Full Research Proposal

-

Supervised On-line Hierarchical Multi-Label Text Categorization with the Application Adaptive and Active Learning

Jesse Read

July 4, 2007

Contents

1 Introduction **4**

2 Problem Definition **6**

3 Research Goals **11**

4 Prior Work **11**

4.1 Multi-label Categorization 11

4.2 Hierarchical Classification 15

4.3 Comparing Local Versus Global Hierarchical Methods 20

4.4 Preprocessing 22

4.5 Active Learning 23

4.6 On-line Adaptive Learning 24

5 Related Work **26**

6 Completed Work **26**

6.1 Datasets 27

6.2 Algorithm Selection 30

6.3 Evaluation Metrics 30

6.4 Results 32

6.4.1 Standard Problem Transformation Methods - Flat Multi-label Classification(MC) 32

6.4.2 Global Hierarchical Multi-label Classification (GHMC) 32

6.4.3 The Value of Labeled Examples 34

6.4.4 Accuracy Over Time 34

6.5 Conclusions 34

7 Plans for Future Work **42**

8 Schedule **43**

8.1 Completed Work 43

8.2 Scheduled Work 44

9 Thesis Outline **46**

10 Resource Requirements **46**

11 Ethical Consent **47**

1 Introduction

It has always been a part human nature to collect data and store information, which in turn begets an urge to categorise that data so that it can be retrieved easily at a later date. Through much of human history there was relatively little information to categorise, and any categorisation task could be completed by hand, now with the introduction of computers and particularly the world wide web, there is a rapidly increasing amount of data. Much of this data is in the form of text data which is on-line in the sense that the current data is being constantly fed by a stream of fresh documents, as opposed to static collections. Emails, web pages, news articles, academic papers, and browser bookmarks are only some examples of textual data which people spend time categorising new documents in an on-line fashion on a daily basis.

Multi-label categorisation, recently also termed "tagging", involves organising documents into a number of pre-defined categories by assigning them labels. Current popular examples of such systems include Gmail[1], Google-Bookmarks[2], del.icio.us[3], CiteULike[4], Wikipedia[5], any sites featuring current news or events as well as applications like medical text classification [46]. In all these current examples, which are specific to on-line text categorisation, new documents are either all categorised manually, or in some cases aided by the manual construction of a set of filtering rules (such as emails in Gmail).

[1]http://www.gmail.com
[2]http://www.google.com/bookmarks/
[3]http://del.icio.us/
[4]http://www.citeulike.org
[5]http://www.wikipedia.org

Manual categorisation is not only slow and labour intensive but inadequate for large rapidly expanding collections of documents. While large web communities known as "folkonomies", distribute the labour of categorising new documents over many users, they are criticised for the unreliability and inconsistency of labelling [11]. Filtering rules are an option for the private labelling of a single, which can label many documents instantly, however rules are often ineffective, and even good filtering logic is quickly made redundant over time as an incoming stream of data begins to evolve [9]. Additionally, the boolean logic and regular expression syntax for the construction of such rules is often beyond the grasp of regular users.

Hierarchical multi-label classification is an extension of the multi-label problem. Currently, widely used multi-label schemes function in a flat sense, where all categories are considered equally independent of each other, but in practice most information naturally tends to form of hierarchic structure, as can be seen to be in place in any website or computer file system.

This research will approach the task of on-line hierarchical multi-label categorisation from a machine learning perspective. While the user will still be in complete control of the predefined set of categories, and any hierarchy it forms, as well as the documents themselves, the task of classification will be undertaken by a machine learning framework. Via supervision/interference from a user or users, such a framework will be able to learn to categorise new documents automatically and improve its accuracy ever more over time.

2 Problem Definition

Multi-label Classification. Multi-label classification is essentially an generalisation of the more common task of multi-class, or "single-label" classification which assigns each document one label from a label set of possible labels, $l \in L$, where $|L| > 1$. The task of multi-label classification on the other hand, is to assign to a document a subset of labels $S \subseteq L$. Traditionally, works assume $|S| \geq 1$ [44, 18, 52, 21, 25], yet because of the on-line and adaptive learning implications of this research, we consider the possibility of $|S| = 0$ to indicate the possible presence of a new topic or the request for manual classification. The many different possible multi-label combinations ($2^{|L|}$) complicate the selection and design of algorithms for the task over a standard multi-class problem [13, 5].

Hierarchical Multi-label Classification The basis of hierarchical multi-label categorisation is simply multi-label categorisation with the inclusion of some method for taking into account relationships between labels – as opposed to the assumption of a "flat" classifier which is that all labels are independent of each other. For example, to a flat multi-label classifier the three categories `sports->rugby`, `sports->soccer` and `politics->us` are treated as three totally distinct categories whereas a hierarchical classifier would take into account that there is some relationship between `soccer` and `rugby`.

The structuring of text documents into a topic hierarchy, known as a **taxonomy**, will be the focus of this research. This type of hierarchy is composed of many parent-child relationships, taking the form of a **tree** with

$|L|$ leaf nodes and only one possible path to each leaf node, which should create an intuitive structure any human familiar with the domain knowledge.

