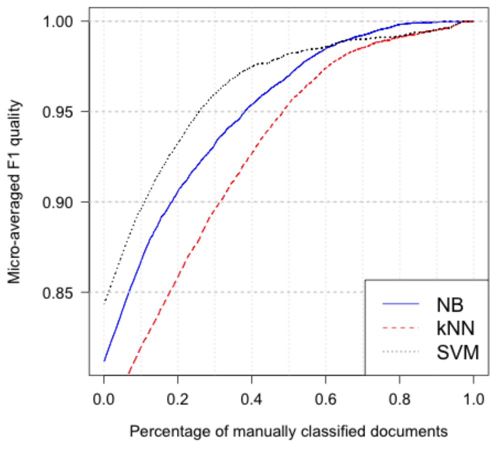
(b) `Reuters-21578` AAW



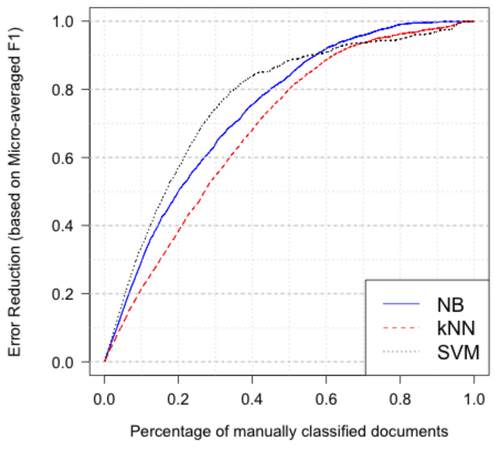
(c) `Reuters-21578-115` AAW



(d) `Reuters-21578-115` AAW

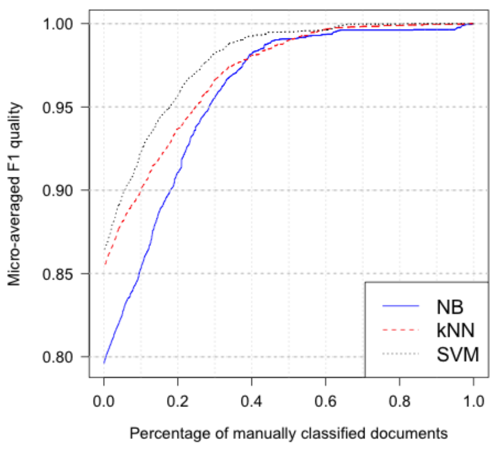


(e) `20-newsgroups` AAW

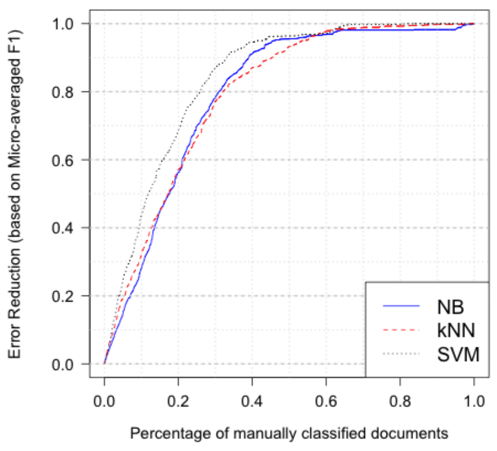


(f) `20-newsgroups` AAW

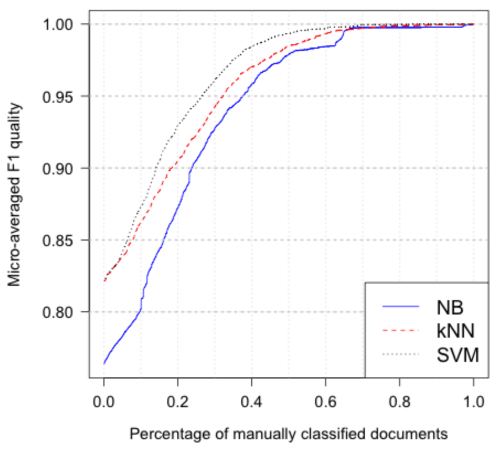
Figure A.9: Micro-averaged F_1 (left) and ER (right) evaluation for different ratios of manually classified documents using AAW for `Reuters-21578`, `Reuters-21578-115`, and `20-newsgroups`.



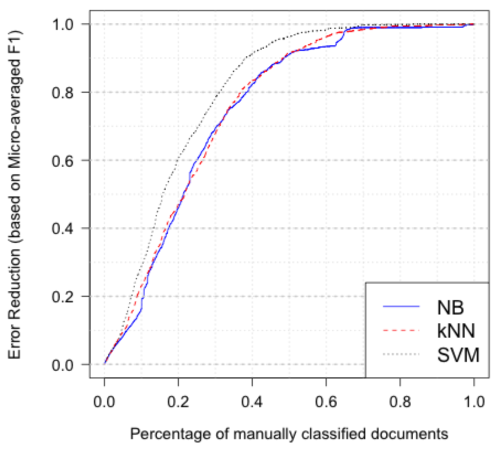
(a) `Reuters-21578` APM



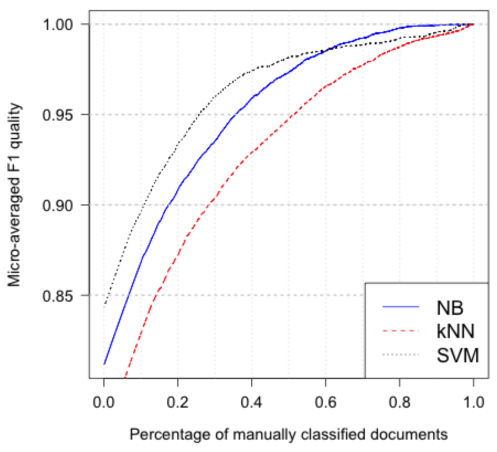
(b) `Reuters-21578` APM



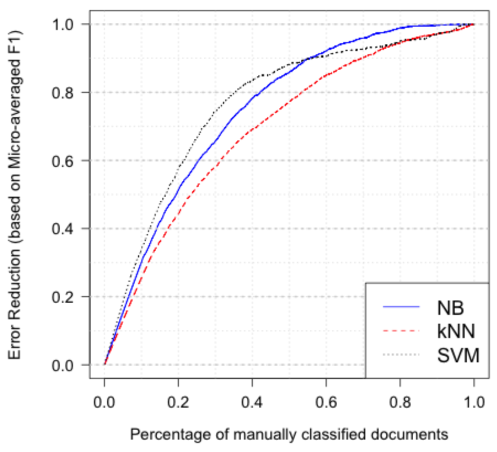
(c) `Reuters-21578-115` APM



(d) `Reuters-21578-115` APM

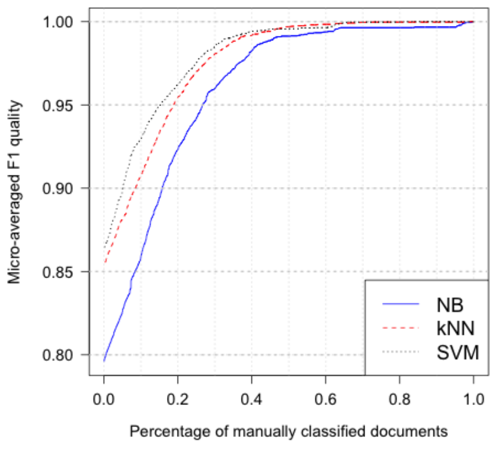


(e) `20-newsgroups` APM

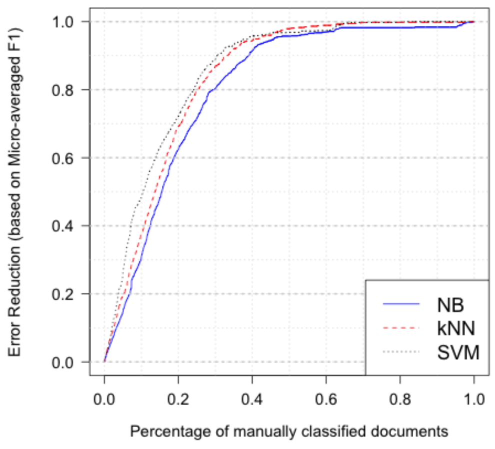


(f) `20-newsgroups` APM

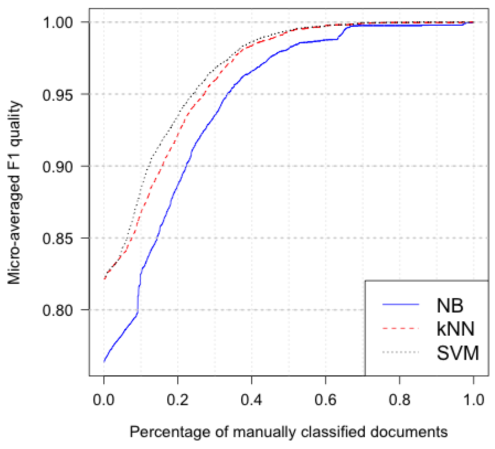
Figure A.10: Micro-averaged F_1 (left) and ER (right) evaluation for different ratios of manually classified documents using APM for `Reuters-21578`, `Reuters-21578-115`, and `20-newsgroups`.