Hierarchical classification approaches generally divides into **global hierarchical** and **local hierarchical** methods (though it should be noted that it is also possible to use a flat multi-label classification scheme with hierarchical data).

Global hierarchical classification involves the construction of a single multi-label classifier which has functionality to take into account the predetermined hierarchical relationships between classes. The classification process involves a once-off consideration all possible leaf categories as potential labels for a document.

Local hierarchical classification on the other hand involves multiple multi-label classifiers arranged in a form mirroring the hierarchic structure which was manually defined in the dataset. Beginning at the root, classifications are made at internal node, each label leading to another node, and so on down the hierarchy and ending with classification at a number of leaf nodes. Figure 1 illustrates the distinction between global and local classification.

Figure 1: Global and Local Classification

In the most common local approach known as a "pachinko machine", only branches are considered where the presence of that label was detected. An alternative is a probabilistic classifier which considers all possible paths in the hierarchy and the most probable paths become the final classification decision.

Feature Selection A distinction also exists between global and local hierarchical feature selection, compared in figure 2.

Figure 2: Global and Local Feature Selection

Global feature selection involves a single filtering process for extracting attributes from the dataset, which can be used either to train a global classifier, or a hierarchy of local classifiers.

Local feature selection, comparably to local classification, makes use of the hierarchical structure present, and a filtering process takes place at each internal node to extract attributes, meaning that word features are more general near the root, growing more specific nearer the leaf nodes. The terms at each node can be used to train a node classifier (local classification) or bagged together and used to train a single global classifier.

On-line Classification An on-line classification context entails that the set of documents D will grow and change as it is fed by new documents over time. The focus here is not the strict sense of streaming where only a current window of w documents or even w words are available at a time, but rather, a dynamic pool, or collection of documents. Email accounts, file-systems and databases are some examples of dynamic document collections. Over time new documents are expected to enter the document pool and documents may loose their relevancy over time as the incoming stream of documents continues to evolve, implying that not only the set of multi-labeled documents D changes over time, but also that the set of labels L and the document-label combinations they form. When evaluating such a framework,documents

cannot be taken out of their time context and must be treated as a sequence, and not just as a random occurrences.

Supervised Learning Supervised learning implies a set of manually labeled and manually structured examples to made available to a classifier, but because this is an on-line problem, the classification process is not done in a one-off fashion, but runs over time. This means that manual labelling and relabelling of documents (which may imply changes to the hierarchical structure) is expected to take place after the classification process begins – in an on-line problem, there is never a "final" classification. In short, the user or users of the system are granted full control over all the documents in the collection, the hierarchic structure they form implied by the labels which they are assigned.

Such supervision is readily available in real world scenarios, particularly in applications such as a user's email or personal file or bookmark collection where people are able to form their own personalised structure, but also in evolving web communities, such as Wikipedia and CiteULike. This will have a significant effect on the path of the research by obliging efficient and adaptive classifiers which are able to work with a variety of document sets and labelling schemes, and deal with significant and unpredictable changes made to their structure.

Adaptive Learning Adaptive learning techniques are essential to on-line context, in this case implying a classifier which can adapt to a continuously changing collection of documents, and the set of possible labels which each

document can be assigned. By incorporating new information and phasing out information when it is no longer relevant a classification framework can adapt to changes implied by the stream of new documents.

Labels themselves can be deprecated when they are inactive for some time such as stale threads or topics, meaning they remain assigned to existing documents, but the information learned about them can be phased out over time, thereby freeing up memory and decreasing the computation complexity associated with retaining the information.

Active Learning Active learning is be defined as "any form of learning whereby the learning algorithm has some degree of control over the inputs on which it is trained" [**?**, 29]. This component will be necessary in any developed framework to minimise the need for interaction by increasing the value of labeled examples – only requesting a user interaction when it will be most valuable. Such requests are most likely to stem from examples of undetermined classification ($|S| = 0$), which would lead to the establishment of new labels, which the user specifies. In this sense – topic detection – although the actual process of topic creation (*i.e.* a new label) at a location in the hierarchy is done by the user. Ideally, the failure to classify a document into the existing structure and the request of a new label for it by the framework should only coincide with the emergence of a new topic or the division of an existing one.

3 Research Goals

A framework for hierarchical multi-label classification to function in an online context. The framework will ideally meet the following requirements:

- low computational requirements, suitable for use on an average modern desktop machine;
- either automatic, or intuitive parameter tuning where no machine learning knowledge is required;
- flexible for use with a variety of text collections and labelling schemes;
- able to perform efficiently in an on-line context and deal with time-evolving data, adapting to new documents and also manual changes to the hierarchic structure;
- minimise the need for human-labeled examples;
- accurate and able to outperform existing approaches and prove viable for practical application over existing methods.

4 Prior Work

4.1 Multi-label Categorization

Multi-class single-label problems have been widely approached by the machine learning community for some time [14], and commonly used algorithms support multi-class datasets, either inherently like naïve Bayes, or via transformation into multiple binary problem for each class such as with SVMs.