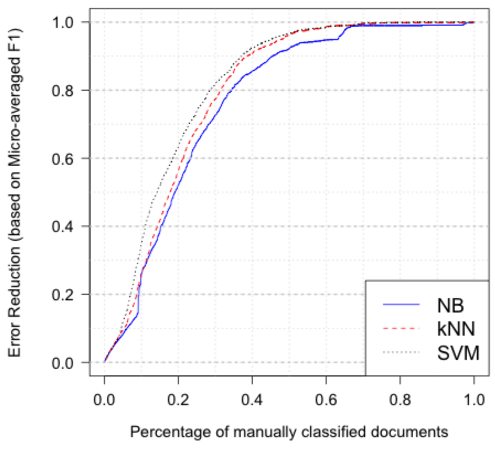
(a) `Reuters-21578` APA



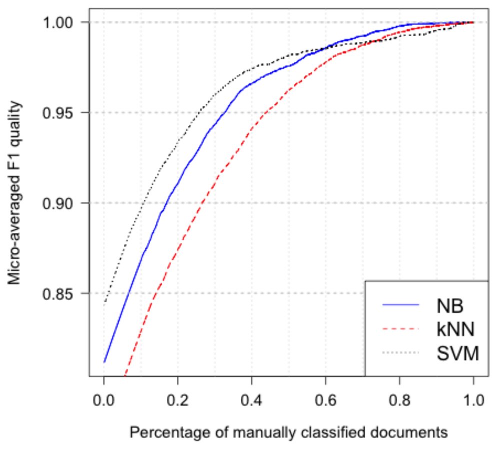
(b) `Reuters-21578` APA



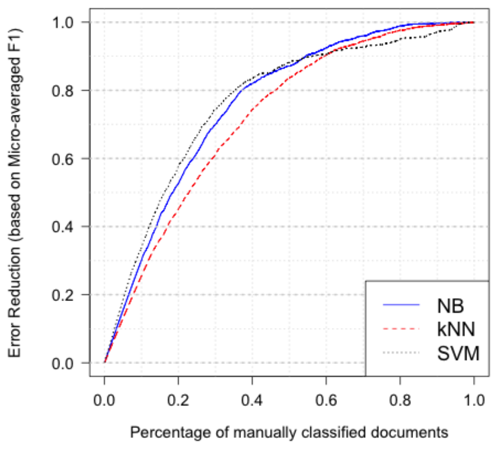
(c) `Reuters-21578-115` APA



(d) `Reuters-21578-115` APA

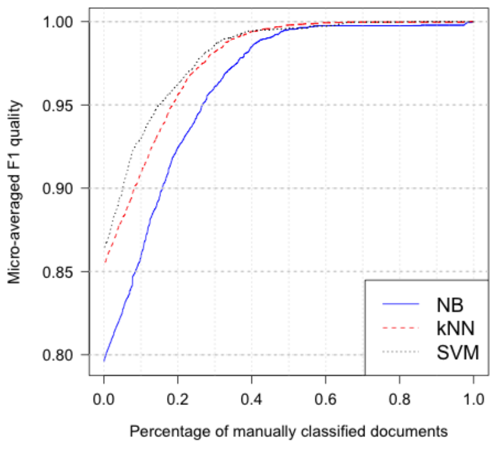


(e) `20-newsgroups` APA

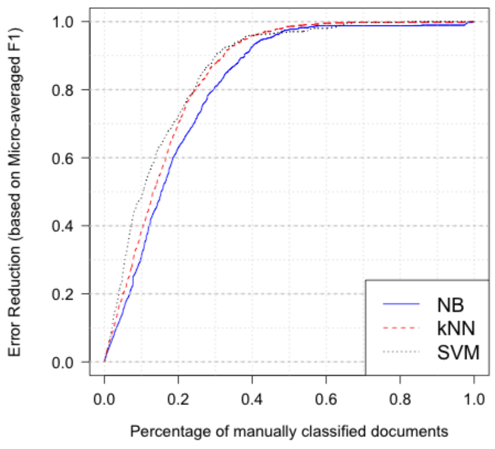


(f) `20-newsgroups` APA

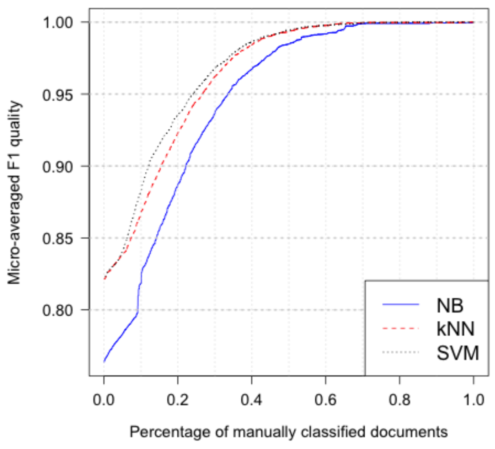
Figure A.11: Micro-averaged F_1 (left) and ER (right) evaluation for different ratios of manually classified documents using APA for `Reuters-21578`, `Reuters-21578-115`, and `20-newsgroups`.



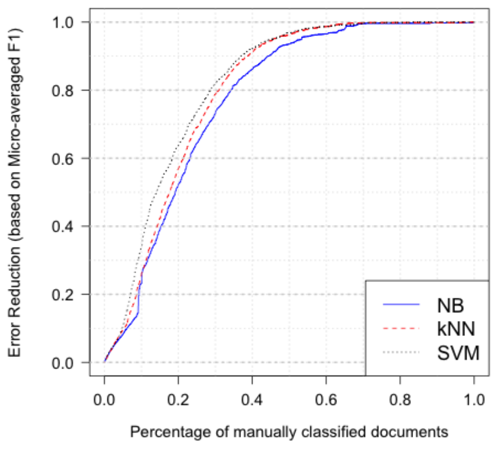
(a) Reuters-21578 APW



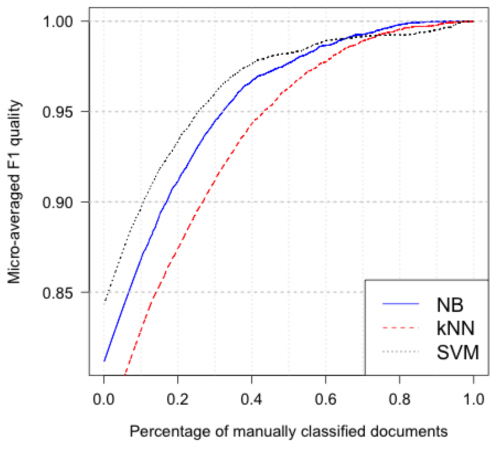
(b) Reuters-21578 APW



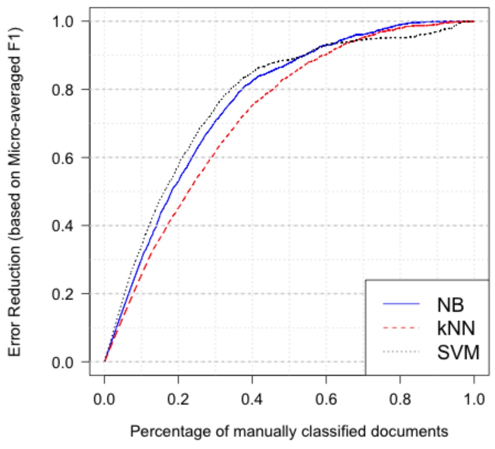
(c) Reuters-21578-115 APW



(d) Reuters-21578-115 APW

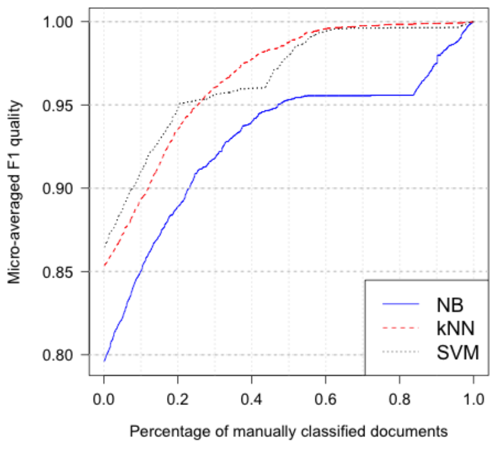


(e) 20-newsgroups APW

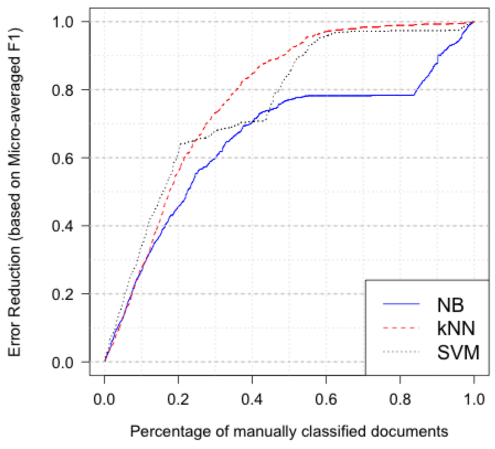


(f) 20-newsgroups APW

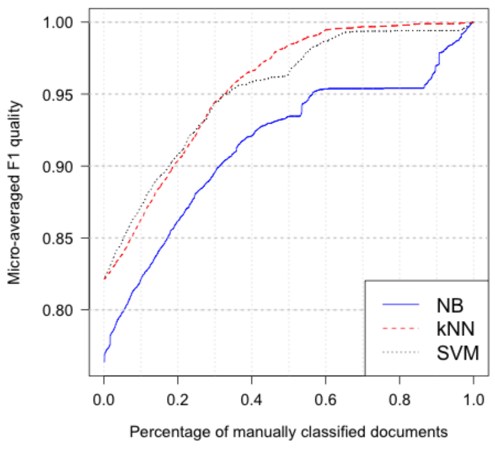
Figure A.12: Micro-averaged F_1 (left) and ER (right) evaluation for different ratios of manually classified documents using APW for Reuters-21578, Reuters-21578-115, and 20-newsgroups.