However there has been relatively little work approaching the multi-label problem [3, 14, 52], since initial works like that of Dumais and Chen [15] of the late 1990s. This lack of attention is also evident in the fact that widely used machine learning tools such as WEKA [23] do not have built-in support for multi-label classification tasks. Also of note is that much of the current work in multi-label classification is specific to bioinformatics [8, **?**, 25, 3, 2] where data and hierarchic structure are formed differently than text data, and the algorithms which work best on such data are accordingly different [25].

Problem Transformation The most common approach to multi-label categorisation is simply the transformation of a multi-label problem into one or more multi-class problems prior to classification, allowing the use of regular multi-class classifiers. The multi-class classification output is then transformed, via the process in reverse, back to reflect a multi-label representation. Such methods, known as **problem transformation** or **data transformation** form the basis of virtually all more advanced multi-label classification problems, either externally utilising multi-class algorithms, or via the internal modification of an existing multi-class algorithm.

Primitive Transformation The most basic two transformation methods from multi-label to multi-class are one-way only, taking place prior to standard multi-class training. Simply, either all multi-labeled documents are removed from the dataset, or alternatively, all but one label from each multi-labeled document are randomly deleted. In both cases this forms a regular multi-class problem, but all multi-label information is lost, and performance

suffers accordingly [46]. Both these methods can be safely discarded from further analysis.

The Label Combination Method (LC) One of the most commonly used methods of transforming a problem, the Label Combination (LC) method, combines each document's label subset $S \subseteq L$ to form a single label for that document. The result is a regular multi-class/single-label problem suitable for any standard multi-class-capable classifier. For example, if a document is originally multi-labeled with the set $S = a, c, f$, under this method it would be reassigned a single label $l = acf$. The disadvantage to this method is the potential to generate too many class labels from only a few examples – a possible maximum of $2^{|L|}$ – depending on the dataset. It may also be too inflexible for non-randomised time based data, being unable to take into account new label combinations that may be found in the test data.

The Multiple Binary Classifier Method (MBC) Another very widely documented problem transformation approach is the Multiple Binary Classifier (MBC) method , where $|L|$ binary classifiers are built – one for each possible label category. Each binary classifier is responsible for predicting the association of a single label $l_i \in 0, 1$, 1 indicating that the label is assigned. This is done by training each classifier in a "one-vs-rest" fashion. The first binary classifier B_1 is trained with all documents that are assigned label l_1 ($l_1 \in S$) versus all the documents where $l_1 \notin S$, and the classifier B_2 with documents of label l_2 against $l \notin S$ and so on to l_n, for a total of $|L|$ classifiers. To classify a new unlabeled document, each classifier will

independently make a 0/1 prediction for the label it was trained for. The disadvantage of this method is that it fails to take into account any possible relationships between labels, and the process may run significantly slower than other methods due to the number of classifiers involved, depending on $|L|$. A similar alternative is the "all-vs-all" [4] approach but the number of classifiers which this method entails rises exponentially with $|L|$.

The Ranking Threshold Method (RT) A lesser documented technique, the Ranking Threshold (RT) method coming from the task of multi-label "ranking" advantages from classifiers like Naïve Bayes which can easily output an ordered certainty distribution representing their confidence $\lambda_1, \cdots, \lambda_{|L|}$, in other words the posterior probabilities of the labels $P(l_j|d_i)$. Such classifiers provide us with another possible transformation method RT. A straightforward way to train the classifier is to duplicate each document $|S| - 1$ times providing $|S|$ copies, each of which is assigned just one label $l \in S$, thus producing a regular single-label multi-class dataset, albeit one with more documents now in the training set. Any regular multi-class classifier can then be trained and many will be able to produce a ranked certainty distribution of L for any test instance. To turn this ranking into a multi-label classification, some method must be involved to select the label classification subset $S' \subseteq L$. Most commonly, all elements of L which fall above a threshold T become the label classification S'. The disadvantage of this method is the difficulty of selecting an ideal T, which varies from dataset to dataset. Also, like the MBC method, it is truly flat, assuming that all labels are independent, therefore failing to benefit from any relationships between labels

which are present in the dataset.

4.2 Hierarchical Classification

Currently there is a significant interest in hierarchical classification with multi-class problems [24, 12, 39, 48, **?**, 28], much of it also with on-line related applications such as Email, but unfortunately these works are not directly applicable to multi-label problems [**?**, 7]. Although it is true that many Email applications presume single category organisation (most often represented by folders), in many real situations it is often desirable to apply more than one label to a document (*i.e.* cross-indexing), evident by the introduction of the modern multi-labelling ability of Google's Gmail.

Of the relatively little research on multi-labeled data, even fewer works have focused on hierarchically structured data [44] (the example of Gmail just given is in fact a flat multi-label system). Tsoumakas and Katakis [46], who did the first substantial general overview of the subject, ignore the concept of hierarchy altogether and work only with flat datasets, and flatten out any hierarchically organised datasets which they use. This is surprising, as a hierarchical framework stands to show superiority on any dataset which does have hierarchical structure present; the relationships and similarities between labels are not only visually intuitive but can potentially be taken advantage of by a classifier to boost accuracy and reduce computational complexity [25, 26, 44, 28].

Expectation Maximisation Several global hierarchical works use a semi-supervised learning approach with Expectation Maximisation (EM) which

utilises data from unlabeled examples to increase classification accuracy [36, 35, 13, 32]. McCallum et al. [32] successfully implement the EM algorithm with Naïve Bayes on top of the PT-LC method. In theory, variations of EM could also be applied to the other PT methods.

Stacking Variations of stacking have been applied upon to the outputs of the MBC problem transformation method. The underlying rational is to use the inherent ability of stacking to take into account relationships between those classifications, which represent the presence of labels, hence tackling the main disadvantage of MBC – the assumption of label independence – thus making it a global hierarchical method.

Zhang and Zhaou [1] create $|L|$ k-Nearest Neighbor (kNN) algorithms under the MBC method, and for each test instance, they take the k nearest examples, and consider all those with label l as positive, and the rest as negative. They also take into account prior probabilities which making classifications.

Godbole and Sarawagi [21] employ an array of Support Vector Machines (SVM) with a stacking procedure over PT MBC. They use the outputs of an MBC classification to add another $|L|$ features to the datasets, then do a another round of training, thereby taking into account hierarchical relations between the labels. They also implement minor SVM-specific modifications. We reimplemented their stacking procedure generally, see **??**.

Ghamrawi and McCallum [18] also use an additional set of features with PT MBC in order to recognise dependencies between labels, by utilising a form of stacking.

It should also be possible to stack the outputs of classifiers running a LC or MBC framework though there does not appear to be prior work attempting this.

Ranking There are various ranking methods which expand on the PT RT method, with various ways to turn a multi-label ranking into a multi-label classification. Essentially the task is to determine the set size of S which determines how many labels are elected from L to become the actual classification for any particular document.

Luo and Zincir-Heywood [30] developed a kNN-based classifier, using novel pre-processing techniques for representing the relationships between labels. Dealing with $|L|$ possible categories, they classify each test document with the top $|S|$ ranked labels. Tsoumakas and Katakis [46], criticise their method as not being practical for real world use where $|S|$ cannot be known for each test document. This would be especially true in an on-line learning scenario – $|S|$ must be determined by the classifier as well as the elements of S.

Elisseff and Weston [16] apply a ranking approach to SVMs. They try to predict the set size $|S|$ for each test document by using the MBC method for inspiration. They use ranking loss as the cost function, being the percentage of label pairs which are ordered incorrectly.

Brinker, Fürnkranz and Küllermeier [6] provide significant work with turning ranking into multi-label classification by introducing the concept of calibrated ranking, which they describe as using a virtual label as a 0 split point to separate the applicable top labels S out from the full label set L,

for each document.

Vilar *et al.* [47] work with Naïve Bayes Multinomial and (-..@todo..-), using a threshold to split set L to create $S \subseteq L$. They also note that a similar thresholding approach can be applied to the MBC method, though their application to RT showed better results.

Boosting The well known AdaBoost.MH and AdaBoost.MR algorithms [17, 40, 25, 26] are based upon modifications to the Adaboost algorithm, designed for multi-class multi-label classification. The base of these algorithms is a different problem transformation method, which involves making $|L|$ copies of each document, each given a new attribute a_i, to have either the value 1 if $l_i \in S$ or -1 otherwise. A boosting and re-weighting process then takes place. At classification time, AdaBoost.MH outputs a multi-label classification, while the AdaBoost.MR version just outputs a multi-label ranking.

Without any special efforts to represent the copies more efficiently, the initial transformation will create a dataset $|L|$ times greater in size than the original, certainly one of the downsides of the algorithm as far as textual data is concerned. It appears to have mainly been focused on for bioinformatics applications, and though it can be given textual data, it can fail to perform well on sparse data [17], which text invariably is as a numerical representation. The results of Elisseff and Weston (see 4.2 above) claim to be superior to the multi-label AdaBoost family.

Association Rules Like filtering rules, single association rules can also support multiple actions, such as assigning multi-labels. Thabtah *et al.* [**?**]

create MMAC, an algorithm which performs multi-label classification by continuing to learn new rules until it has covered all training examples, then merges rules with similar precondition into single rules. The method is comparable to PT-LC where each combination of S is merged into a single label l.

Predictive Clustering Trees A predictive clustering tree (PCT) is a decision tree algorithm which has been modified to support multi-label classification. Most significant is the work by Clare *et al.* [42, 3] who modify the expression for entropy for the C4.5 algorithm to create a PCT. While they mention the benefits of a single-algorithm (global) hierarchical multi-label method, theirs concentrates on rule sets which she uses for gene function prediction, not the bag-of-words method known to text processing, and results are not directly comparable.

Neural Networks have been used instead of decisions trees in a similar fashion to PCTs by Ruiz and Srinivasan [39], but are noted as not being as interpretable and not multi-label [42].