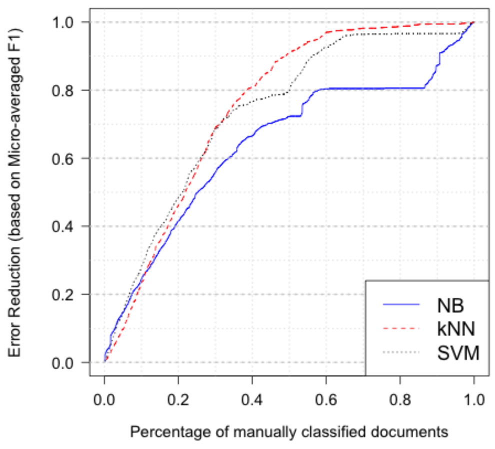
(a) `Reuters-21578` RAM



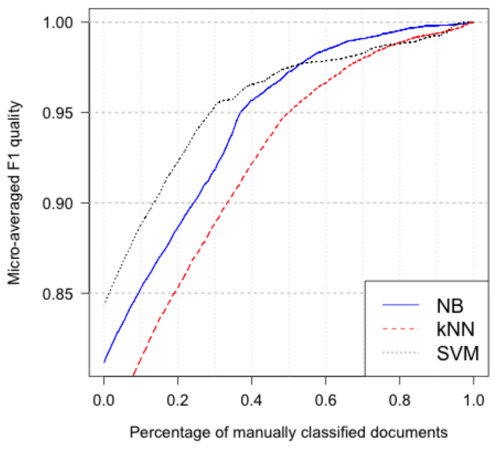
(b) `Reuters-21578` RAM



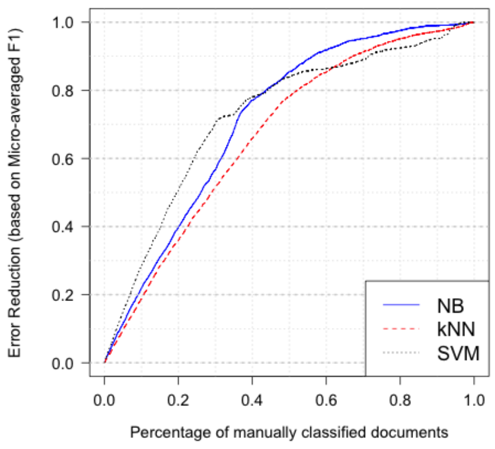
(c) `Reuters-21578-115` RAM



(d) `Reuters-21578-115` RAM

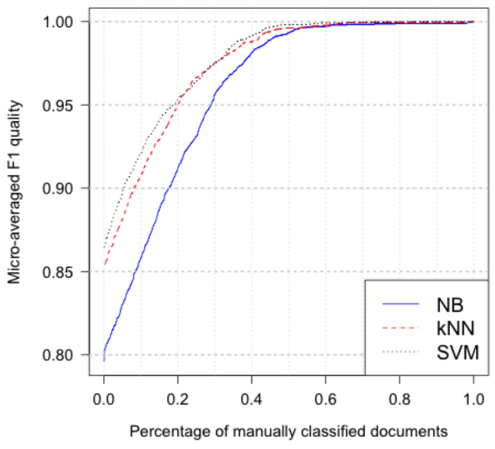


(e) `20-newsgroups` RAM

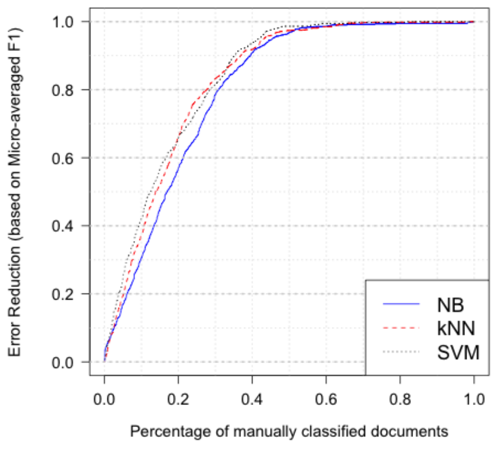


(f) `20-newsgroups` RAM

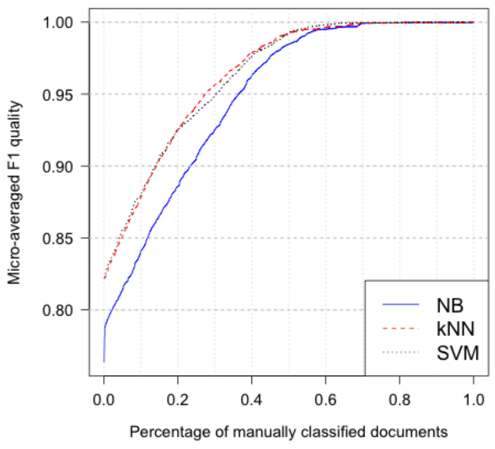
Figure A.13: Micro-averaged F_1 (left) and ER (right) evaluation for different ratios of manually classified documents using RAM for `Reuters-21578`, `Reuters-21578-115`, and `20-newsgroups`.



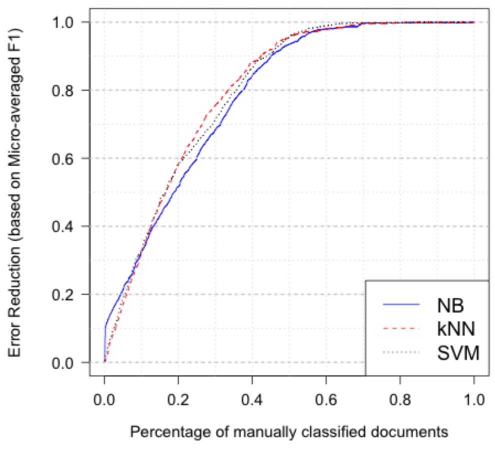
(a) `Reuters-21578` RAA



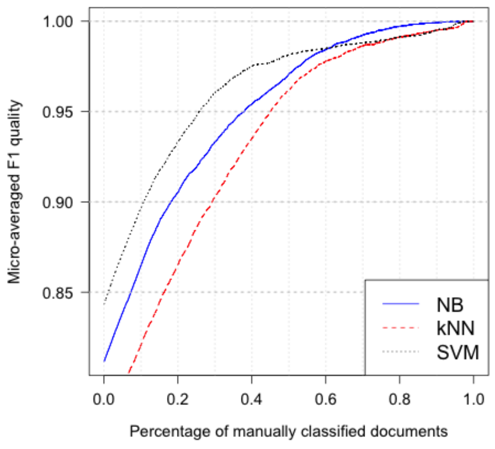
(b) `Reuters-21578` RAA



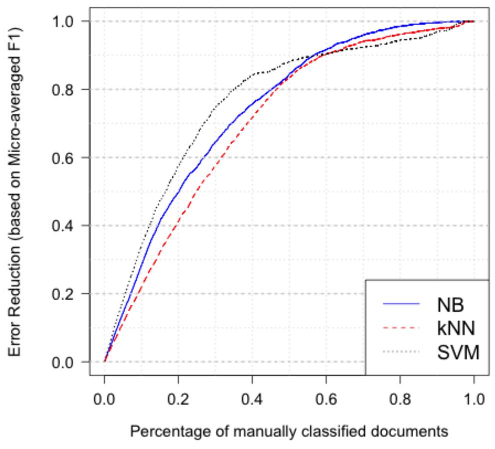
(c) `Reuters-21578-115` RAA



(d) `Reuters-21578-115` RAA

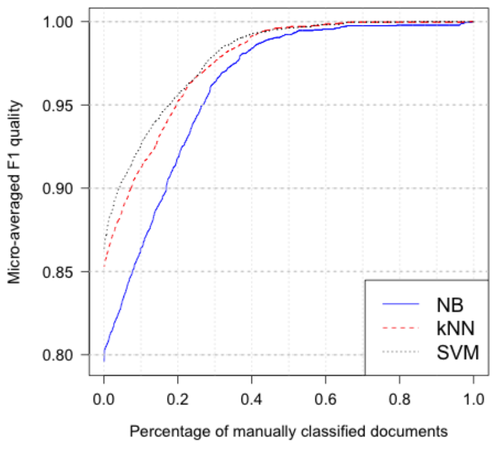


(e) `20-newsgroups` RAA

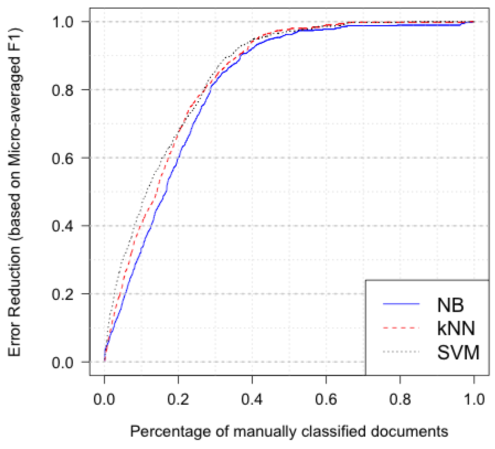


(f) `20-newsgroups` RAA

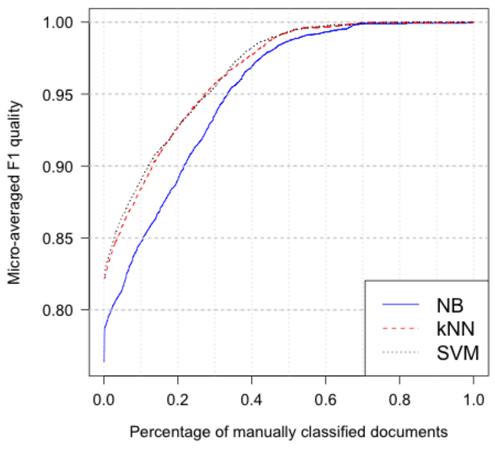
Figure A.14: Micro-averaged F_1 (left) and ER (right) evaluation for different ratios of manually classified documents using RAA for `Reuters-21578`, `Reuters-21578-115`, and `20-newsgroups`.



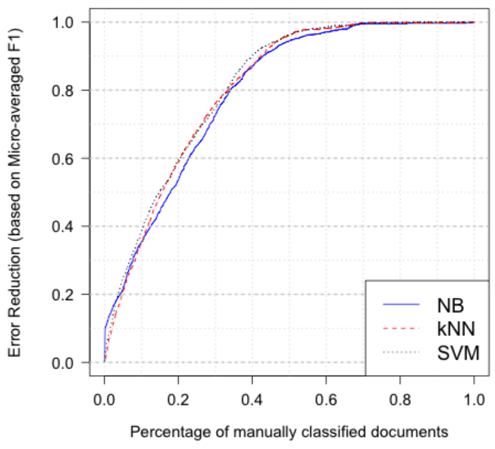
(a) `Reuters-21578` RAW



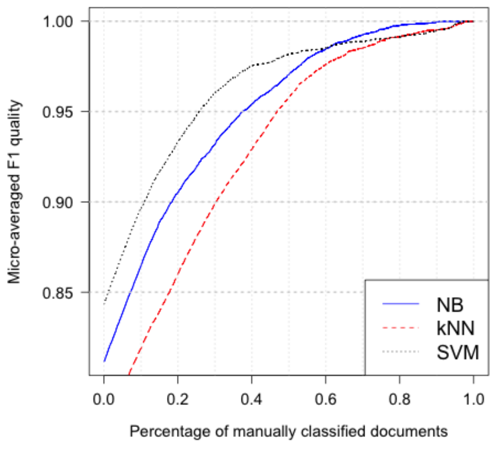
(b) `Reuters-21578` RAW



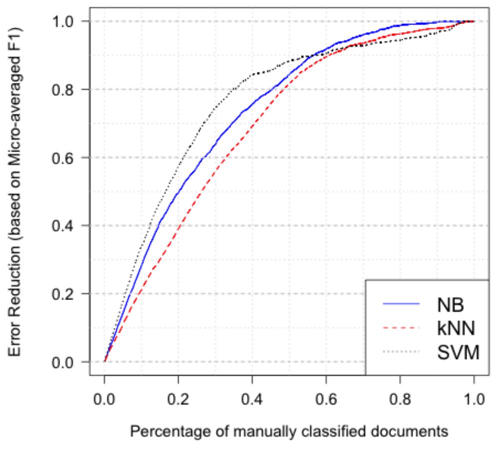
(c) `Reuters-21578-115` RAW



(d) `Reuters-21578-115` RAW

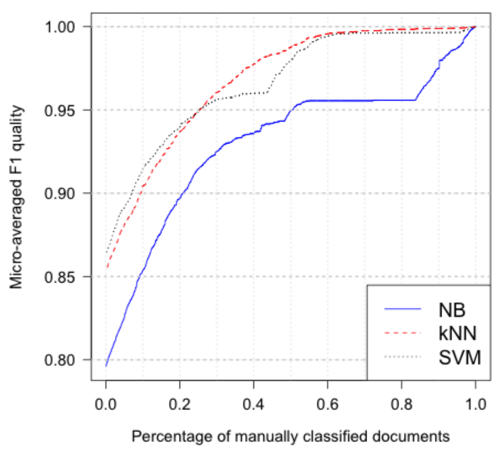


(e) `20-newsgroups` RAW

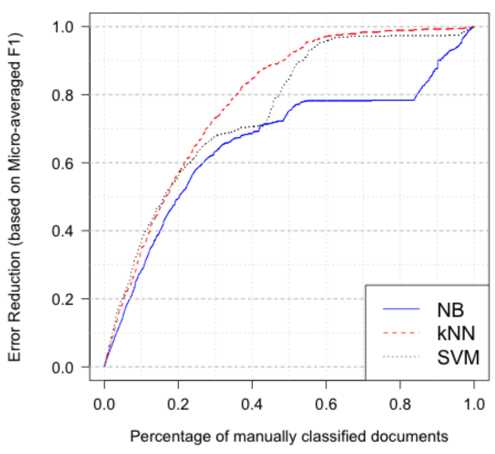


(f) `20-newsgroups` RAW

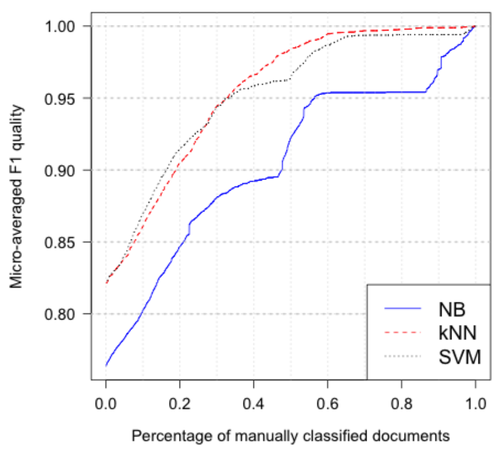
Figure A.15: Micro-averaged F_1 (left) and ER (right) evaluation for different ratios of manually classified documents using RAW for `Reuters-21578`, `Reuters-21578-115`, and `20-newsgroups`.



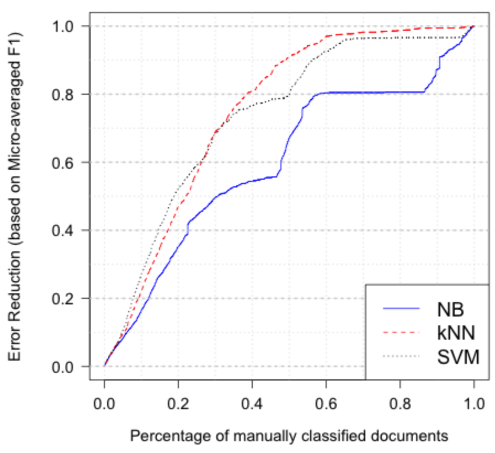
(a) `Reuters-21578` RPM



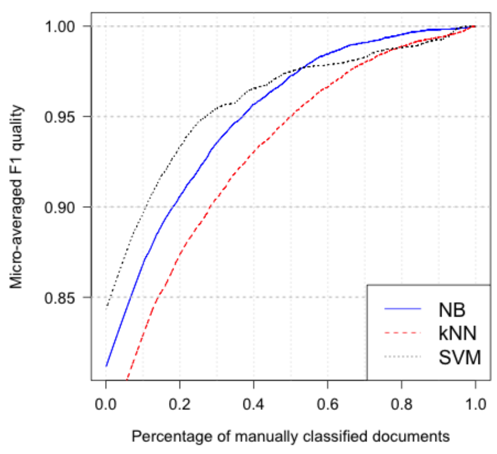
(b) `Reuters-21578` RPM



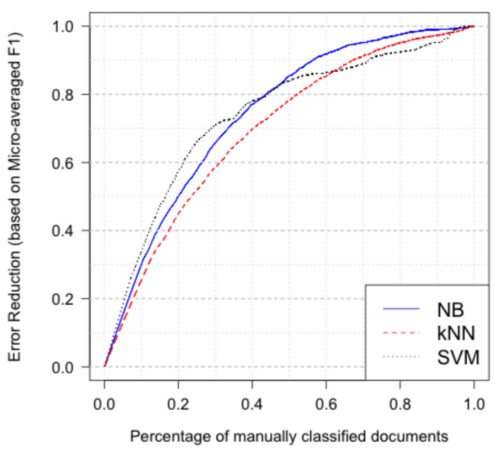
(c) `Reuters-21578-115` RPM



(d) `Reuters-21578-115` RPM

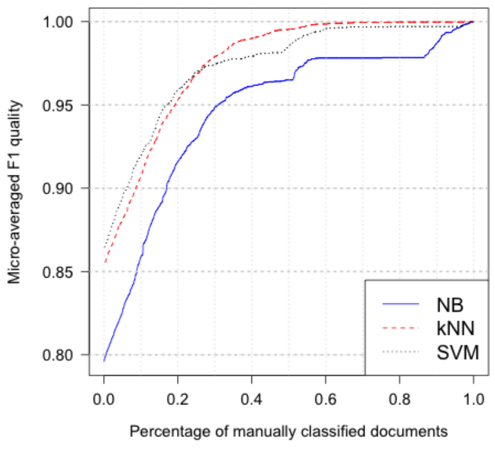


(e) `20-newsgroups` RPM

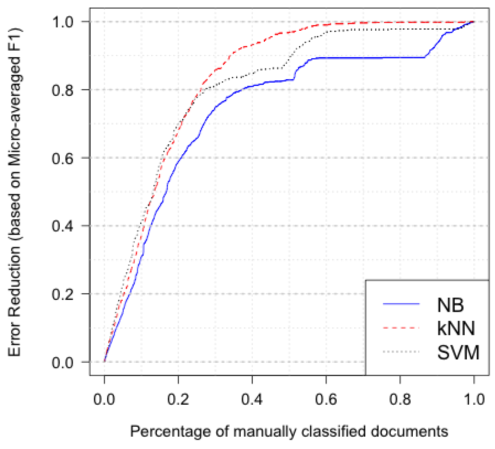


(f) `20-newsgroups` RPM

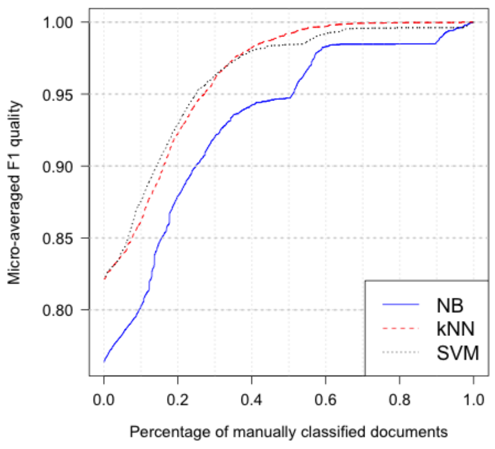
Figure A.16: Micro-averaged F_1 (left) and ER (right) evaluation for different ratios of manually classified documents using RPM for `Reuters-21578`, `Reuters-21578-115`, and `20-newsgroups`.