Fuzzy Relational Thesuri (FRT) Fuzzy Relational Thesauri (FRT) are a form of local hierarchical classification, specific to text, where nodes in the taxonomy are represented by a set of terms. Because of this representation, most FRTs tend to be intuitive, as well as adaptive, possibly suiting an incremental context as opposed to batch learning. Although hierarchically based on a taxonomy, FRTs are not necessarily multi-label.

Koller and Sahami, who develop one of the first FRTs [28], claim that 10

– 100 words per hierarchy node can produce good results and they show their classifier's success in use with the Reuters dataset. Wilbowo and Williams also claim that a few features can work well [49]. They both note that a small term set encourages very fast performance.

Tikk *et al.* [43], who also create an FRT framework, criticise the reduced word set approach of Koller and Sahami. They play heavily with term weightings and claim to outperform existing approaches by up to 10%.

Shrinkage McCallum and colleges [31] also criticise the work of Koller and Sahami, claiming that larger vocabulary sizes improve accuracy and that local hierarchical pachinko machine approaches actually gain similar accuracy to plain flat classification. They employ **shrinkage** in global classification, but use local feature selection, to add more words to sparse children, based on the words of their ancestors. The main idea being to improve performance when there are few manually labeled examples.

Maximum Margin HMC Rousu et al. implemented what they called "Maximum Margin Hierarchical Multi-label Classification" [38] with a Markov network and compared it to SVM but admitted very slight accuracy gains and large CPU requirements.

4.3 Comparing Local Versus Global Hierarchical Methods

Advantageously local hierarchical classification lends itself toward greater computational efficiency as only a few nodes of any tree need to be visited,

and it is much more flexible – structural changes affect relatively smaller numbers of data to be retrained on – depending on how deep in the tree the affected node is. The local hierarchy of classifiers is therefore especially suited to a supervised on-line learning problem where the hierarchy can be rearranged and new incoming data are expected regularly and computation efficiency when managing such changes is very important, if not paramount.

Despite losing intuition and computational efficiency when using a global approach in preference to a local approach, S. Kirchenko *et al.* claim that in many real-life applications, the global approach can be easier to maintain and interpret as well as potentially demonstrate higher accuracy [26]. They work with the AdaBoost family and conclude that AdaBoost works better in a global hierarchical sense, than in a flat or local hierarchical arrangement. The global approach certainly has been implemented in several text categorisation systems [25].

The claims of seeing little or no improvement in accuracy within a local hierarchical versus flat approach, is perhaps explained by the fact that the hierarchical structure found in a taxonomy is for the browsing and searching value of the user ([**?**]), not necessarily for the value of the algorithm. However, there have been no claims against the computation efficiency of local hierarchical classification.

Pachinko machines are usually chosen over probabilistic methods due to their faster computational performance, yet probabilistic methods can expect potentially higher accuracy by avoiding the well-recognised downside of pachinko machines – that of making mistakes earlier on that lead to premature error on initial branches [25]. As a mid ground between the two, work

has been done on error recovery for the pachinko machines such as an error-correcting hierarchical method documented by Wibowo and Williams [50], Error-Correcting Output Codes (ECOC) [19] and the possibility to return to higher nodes[10]. These additions to the standard pachinko machine, come, of course, at some computational cost.

Godbole, Sarawagi and Chakrabarti [20] use a local approach where Naïve Bayes is used primarily, for speed, and SVM classifications are used as a kind of "backup" if the Bayes classifiers lack confidence in their prediction.

In rare implementations of the pachinko machine, internal nodes have been considered as final classifications [**?**, 7, 25].

4.4 Preprocessing

In text classification at least some pre-processing is needed on the data in the form of feature selection, to encode word information in numeric format. This is usually in the form of a filter which chooses the most relevant terms without consideration for the classifier to be used. Terms are usually bundled together in a "bag of words" approach, which has proven to be just as accurate and much more efficient than other techniques such as encoding bi-grams and trigrams [**?**]. There have been many approaches used in multi-label learning for selecting word features, the most common [25, 28, 34] being:

- Binary presence of terms;
- TFIDF;
- information gain; and

- cross entropy.

A wrapper approach – which selects features based on classification – is generally not considered because it is much slower, making it inadequate for large corpora, or an on-line adaptive algorithm [25, 34].

Pachinko machines in particular depend heavily on intelligent feature selection for limiting the mistakes made by classifiers at each node [28, 49], though as we have seen, approaches to it vary considerably.

Additionally to feature selection, there have been several other pre-processing strategies, including attempts to use a pre-processing step to encode the relationships between labels or terms [30, **?**, **?**]

Bade *et al.* [**?**] exploit the hierarchy only as a post-processing step. They work with a unique dataset "banksearch" which does not appear to be a true topic hierarchy, yet rather a collection of 4 multi-class datasets joined together by a root node. This is perhaps very atypical of a multi-label dataset, which are likely to have second-level nodes that have much in common.