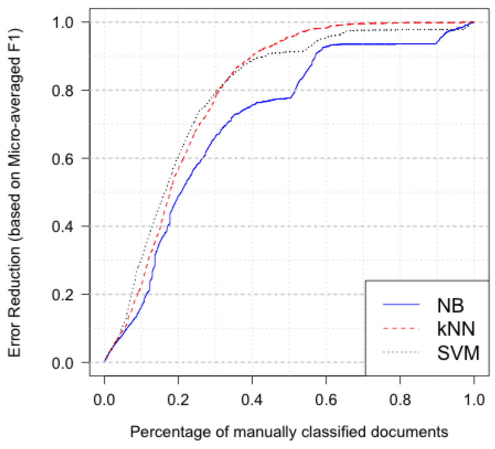
(a) `Reuters-21578` RPA



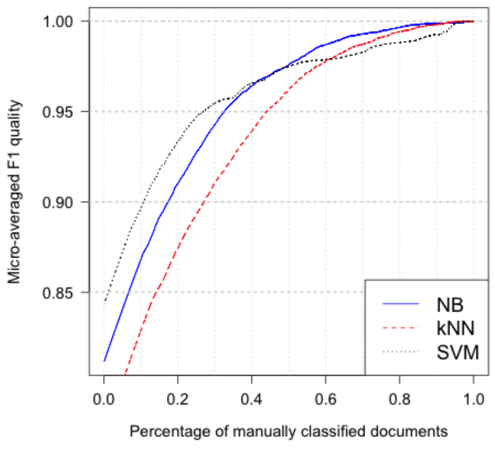
(b) `Reuters-21578` RPA



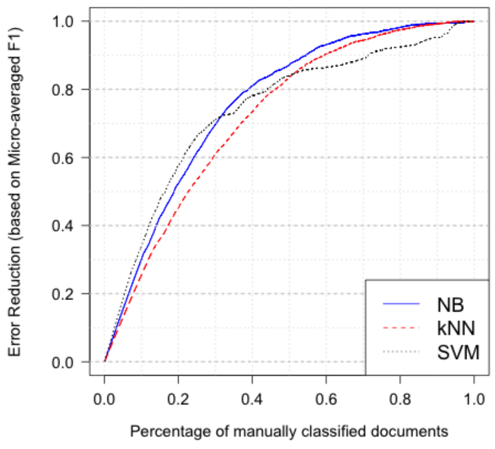
(c) `Reuters-21578-115` RPA



(d) `Reuters-21578-115` RPA

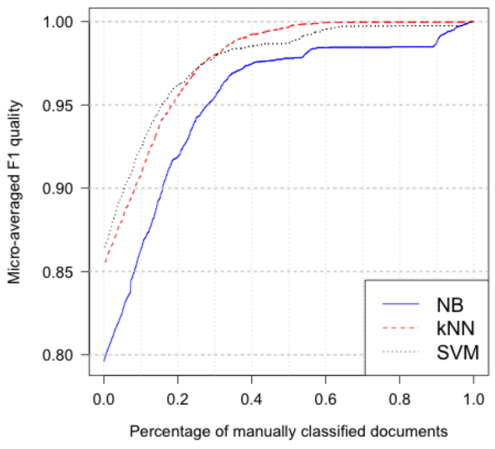


(e) `20-newsgroups` RPA

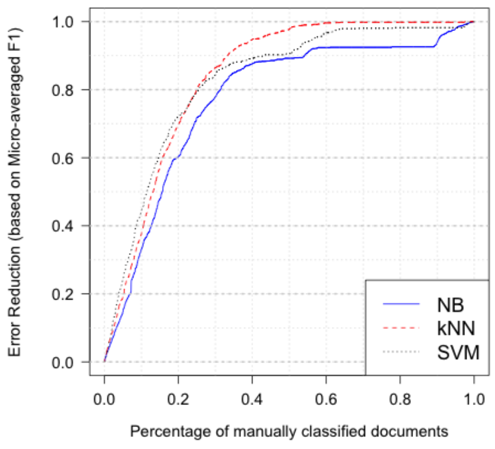


(f) `20-newsgroups` RPA

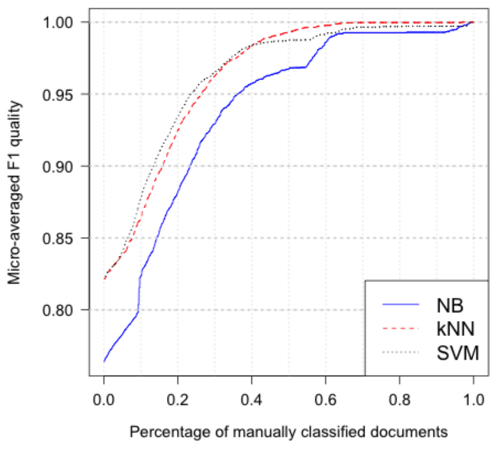
Figure A.17: Micro-averaged F_1 (left) and ER (right) evaluation for different ratios of manually classified documents using RPA for `Reuters-21578`, `Reuters-21578-115`, and `20-newsgroups`.



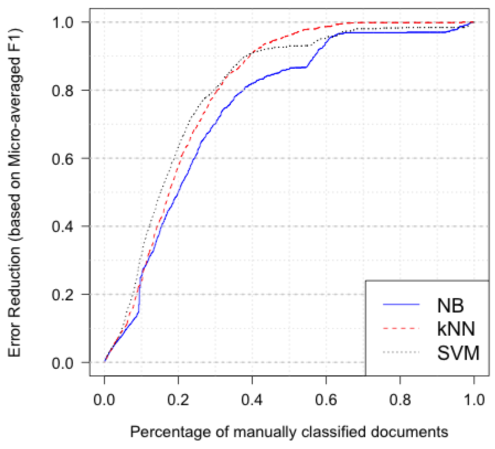
(a) `Reuters-21578` RPW



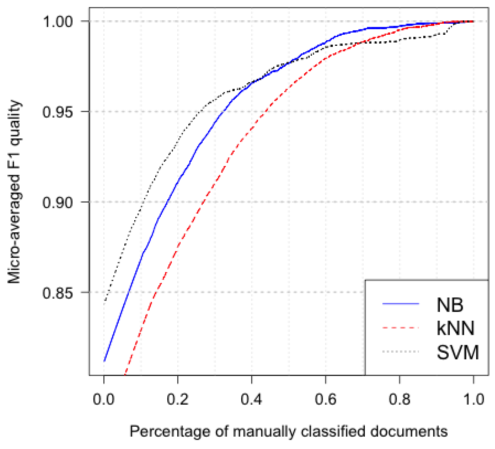
(b) `Reuters-21578` RPW



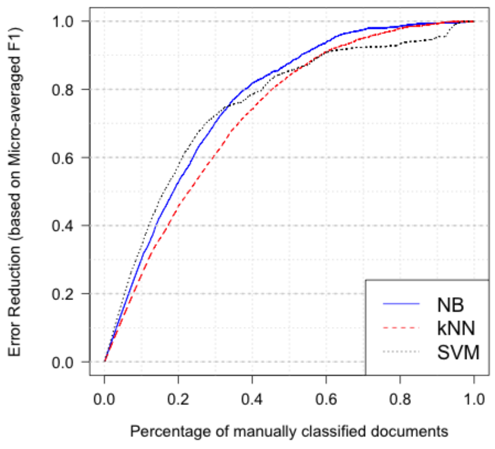
(c) `Reuters-21578-115` RPW



(d) `Reuters-21578-115` RPW



(e) `20-newsgroups` RPW



(f) `20-newsgroups` RPW

Figure A.18: Micro-averaged F_1 (left) and ER (right) evaluation for different ratios of manually classified documents using RPW for `Reuters-21578`, `Reuters-21578-115`, and `20-newsgroups`.

Bibliography

[Abrol et al., 2001] Abrol, M., Latarche, N., Mahadevan, U., Mao, J., Mukherjee, R., Raghavan, P., Tourn, M., Wang, J., and Zhang, G. (2001). Navigating large-scale semi-structured data in business portals. In *Proceedings of the 27th International Conference on Very Large Data Bases (VLDB)*, pages 663–666. Morgan Kaufmann Publishers Inc.

[Aiolli et al., 2009] Aiolli, F., Cardin, R., Sebastiani, F., and Sperduti, A. (2009). Preferential text classification: learning algorithms and evaluation measures. *Information Retrieval*, 12(5):559–580.

[Amati et al., 2004] Amati, G., Carpineto, C., and Romano, G. (2004). Query difficulty, robustness, and selective application of query expansion. In *Proceedings of the 26th European Conference on Information Retrieval (ECIR)*, pages 127–137. Springer.

[Anglade and Dixon, 2008] Anglade, A. and Dixon, S. (2008). Characterisation of harmony with inductive logic programming. In *Proceedings of the 9th International Conference on Music Information Retrieval (ISMIR)*, pages 63–68.

[Anglade et al., 2009] Anglade, A., Ramirez, R., and Dixon, S. (2009). First-order logic classification models of musical genres based on harmony. In *Proceedings of the 6th Sound and Music Computing Conference (SMC)*, pages 309–314.

[Aslam and Pavlu, 2007] Aslam, J. A. and Pavlu, V. (2007). Query Hardness Estimation Using Jensen-Shannon Divergence Among Multiple Scoring Functions. In *Proceedings of the 29th European Conference on Information Retrieval (ECIR)*, pages 198–209. Springer.

[Azzam and Roelleke., 2010] Azzam, H. and Roelleke., T. (2010). An attribute-based model for semantic retrieval. In *Proceedings of the Workshop-Woche: Lernen, Wissen & Adaptivitaet (LWA)*, Kassel, Germany.

[Azzam and Roelleke, 2011] Azzam, H. and Roelleke, T. (2011). A generic data model for schema-driven design in information retrieval applications. In *Proceedings of the 3rd International Conference on the Theory of Information Retrieval (ICTIR)*. Springer.

[Azzam et al., 2012] Azzam, H., Yahyaei, S., Bonzanini, M., and Roelleke, T. (2012). A schema-driven approach for knowledge-oriented retrieval and query formulation. In *Proceedings of the ACMCIKM 3rd International Workshop on Keyword Search on Structured Data (KEYS)*, pages 39–46. ACM.

[Balasubramanian and Allan, 2010] Balasubramanian, N. and Allan, J. (2010). Learning to select rankers. In *Proceedings of the 33th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, pages 855–856. ACM.

[Baoli et al., 2004] Baoli, L., Qin, L., and Shiwen, Y. (2004). An adaptive k-nearest neighbor text categorization strategy. *ACM Transactions on Asian Language Information Processing (TALIP)*, 3(4):215–226.

[Batal and Hauskrecht, 2009] Batal, I. and Hauskrecht, M. (2009). Boosting KNN text classification accuracy by using supervised term weighting schemes. In *Proceedings of the 18th ACM Conference on Information and Knowledge Management (CIKM)*, pages 2041–2044. ACM.

[Belkin and Croft, 1992] Belkin, N. J. and Croft, W. B. (1992). Information filtering and information retrieval: two sides of the same coin? *Communications of the ACM*, 35(12):29–38.

[Bellogín et al., 2012] Bellogín, A., Wang, J., and Castells, P. (2012). Bridging memory-based collaborative filtering and text retrieval. *Information Retrieval*, pages 1–28.

[Bennett et al., 2005] Bennett, P. N., Dumais, S. T., and Horvitz, E. (2005). The combination of text classifiers using reliability indicators. *Information Retrieval*, 8(1):67–100.

[Berardi et al., 2012] Berardi, G., Esuli, A., and Sebastiani, F. (2012). A utility-theoretic ranking method for semi-automated text classification. In *Proceedings of the 35th International Conference on Research and Development in Information Retrieval (SIGIR)*, pages 961–970. ACM.

[Berners-Lee et al., 2001] Berners-Lee, T., Hendler, J., Lassila, O., et al. (2001). The semantic web. *Scientific american*, 284(5):28–37.

[Bird et al., 2009] Bird, S., Klein, E., and Loper, E. (2009). *Natural Language Processing with Python*. O'Reilly Media Inc.

[Bishop, 1995] Bishop, C. M. (1995). *Neural networks for pattern recognition*. Oxford University Press, Oxford.

[Bloehdorn and Hotho, 2004] Bloehdorn, S. and Hotho, A. (2004). Text classification by boosting weak learners based on terms and concepts. In *Proceedings of the 4th IEEE International Conference on Data Mining (ICDM)*, pages 331–334. IEEE Computer Society.

[Bonzanini et al., 2012a] Bonzanini, M., Martinez-Alvarez, M., and Roelleke, T. (2012a). Investigating the Use of Extractive Summarisation in Sentiment Classification. In *Proceedings of the 3rd Italian Information Retrieval Workshop (IIR)*, volume 835 of *CEUR Workshop Proceedings*, pages 45–52. CEUR-WS.org.

[Bonzanini et al., 2012b] Bonzanini, M., Martinez-Alvarez, M., and Roelleke, T. (2012b). Opinion summarisation through sentence extraction: an investigation with movie reviews. In *Proceedings of the 35th International SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, pages 1121–1122. ACM.

[Bonzanini et al., 2013] Bonzanini, M., Martinez-Alvarez, M., and Roelleke, T. (2013). Extractive Summarisation via Sentence Removal: Condensing Relevant Sentences into a Short Summary. In *Proceedings of the 36th International SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*. ACM.

[Buckley et al., 1994] Buckley, C., Salton, G., and Allan, J. (1994). The Effect of Adding Relevance Information in a Relevance Feedback Environment. In *Proceedings of the 17th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 292–300. ACM.

[Cai et al., 2005] Cai, D., Shao, Z., He, X., Yan, X., and Han, J. (2005). Mining hidden community in heterogeneous social networks. In *Proceedings of the ACM SIGKDD Workshop on Link Discovery: Issues, Approaches and Applications (LinkKDD)*, pages 2005–2538. ACM Press.

[Carmel and Yom-Tov, 2010] Carmel, D. and Yom-Tov, E. (2010). *Estimating the Query Diffi-*

culty for Information Retrieval. Synthesis Lectures on Information Concepts, Retrieval, and Services. Morgan & Claypool Publishers.

[Carmel et al., 2006] Carmel, D., Yom-Tov, E., Darlow, A., and Pelleg, D. (2006). What makes a query difficult? In *Proceedings of the 29th International ACM SIGIR Conference on Research and development in information retrieval*, pages 390–397. ACM.

[Chakrabarti et al., 1998] Chakrabarti, S., Dom, B., and Indyk, P. (1998). Enhanced hypertext categorization using hyperlinks. In *Proceedings of the 1998 ACM SIGMOD international conference on Management of data*, SIGMOD '98, pages 307–318, New York, NY, USA. ACM.

[Chang and Lin, 2011] Chang, C.-C. and Lin, C.-J. (2011). LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems (TOIS)*, 2:27:1–27:27.

[Cornacchia and de Vries, 2007] Cornacchia, R. and de Vries, A. P. (2007). A Parameterised Search System. In *Proceedings of the 29th European Conference on Information Retrieval (ECIR)*, volume 4425 of *Lecture Notes in Computer Science*, pages 4–15. Springer.

[Cortes and Vapnik, 1995] Cortes, C. and Vapnik, V. (1995). Support-vector networks. *Machine Learning*, 20(3):273–297.

[Courant and Hilbert, 1943] Courant, R. and Hilbert, D. (1943). *Methods of mathematical physics*. Interscience.

[Cronen-Townsend et al., 2002] Cronen-Townsend, S., Zhou, Y., and Croft, W. B. (2002). Predicting query performance. In *Proceedings of the 25th ACM SIGIR International Conference on Research and Development in Information Retrieval (SIGIR)*, pages 299–306. ACM.

[Cumbo et al., 2004] Cumbo, C., Iiritano, S., and Rullo, P. (2004). Reasoning-Based Knowledge Extraction for Text Classification. In *Proceedings of the 7th International Conference on Discovery Science (DS)*, volume 3245 of *Lecture Notes in Computer Science*, pages 380–387. Springer.

[Cummins et al., 2011] Cummins, R., Jose, J. M., and O'Riordan, C. (2011). Improved query performance prediction using standard deviation. In *Proceeding of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, pages 1089–1090. ACM.

[Dasarathy, 1991] Dasarathy, B. V. (1991). *Nearest Neighbor ({NN}) Norms:{NN} Pattern Classification Techniques*. IEEE Computer Society Press.

[de Vries, 2001] de Vries, A. P. (2001). Content independence in multimedia databases. *Journal of the American Society for Information Science and Technology (JASIST)*, 52(11):954–960.

[Dean and Henzinger, 1999] Dean, J. and Henzinger, M. R. (1999). Finding related pages in the world wide web. *Computer Networks*, 31(11-16):1467–1479.

[Dell'Armi et al., 2003] Dell'Armi, T., Faber, W., Wien, T., Leone, N., and Pfeifer, G. (2003). Aggregate functions in disjunctive logic programming: Semantics, complexity, and implementation in dlv. In *In Proceedings of the 18th International Joint Conference on Artificial Intelligence (IJCAI)*, pages 847–852. Morgan Kaufmann.

[Deloo and Hauff, 2013] Deloo, C. and Hauff, C. (2013). Exploiting semantic relatedness measures for multi-label classifier evaluation. In *Proceedings of the 13th Dutch-Belgian Information Retrieal Workshop*. CEUR-WS.org.

[Deng et al., 2011] Deng, H., Zhao, B., and Han, J. (2011). Collective topic modeling for heterogeneous networks. In *Proceedings of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, pages 1109–1110. ACM.

[Dietterich, 2000] Dietterich, T. G. (2000). Ensemble methods in machine learning. In *Multiple classifier systems*, pages 1–15. Springer.

[Dixon et al., 2011] Dixon, S., Mauch, M., and Anglade, A. (2011). Probabilistic and logic-based modelling of harmony. In *Proceedings of the 7th international Conference on Exploring Music Contents (CMMR)*, pages 1–19. Springer-Verlag.

[Domingos, 1999] Domingos, P. (1999). Metacost: a general method for making classifiers cost-sensitive. In *Proceedings of the 5th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 155–164. ACM.

[Eisner et al., 2005] Eisner, J., Goldlust, E., and Smith, N. A. (2005). Compiling comp ling: practical weighted dynamic programming and the dyna language. In *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing (HLT)*, pages 281–290. Association for Computational Linguistics.

[Elkan, 2001] Elkan, C. (2001). The foundations of cost-sensitive learning. In *Proceedings of the 17th International Joint Conference on Artificial intelligence (IJCAI)*, pages 973–978. Morgan Kaufmann Publishers Inc.

[Esuli and Sebastiani, 2009] Esuli, A. and Sebastiani, F. (2009). Active Learning Strategies for Multi-Label Text Classification. In *Proceedings of the 31th European Conference on Information Retrieval (ECIR)*, volume 5478 of *Lecture Notes in Computer Science*, pages 102–113. Springer.

[Forst et al., 2007a] Forst, J. F., Roelleke, T., Tombros, A., and Mary, Q. (2007a). Modelling a summarisation logic in probabilistic datalog. In *Proceedings of the Workshop-Woche: Lernen, Wissen & Adaptivitaet (LWA)*, pages 221–228.

[Forst et al., 2007b] Forst, J. F., Tombros, A., and Roelleke, T. (2007b). Polis: A probabilistic logic for document summarisation. In *Proceedings of the 1st International Conference on the Theory of Information Retrieval (ICTIR)*, pages 201–212.

[Frommholz, 2007] Frommholz, I. (2007). Annotation-based document retrieval with probabilistic logics. In *Proceedings of the 11th European Conference on Research and Advanced Technology for Digital Libraries (ECDL)*, pages 321–332. Springer-Verlag.

[Frommholz and Lechtenfeld, 2008] Frommholz, I. and Lechtenfeld, M. (2008). Determining the polarity of postings for discussion search. In *Proceedings of the Workshop-Woche: Lernen, Wissen & Adaptivitaet (LWA)*, pages 49–56.

[Fuhr, 1995] Fuhr, N. (1995). Probabilistic Datalog: a logic for powerful retrieval methods. In *Proceedings of the 18th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, pages 282–290. ACM.

[Fuhr, 1996] Fuhr, N. (1996). Object-oriented and database concepts for the design of networked information retrieval systems. In *Proceedings of the 5th International Conference on Information and Knowledge Management (CIKM)*, pages 164–172, New York, NY, USA. ACM.

[Fuhr, 2000] Fuhr, N. (2000). Probabilistic datalog: Implementing logical information retrieval for advanced applications. *Journal of the American Society for Information Science*, 51:95–110.

[Fuhr et al., 1998] Fuhr, N., Gövert, N., and Rölleke, T. (1998). Dolores: A system for logic-based retrieval of multimedia objects. In *Proceedings of the 21st International Conference on Research and Development in Information Retrieval (SIGIR)*, pages 257–265. ACM.

[Fuhr and Roelleke, 1997] Fuhr, N. and Roelleke, T. (1997). A probabilistic relational algebra for the integration of information retrieval and database systems. *ACM Transactions on Information Systems (TOIS)*, 15(1):32–66.

[Fuhr and Rölleke, 1998] Fuhr, N. and Rölleke, T. (1998). Hyspirita probabilistic inference engine for hypermedia retrieval in large databases. In *Proceedings of the 6th International Conference on Extending Database Technology (EDBT)*, pages 24–38. Springer.

[Gabrilovich and Markovitch, 2005] Gabrilovich, E. and Markovitch, S. (2005). Feature Generation for Text Categorization Using World Knowledge. In *Proceedings of the 19th International Joint Conference on Artificial Intelligence (IJCAI)*, pages 1048–1053. Professional Book Center.

[Gabrilovich and Markovitch, 2006] Gabrilovich, E. and Markovitch, S. (2006). Overcoming the Brittleness Bottleneck using Wikipedia: Enhancing Text Categorization with Encyclopedic Knowledge. In *Proceedings of the 21st National Conference on Artificial Intelligence (AAAI)*, pages 1301–1306. AAAI Press.

[Gelfond et al., 2006] Gelfond, M., Rushton, J. N., and Zhu, W. (2006). Combining logical and probabilistic reasoning. In *Proceedings of the AAAI Spring Symposium: Formalizing and Compiling Background Knowledge and Its Applications to Knowledge Representation and Question Answering*, pages 50–55. AAAI.

[Getoor et al., 2003] Getoor, L., Friedman, N., Koller, D., and Taskar, B. (2003). Learning probabilistic models of link structure. *Journal of Machine Learning Research*, 3:679–707.

[Getoor and Mihalkova, 2011] Getoor, L. and Mihalkova, L. (2011). Learning statistical models from relational data. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*, pages 1195–1198. ACM.

[Gövert et al., 1999] Gövert, N., Lalmas, M., and Fuhr, N. (1999). A probabilistic description-oriented approach for categorizing web documents. In *Proceedings of the eighth international* *conference on Information and knowledge management*, CIKM '99, pages 475–482, New York, NY, USA. ACM.

[Grossman et al., 1997] Grossman, D. A., Frieder, O., Holmes, D. O., and Roberts, D. C. (1997). Integrating structured data and text: A relational approach. *Journal of the American Society of Information Science (JASIS)*, 48.

[Guo et al., 2004] Guo, G., Wang, H., Bell, D., Bi, Y., and Greer, K. (2004). An knn model-based approach and its application in text categorization. In *Proceedings of the 5th Computational Linguistics and Intelligent Text Processing (CICLing)*, pages 559–570. Springer.

[Hall et al., 2009] Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., and Witten, I. H. (2009). The weka data mining software: an update. *SIGKDD Explor. Newsl.*, 11:10–18.

[Hauff et al., 2009] Hauff, C., Azzopardi, L., and Hiemstra, D. (2009). The Combination and Evaluation of Query Performance Prediction Methods. In *Proceedings of the 31st European Conference on Information Retrieval (ECIR)*, pages 301–312. Springer-Verlag.

[Hauff et al., 2008] Hauff, C., Hiemstra, D., and de Jong, F. (2008). A survey of pre-retrieval query performance predictors. In *Proceeding of the 17th ACM Conference on Information and Knowledge Management (CIKM)*, pages 1419–1420, New York, NY, USA. ACM.