Wibowo and Williams [49] proposed a using global feature selection with a local classification algorithm, the opposite of McCallum *et al.* [31] as well as [48], once again it is evident that in prior works local and global classification is distinctive to local and global feature selection.

4.5 Active Learning

Active learning has been widely employed and shown to greatly improve learning in text categorisation tasks [37, 41, 29, 45] and has been used in combination with many algorithms, including Naïve Bayes EM [33] and

SVMs [45, 41]. Klaus Brinker uses active learning for multi-label classification [5], as have Li, Wang and Sung [**?**] – although their work is for the purpose of image classification. [37] claims to be one of the few works that have worked in an on-line or stream based setting with active learning.

4.6 On-line Adaptive Learning

Some multi-label approaches, especially the local hierarchical FRT variety are designed to be very efficient and to cope incrementally with large datasets which cannot be held in memory all at once, like the early work of Tikk and Biró [14] who work with a 75,000 document patent set. However efficiency and incremental learning, while necessary for an on-line context, does not necessarily beget adaptability and many efficient algorithms use techniques which would not be possible in a data stream such as assuming to know the amount of training documents which will be present. Traditionally, most prior research has assumed a static pool of documents to work with [22].

Hohman and Marchette [22] present an on-line graph model system for news documents, they work with a streaming context whilst retaining statistics about previously seen documents and constantly re-weighting terms with new documents. Their preprocessing is done in a windowed setting, as a constantly moving average. However their method is a clustering system, and they calculate document similarity, instead of classifying them with a predetermined set of labels.

S. Zhong also points out that "one needs to forget in order to learn" and uses exponential decay to phase out information over time in his work with

clustering streams of data [51].

Kleinberg in his work [27] has approached the idea of modelling the emergence and disappearance of themes or topics in on-line text document streams over time, which he calls "bursty" streams, and he also detects hierarchical structure within those bursts. Some elements of detecting bursts of activity – especially in a hierarchical sense – is highly relevant, although modelling them is much closer to clustering than classification.

Cesa-Bianchia *et al.* creates an incremental multi-label local-hierarchical probabilistic-variety classifier. This classifier works in a true on-line setting (adjustments are made after each time step t). The work introduces internal-node labellings, and an H-Loss (hierarchical loss) function for evaluating the classifier. His adaptive approach stands out as being classification (only generating multi-labels that respect the underlying hierarchy of the dataset), as opposed to clustering. He also appears to consider the case where $S = \emptyset$ ($|S| = 0$).

5 Related Work

There are also applications which are related but do not fall under the focus of this research. This includes clustering, and other forms of unsupervised learning, which are occasionally employed to work in a multi-labelling fashion data, but involve data with no previously known structure and groups of documents are created which by no means necessarily represent a structure useful to a human[**?**], therefore not being compatible with the fully supervised goals of the research described in section **??**.

Labels may be very different even within the same example. Labels could be very obvious to distinguish (e.g. a Language Category: English, Spanish or French) or much more subtle (e.g. Machine Learning Categories: Semi-Supervised Learning, Active Learning). These differences may even appear in different levels of the same hierarchy.

6 Completed Work

To facilitate the re-implementation of prior work, a Java framework was constructed, based upon the WEKA machine learning toolkit [23] to handle multi-label and hierarchical multi-label data. Credit is to be given to Tsoumakas and Katakis [46] who also developed a similar extension for working with flat multi-label datasets, and I have based my implementation of PT3 and PT4 and evaluation functionality upon some of their Java code[6].

6.1 Datasets

The Reuters 21578 and 20 Newsgroups datasets are the only widely referenced hierarchical multi-label text-based datasets. A variety of multi-label datasets were put together to reflect the many different possible applications of multi-label algorithms. Table 6.1 lists the datasets. The following paragraphs will explain the sources and the measurements given in the table.

The YEAST dataset is a widely used biological data set, which we use as a comparison against textual data. Unlike text, it's original format is numeric data, and not sparse.

[6]Available from http://www.

MEDC is textual multi-label data with no specified hierarchy. A very brief free text summary of patient symptom history and future reccommendation are used to predict insurance codes.

20NG is the classic 20 Newsgroups dataset, but ordered strictly by date, and a small changes to the hierarchy (for the sake of intuition, for example combining branch nodes `soc.religion.` and `talk.religion.`).

The ENRN dataset is a subset of the Enron Email Corpus [7], labeled with a hierarchical set of categories developed by the UC Berkeley Enron Email Analysis Project. Top level labels are 1 *Coarse genre*,2 *Included/forwarded information*, 3 *Primary topics* and 4 *Emotional tone*. Each message was labeled by two people, but no claims of consistency, comprehensiveness. The set was organised by date.

MARX is essentially a website mirror of http://www.marxist.com. It mainly consists of news and other left-wing articles, organised by language, continent and country, and other categories such as *Sci/Tech*,*Globalisation*, and *Economy* as well. The title of each page and a maximum of three paragraphs were taken from each article, as some are very lengthy. The set was then sorted by date. Articles in other languages represent a small percentage of the data, only those not based on the Latin character set were excluded.

REUT is a subset of the modern high quality Reuters RCV1 corpus, used here in place of the older 21578 set. The 'Topics' hierarchy is used. This data comes in numeric form, of which we reduce the feature set from approximately 46,000 to approximately 1000, comparable to the other datasets used (more on this shortly). The set involves internal node classifications, at which only

[7] http://www.cs.cmu.edu/ enron/

a few efforts have been made by hierarchical multi-label classification [25]. We deal with the internal-node classification task by simply creating an extra leaf node in each case.

Web hierarchies such as those of YAHOO![8] and Google[9] were avoided because the structure of pages of links offer far too much potential interference from HTML or Advertisements such as occurred in the work of Hohman and Marchette [22]. Also, such hierarchies are mainly intended to be multi-class and any multi-labeled, too hard to detect for the reason mentioned of potential interference.

Note that three of the corpora (20NG,ENRN, and MARX) represent online data as they are strictly ordered by their documents' dates, which span several years. This implies that the data will not be randomised during experiments as it is unreasonable for an algorithm to expect to be able to deal train on subject matter of a subject which has not yet begun. The Reuters data is also used in a non-randomised train and test split due to its size and the form it came in. MEDC, also the only non-hierarchical set, is trained via a 10 fold randomise train/test split.

Our reimplementation of the local hierarchical approach uses a simple, yet apparently undocumented method of creating extra leaf nodes to avoid the need for special functionality for internal classifications.

Errors in the datasets are not concerning, as long as they do not bias the data, by date for example. The application of this research targets users and non-perfect categorisation.

[8]http://dir.yahoo.com/
[9]http://directory.google.com

Key	$\|D\|$	$\|L\|$	Split	$LD(D)$	$LC(D)$	CD	Seq.	Text
YEAST[10]	2,417	14	62/38	0.30	4.24	198	N	N
MEDC[11]	978	45	60/40	0.03	1.25	94	N	Y
20NG[12]	19,300	20	50/50	0.05	1.03	55	Y	Y
ENRN[13]	1,702	53	60/40	0.06	3.38	753	Y	Y
MARX[14]	3,617	101	60/40	0.01	1.13	208	Y	Y
REUT[15]	6,000	103	50/50	0.01	1.46	811	N	Y

Table 1: Datasets

Apart from the number of documents $|D|$ and labels $|L|$, datasets are also measured here by label density and label cardinality. The label cardinality of a set D, where $(x_i, S_i), i = 1..|D|$, is defined as:

$$LC(D) = \frac{1}{|D|} \sum_{i=1}^{|D|} |S_i| \tag{1}$$

Note that generally, textual data has a much lower LC as each document is only likely to fit into a few categories of a taxonomy. The ENRN set is an exception due to its atypical hierarchy.

6.2 Algorithm Selection

The six multi-class capable algorithms most commonly used in previous multi-label work can be found listed in table 6.2.

Key	Type	Description
NB	Bayes	Naïve Bayes
BAG.	Meta	Bagging (with J48)
SMO	Function	Support Vector Machines
J48	Tree	J48
IBk	kNN	k Nearest Neighbor
NN	Neural	Neural Networks

Table 2: Algorithms

A pilot experiment approximated the effectiveness of the algorithms on textual vs biological data; the MEDI and YEAST datasets were used, for this comparison. Figure 3 displays the results of those algorithms employed by each of the three main problem transformation methods, clearly showing the inferior accuracies of IBk with textual data. The NN algorithm also performs relatively poorly, and while accuracies are comparable to other algorithms in some cases, the notably slower build time stands out in table 6.2. These factors allow us justification to discard these algorithms from the following experiments.

Figure 3: Classifier Accuracy for YEAST Dataset vs MEDI dataset. 10x 60/40 Train/Test Split

	MEDI			YEAST		
Dataset	PT3	PT4	PT5	PT3	PT4	PT5
NB	2	41	11	1	1	1
SMO	42	10	31	97	97	97
IBk	1	5	10	174	174	48
BAG	43	153	40	25	25	35
J48	17	74	21	0	0	1
NN	314	5,112	257	1,919	1,919	1,357

Table 3: Classifier Build Times for YEAST Dataset vs MEDI dataset. 10x 60/40 Train/Test Split

6.3 Evaluation Metrics

There is some debate about evaluation measures for multi-label classification, specifically hierarchical classification. Multi-label classification requires different evaluation methods than traditional multi-class classification; evaluating by correct instances overly penalises classifiers, while counting correct

labels is overly lenient. This is especially the case when involving datasets with large label sets and low label cardinality – where the implication would be that setting all labels to zero creates excellent results. Let C be a multi-label classifier, $S \subseteq L$ and $Z_i = C(x_i)$ be label predictions by C for document x_i. Accuracy for multi-label data [46, 21] can be defined as:

$$Accuracy(C, D) = \frac{1}{|D|} \sum_{i=1}^{|D|} \frac{|S_i \cap Z_i|}{|S_i \cup Z_i|} \tag{2}$$

Also commonly used is the hamming loss [40], which represents the percentage of incorrectly assigned labels. Precision and recall are also other measures [47]. All have been implemented yet we use accuracy because it suffices for comparing a variety of algorithms and datasets and tends to correlate to the other measures anyway.