[Hawking, 2004] Hawking, D. (2004). Challenges in enterprise search. In *Proceedings of the Australasian Database Conference (ADC)*, pages 15–26.

[He and Ounis, 2006] He, B. and Ounis, I. (2006). Query performance prediction. *Information Systems*, 31(7):585–594.

[He et al., 2008] He, J., Larson, M., and De Rijke, M. (2008). Using coherence-based measures to predict query difficulty. In *Proceedings of the 30th European Conference on Information Retrieval (ECIR)*, pages 689–694. Springer-Verlag.

[Hiemstra and Mihajlovic, 2010] Hiemstra, D. and Mihajlovic, V. (2010). A database approach to information retrieval: The remarkable relationship between language models and region models. *CoRR*, abs/1005.4752.

[Hosseini et al., 2011] Hosseini, M., Cox, I. J., Milic-Frayling, N., Vinay, V., and Sweeting, T. (2011). Selecting a subset of queries for acquisition of further relevance judgements. In *Pro-*

ceedings of the 3rd International Conference on the Theory of Information Retrieval (ICTIR), pages 113–124. Springer-Verlag.

[Hunter and Liu, 2010] Hunter, A. and Liu, W. (2010). A survey of formalisms for representing and reasoning with scientific knowledge. *The Knowledge Engineering Review*, 25(2):199–222.

[Ji et al., 2010] Ji, M., Sun, Y., Danilevsky, M., Han, J., and Gao, J. (2010). Graph Regularized Transductive Classification on Heterogeneous Information Networks. In *Proceedings of the 10th Machine Learning and Knowledge Discovery in Databases (ECML/PKDD)*, volume 6321 of *Lecture Notes in Computer Science*, pages 570–586. Springer.

[Joachims, 1998] Joachims, T. (1998). Text Categorization with Suport Vector Machines: Learning with Many Relevant Features. In *Proceedings of the 10th European Conference on Machine Learning*, volume 1398 of *Lecture Notes in Computer Science*, pages 137–142. Springer.

[Jones and Diaz, 2007] Jones, R. and Diaz, F. (2007). Temporal profiles of queries. *ACM Transactions on Information Systems (TOIS)*, 25(3).

[Khan and Madden, 2010] Khan, S. S. and Madden, M. G. (2010). A Survey of Recent Trends in One Class Classification. In *AICS '09: Proceedings of the 20th Irish Conference on Artificial Intelligence and Cognitive Science*, pages 188–197. Springer-Verlag.

[Klampanos et al., 2010] Klampanos, I. A., Wu, H., Roelleke, T., and Azzam, H. (2010). Logic-based retrieval: Technology for content-oriented and analytical querying of patent data. In *Proceedings of the 1st Information Retrieval Facility Conference (IRFC)*, pages 100–119. Springer.

[Klimt and Yang, 2004] Klimt, B. and Yang, Y. (2004). The Enron Corpus: A New Dataset for Email Classification Research. In *Proceedings of the 15th European Conference on Machine Learning (ECML)*, volume 3201 of *Lecture Notes in Computer Science*, pages 217–226. Springer.

[Kurland et al., 2011] Kurland, O., Shtok, A., Carmel, D., and Hummel, S. (2011). A unified framework for post-retrieval query-performance prediction. In *Proceedings of the 3rd International Conference on Information Retrieval Theory (ICTIR)*, pages 15–26. Springer-Verlag.

[Lam et al., 1999] Lam, W., Ruiz, M., Ruiz, M., Srinivasan, P., and Srinivasan, P. (1999). Automatic text categorization and its application to text retrieval. *IEEE Transactions on Knowledge and Data Engineering*, 11:865–879.

[Larkey and Croft, 1996] Larkey, L. S. and Croft, W. B. (1996). Combining Classifiers in Text Categorization. In *Proceedings of the 19th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 289–297. ACM.

[Lewis and Gale, 1994] Lewis, D. D. and Gale, W. A. (1994). A Sequential Algorithm for Training Text Classifiers. In *Proceedings of the 17th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 3–12. ACM.

[Lewis et al., 2004] Lewis, D. D., Yang, Y., Rose, T., and Li, F. (2004). Rcv1: A new benchmark collection for text categorization research. *The Journal of Machine Learning Research*, 5:361–397.

[Li and Jain, 1998] Li, Y. H. and Jain, A. K. (1998). Classification of text documents. *The Computer Journal*, 41(8):537–546.

[Liere and Tadepalli, 1997] Liere, R. and Tadepalli, P. (1997). Active learning with committees for text categorization. In *Proceedings of the 14th National Conference on Artificial Intelligence (AAAI)*, pages 591–597. John Wiley and Sons Ltd.

[Liu, 2009] Liu, T.-Y. (2009). Learning to rank for information retrieval. *Foundations and Trends in Information Retrieval*, 3(3):225–331.

[Lloyd, 1994] Lloyd, J. W. (1994). Practical Advantages of Declarative Programming. In *Proceedings of Joint Conference on Declarative Programming (GULP-PRODE'94)*.

[Lopez, 2009] Lopez, A. (2009). Translation as weighted deduction. In *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics (EACL)*, pages 532–540. Association for Computational Linguistics.

[Lu and Getoor, 2003] Lu, Q. and Getoor, L. (2003). Link-based classification. In *Proceedings of the 20th International Conference on Machine Learning (ICML)*, pages 496–503. AAAI Press.

[Luhn, 1960] Luhn, H. P. (1960). Key word-in-context index for technical literature (kwic index). *American Documentation*, 11(4):288–295.

[Martinez-Alvarez, 2011] Martinez-Alvarez, M. (2011). Descriptive modelling of text classification and its integration with other IR tasks. In *Proceedings of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, pages 1317–1318. ACM.

[Martinez-Alvarez et al., 2013] Martinez-Alvarez, M., Bellogin, A., and Roelleke, T. (2013). Document Difficulty Framework for Semi-Automatic Text Classification. In *Proceedings of the 15th International Conference on Data Warehousing and Knowledge Discovery (DaWak)*. ACM.

[Martinez-Alvarez and Roelleke, 2010] Martinez-Alvarez, M. and Roelleke, T. (2010). Modelling Probabilistic Inference Networks and Classification in Probabilistic Datalog. In *Proceedings on the 4th International Conference on Scalable Uncertainty Management (SUM)*, volume 6379 of *Lecture Notes in Computer Science*, pages 278–291. Springer.

[Martinez-Alvarez and Roelleke, 2011] Martinez-Alvarez, M. and Roelleke, T. (2011). A Descriptive Approach to Classification. In *Proceedings of the 3rd International Conference on the Theory of Information Retrieval (ICTIR)*, volume 6931 of *Lecture Notes in Computer Science*, pages 297–308. Springer.

[Martinez-Alvarez and Roelleke, 2013] Martinez-Alvarez, M. and Roelleke, T. (2013). Mathematical Specification and Logic Modelling in the context of IR. In *Proceedings of the 4th International Conference on Theory on Information Retrieval (ICTIR)*. Springer.

[Martinez-Alvarez et al., 2010] Martinez-Alvarez, M., Smeraldi, F., and Roelleke, T. (2010). A Descriptive Approach to Modelling Learning. In *Proceedings of the 1st Spanish Conference on Information Retrieval (CERI)*, pages 183–194.

[Martinez-Alvarez et al., 2012] Martinez-Alvarez, M., Yahyaei, S., and Roelleke, T. (2012). Semi-automatic Document Classification: Exploiting Document Difficulty. In *Proceedings of the 34th European Conference on Information Retrieval (ECIR)*, volume 7224 of *Lecture Notes in Computer Science*, pages 468–471. Springer.

[McCallum and Nigam, 1998] McCallum, A. and Nigam, K. (1998). A Comparison of Event Models for Naive Bayes Text Classification. In *Proceedings of the AAAI Workshop on Learning for Text Categorization*, pages 41–48.

[Mierswa et al., 2006] Mierswa, I., Wurst, M., Klinkenberg, R., Scholz, M., and Euler, T. (2006). Yale: rapid prototyping for complex data mining tasks. In *KDD '06: Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 935–940, New York, NY, USA. ACM.

[Miller, 1995] Miller, G. A. (1995). WordNet: a lexical database for English. *Communications of the ACM*, 38(11):39–41.

[Mitchell, 1997] Mitchell, T. M. (1997). *Machine Learning*. McGraw-Hill, Inc., New York, NY, USA, 1 edition.

[Mothe and Tanguy, 2005] Mothe, J. and Tanguy, L. (2005). Linguistic features to predict query difficulty. In *Proceedings of the ACM SIGIR Workshop on Predicting Query Difficulty - Methods and Applications*. ACM.

[Moya et al., 1993] Moya, M. M., Koch, M. W., and Hostetler, L. D. (1993). One-class classifier networks for target recognition applications. *NASA STI/Recon Technical Report N*, 1993.

[Muggleton, 1991] Muggleton, S. (1991). Inductive logic programming. *New generation computing*, 8(4):295–318.

[Nottelmann, 2005] Nottelmann, H. (2005). PIRE: An Extensible IR Engine Based on Probabilistic Datalog. In *Proceedings of the European Conference on Information Retrieval (ECIR)*, pages 260–274. Springer.

[Nottelmann and Fuhr, 2001] Nottelmann, H. and Fuhr, N. (2001). Learning Probabilistic Datalog Rules for Information Classification and Transformation. In *Proceedings of the 10th International Conference on Information and Knowledge Management (CIKM)*, pages 387–394. ACM.

[Nowak et al., 2010] Nowak, S., Llorente, A., Motta, E., and Rüger, S. (2010). The effect of semantic relatedness measures on multi-label classification evaluation. In *Proceedings of the ACM International Conference on Image and Video Retrieval*, CIVR '10, pages 303–310, New York, NY, USA. ACM.

[Ogilvie and Callan, 2003] Ogilvie, P. and Callan, J. P. (2003). Combining document representations for known-item search. In *Proceedings of the 26th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, pages 143–150. ACM.

[Pérez-Iglesias and Araujo, 2009] Pérez-Iglesias, J. and Araujo, L. (2009). Ranking List Dispersion as a Query Performance Predictor. In *Proceedings of the 2nd International Conference on the Theory of Information Retrieval (ICTIR)*, pages 371–374. Springer-Verlag.

[Platt, 1999] Platt, J. C. (1999). Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. In *ADVANCES IN LARGE MARGIN CLASSIFIERS*, pages 61–74. MIT Press.

[Polleres, 2007] Polleres, A. (2007). From sparql to rules (and back). In *Proceedings of the 16th International Conference on World Wide Web*, WWW '07, pages 787–796, New York, NY, USA. ACM.

[Porter, 1980] Porter, M. F. (1980). An algorithm for suffix stripping. *Program: electronic library and information systems*, 14(3):130–137.

[Quinlan, 1986] Quinlan, J. R. (1986). Induction of decision trees. *Machine learning*, 1(1):81–106.

[Raedt et al., 2007] Raedt, L. D., Kimmig, A., and Toivonen, H. (2007). ProbLog: a probabilistic Prolog and its application in link discovery. In *Proceedings of the 18th International Joint Conference on Artificial Intelligence (IJCAI)*, pages 2468–2473. IJCAI/AAAI.

[Read et al., 2009] Read, J., Pfahringer, B., Holmes, G., and Frank, E. (2009). Classifier chains for multi-label classification. *Machine Learning and Knowledge Discovery in Databases*, pages 254–269.

[Robertson et al., 2004] Robertson, S. E., Zaragoza, H., and Taylor, M. J. (2004). Simple bm25 extension to multiple weighted fields. In *Proceedings of the 13th ACM Conference on Information and Knowledge Management (CIKM)*, pages 42–49. ACM.

[Roelleke et al., 2013a] Roelleke, T., Azzam, H., Bonzanini, M., Martinez-Alvarez, M., and Lalmas, M. (2013a). The D2Q2 Framework: On the Relationship and Combination of Language Modelling and TF-IDF. In *Proceedings of the Workshop-Woche: Lernen, Wissen & Adaptivitaet (LWA)*. .

[Roelleke et al., 2013b] Roelleke, T., Bonzanini, M., and Martinez-Alvarez, M. (2013b). On the Modelling of Ranking Algorithms in Probabilistic Datalog. In *Proceedings of the 7th International Workshop on Ranking in Databases (DBRank)*. ACM.

[Roelleke et al., 2006] Roelleke, T., Tsikrika, T., and Kazai, G. (2006). A general matrix framework for modelling information retrieval. *Journal on Information Processing & Management (IP&M), Special Issue on Theory in Information Retrieval*, 42(1).

[Roelleke et al., 2008] Roelleke, T., Wu, H., Wang, J., and Azzam, H. (2008). Modelling Retrieval Models in a Probabilistic Relational Algebra with a new Operator: The Relational Bayes. *The International Journal on Very Large Data Bases (VLDB)*, 17(1):5–37.

[Rolleke et al., 2001] Rolleke, T., Lubeck, R., and Kazai, G. (2001). The HySpirit retrieval platform. In *Proceedings of the 24th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, page 454. ACM.

[Salton, 1971] Salton, G. (1971). *The SMART Retrieval System: Experiments in Automatic Document Processing*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA.

[Salton, 1988] Salton, G. (1988). Automatic text indexing using complex identifiers. In *Proceedings of the ACM conference on Document processing systems*, DOCPROCS '88, pages 135–144, New York, NY, USA. ACM.

[Schapire et al., 1998] Schapire, R. E., Singer, Y., and Singhal, A. (1998). Boosting and Rocchio Applied to Text Filtering. In *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 215–223. ACM.

[Sebastiani, 2002] Sebastiani, F. (2002). Machine learning in automated text categorization. *ACM computing surveys (CSUR)*, 34(1):1–47.

[Shen et al., 2009] Shen, D., Wu, J., Cao, B., Sun, J.-T., Yang, Q., Chen, Z., and Li, Y. (2009). Exploiting term relationship to boost text classification. In *Proceeding of the 18th ACM Conference on Information and Knowledge management (CIKM)*, pages 1637–1640. ACM.

[Shen et al., 2007] Shen, W., Doan, A., Naughton, J. F., and Ramakrishnan, R. (2007). Declarative information extraction using datalog with embedded extraction predicates. In *Proceedings of the 33rd international conference on Very large data bases (VLDB)*, pages 1033–1044. VLDB Endowment.

[Slattery and Craven, 1998] Slattery, S. and Craven, M. (1998). Combining statistical and relational methods for learning in hypertext domains. In Page, D., editor, *Inductive Logic Programming*, volume 1446 of *Lecture Notes in Computer Science*, pages 38–52. Springer Berlin Heidelberg.

[Smeraldi et al., 2011] Smeraldi, F., Martinez-Alvarez, M., Frommholz, I., and Roelleke, T. (2011). On the Probabilistic Logical Modelling of Quantum and Geometrically-Inspired IR. In *Proceedings of the 2nd Italian Information Retrieval Workshop (IIR)*, volume 704 of *CEUR Workshop Proceedings*. CEUR-WS.org.

[Sparck-Jones, 1972] Sparck-Jones, K. (1972). A statistical interpretation of term specificity and its application in retrieval. *Journal of documentation*, 28(1):11–21.

[Sun and Lim, 2001] Sun, A. and Lim, E.-P. (2001). Hierarchical Text Classification and Evaluation. In *Proceedings of the 2001 IEEE International Conference on Data Mining (ICDM)*, pages 521–528. IEEE Computer Society.

[Ting and Witten, 1999] Ting, K. M. and Witten, I. H. (1999). Issues in stacked generalization. *Journal of Artificial Intelligence Research*, 10:271–289.

[Tong and Koller, 2001] Tong, S. and Koller, D. (2001). Support vector machine active learning with applications to text classification. *Journal of Machine Learning Research*, 2:45–66.

[Tsoumakas and Katakis, 2007] Tsoumakas, G. and Katakis, I. (2007). Multi-label classification: An overview. *International Journal of Data Warehousing and Mining (IJDWM)*, 3(3):1–13.

[Tumer and Ghosh, 1999] Tumer, K. and Ghosh, J. (1999). Linear and Order Statistics Combiners for Pattern Classification. *CoRR*, cs.NE/9905012.

[Ullman, 1989] Ullman, J. D. (1989). *Principles of Database and Knowledge-Base Systems: The New Technologies*, volume II. Computer Science Press, Rockville, MD.

[van Rijsbergen, 1979] van Rijsbergen, C. J. (1979). *Information retrieval, 2nd edition*. ButterworthsLondon.

[Vapnik, 1995] Vapnik, V. (1995). *The Nature of Statistical Learning Theory*. Springer.

[Vinay et al., 2006] Vinay, V., Cox, I., Milic-Frayling, N., and Wood, K. (2006). Measuring the complexity of a collection of documents. In *Proceedings of the 28th European Conference on Information Retrieval (ECIR)*, volume 3936 of *Lecture Notes in Computer Science*, pages 107–118. Springer Berlin Heidelberg.

[Voorhees, 2000] Voorhees, E. M. (2000). Variations in relevance judgments and the measurement of retrieval effectiveness. *Information Processing & Management*, 36(5):697–716.

[Waal and Gaag, 2007] Waal, P. R. and Gaag, L. C. (2007). Inference and learning in multidimensional bayesian network classifiers. In *Proceedings of the 9th European Conference on Symbolic and Quantitative Approaches to Reasoning with Uncertainty (ECSQARU)*, pages 501–511. Springer-Verlag.

[Wang et al., 2007] Wang, P., Hu, J., Zeng, H.-J., Chen, L., and Chen, Z. (2007). Improving Text Classification by Using Encyclopedia Knowledge. In *Proceedings of the 7th IEEE International Conference on Data Mining (ICDM)*, pages 332–341. IEEE Computer Society.

[Webber and Pickens, 2013] Webber, W. and Pickens, J. (2013). Assessor disagreement and text classifier accuracy. In *Proceedings of the 36th international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '13, pages 929–932, New York, NY, USA. ACM.

[Wettschereck and Dietterich, 1994] Wettschereck, D. and Dietterich, T. (1994). Locally adaptive nearest neighbor algorithms. *Advances in Neural Information Processing Systems (ANIPS)*, 6:184–191.

[Wolpert, 1992] Wolpert, D. H. (1992). Stacked generalization. *Neural networks*, 5(2):241–259.

[Xiaoyue and Rujiang, 2009] Xiaoyue, W. and Rujiang, B. (2009). Applying rdf ontologies to improve text classification. *Computational Intelligence and Natural Computing, International Conference on*, 2:118–121.

[Yan et al., 2003] Yan, R., Yang, J., and Hauptmann, A. (2003). Automatically Labeling Video Data Using Multi-class Active Learning. In *Proceedings of the 9th IEEE International Conference on Computer Vision (ICCV)*, volume 2, pages 516–523, Washington, DC, USA. IEEE Computer Society.

[Yang et al., 2009] Yang, B., Sun, J.-T., Wang, T., and Chen, Z. (2009). Effective multi-label active learning for text classification. In *Proccedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 917–926. ACM.

[Yang, 2001] Yang, Y. (2001). A Study on Thresholding Strategies for Text Categorization. In *Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 137–145. ACM.

[Yang et al., 2000] Yang, Y., Ault, T., Pierce, T., and Lattimer, C. W. (2000). Improving text categorization methods for event tracking. In *Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '00, pages 65–72. ACM.

[Yang and Liu, 1999] Yang, Y. and Liu, X. (1999). A re-examination of text categorization methods. In *Proceedings of the 22nd International Conference on Research and Development in Information Retrieval (SIGIR)*, pages 42–49. ACM.

[Yang and Pedersen, 1997] Yang, Y. and Pedersen, J. O. (1997). A comparative study on feature selection in text categorization. In *Proceedings of the 14th International Conference on Machine Learning (ICML)*, pages 412–420. Morgan Kaufmann.

[Yom-Tov et al., 2005] Yom-Tov, E., Fine, S., Carmel, D., and Darlow, A. (2005). Learning to estimate query difficulty: including applications to missing content detection and distributed information retrieval. In *Proceedings of the 28th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, pages 512–519. ACM.

[Zaragoza et al., 2011] Zaragoza, J. H., Sucar, L. E., Morales, E. F., Bielza, C., and Larrañaga, P. (2011). Bayesian Chain Classifiers for Multidimensional Classification. In *Proceedings of the 22th International Joint Conference on Artificial Intelligence (IJCAI)*, pages 2192–2197. IJCAI/AAAI.

[Zelikovitz and Hirsh, 2002] Zelikovitz, S. and Hirsh, H. (2002). Integrating background knowledge into nearest-neighbor text classification. In *Proceedings of the European Conferences on Case-Based Reasoning (ECCBR)*, page 5. Springer.

[Zhou and Croft, 2007] Zhou, Y. and Croft, W. B. (2007). Query performance prediction in web search environments. In *Proceedings of the 30th International ACM SIGIR conference* *on Research and Development in Information Retrieval (SIGIR)*, SIGIR '07, pages 543–550. ACM.

[Zhou and Liu, 2010] Zhou, Z.-H. and Liu, X.-Y. (2010). On multi-class cost-sensitive learning. *Computational Intelligence*, 26(3):232–257.

Index