Svetlana [25] and Cesa-Bianchi *et al.* who both allow for internal node classifications, claim that partial credit should be given to partially correct hierarchical classified documents, that is to say documents classified with some correct branches but incorrect leaf nodes. However the primary goal of the preliminary work for this thesis is interested in the direct comparison of the performance of hierarchical methods versus flat methods for which using an identical evaluation measure is optimal.

While in practice, adaptive classification and active learning involves human interaction, we can simulate them by training in steps on a strictly time-ordered datasets where the training set-size is progressively increased as the test set size remains the same. The cumulative number of incorrect labels assigned represents the degree of human interaction required.

6.4 Results

6.4.1 Standard Problem Transformation Methods - Flat Multi-label Classification(MC)

The three common problem transformation methods PT3[16], PT4 and PT5. Table

Represented in weka a vector $v = (v_i, ..., v_N) \in 0, 1^N$ where $i \in 1, ..., N$ belongs to the multilabel of x if and only if $v_i = 1$.

for a graph: $P(n_4|n_3, x)$ (n are nodes)

Accuracy (%)

	PT3				PT4				PT5			
Dataset	NB	BAG	SMO	J48	NB	BAG	SMO	J48	NB	BAG	SMO	J48
MEDI	68.05	71.77	71.10	72.13	55.82	75.58	73.59	65.83	67.81	64.20	65.72	60.22
20NG	57.47	57.58	57.35	52.74	32.33	-	47.67	41.09	56.05	47.19	54.61	50.55
ENRN	32.72	25.42	-	22.96	21.82	31.35	30.56	26.26	15.16	30.25	24.09	27.82
MARX	48.15	48.93	43.26	44.79	32.6	31.69	38.64	33.95	48.44	36.07	40.46	38.71
REUT	43.76	51.47	-	41.68	18.21	44.09	56.23	43.83	37.13	45.9	58.65	45.31

Time (s)

	PT3				PT4				PT5			
Dataset	NB	BAG	SMO	J48	NB	BAG	SMO	J48	NB	BAG	SMO	J48
MEDI	2	43	42	17	41	153	10	74	11	40	31	21
20NG	26.03	14554.3	169.45	1754.06	530.21	-	815.72	18249.87	69.18	7496.08	442.11	849.1
ENRN	3.5	913.78	0.08	125.13	1372.82	118.09	651.42	42.24	1915.37	682.34	234.61	
MARX	10.27	2543.34	299.11	211.06	1149.85	4346.88	182.05	3562.61	22.56	1020.5	291.48	150.38
REUT	6.46	3077.39	0.02	508.39	4204.55	167.15	3955.84	31.46	2057.92	698.16	267.5	

Table 4: Pachinko Machine - GFS

6.4.2 Global Hierarchical Multi-label Classification (GHMC)

PT3-EM represents the work of McCallum *et al.*, which here is implemented, and generalised to work with algorithms other than naïve Bayes.

PT4-Stack the generalised reimplementation of Godbole and Sarawagi's SVM-specific stacking procedure. We use a variety of algorithms at the core

[16]Arguably PT3 is global..

Accuracy (%)

	PT3-EM				PT4-Stack(PT5-NB)				PT6	
Dataset	NB	BAG	SMO	J48	NB	BAG	J48	BAG	J48	
MEDI	67.446	74.705	70.748	72.306	56.094	70.762	73.65	65.854	67.057	67.815
ENRN	34.6	25.46	-	23.31	20.66	31.79	27.01	25.35	-	-
MARX	48.18	50.64	43.29	44.82	39.09	32.08	38.87	34.25	-	-

Time (s)

	PT3-EM				PT4-Stack(PT5-NB)				PT6	
Dataset	NB	BAG	SMO	J48	NB	BAG	J48	BAG	J48	
MEDI	231.212	829.144	979.876	79.363	196.098	259.823	23.629	102.006	5590.016	906.849
ENRN	4614.79	2538.96	-	430.36	327.35	1384.38	134.26	650.58	-	-
MARX	1407.51	7353.49	6089.41	489.12	891.54	2258.01	131.41	1596.74	-	-

Table 5: GHMC

of the algorithm, which is PT4, and use PT5 in combination with Naïve Bayes in the stacking procedure.

PT6-Boost is the implementation of PT6, the method at the core of Adaboost.MR (most of the implementation based on the Java implementation of Tsoumakas and Katakis).

Accuracy (%)

	PT3				PT4				PT5			
Dataset	NB	BAG	SMO	J48	NB	BAG	SMO	J48	NB	BAG	SMO	J48
20NG	52.59	57.28	56.49	51.95	39.87	-	49.75	41.93	51.15	40.48	52.08	48.46
ENRN	21.26	23.4	24.81	20.96	13.73	27.37	23.87	21.49	4.47	19.03	18.63	19.77
MARX	45.1	50.65	44.94	42.82	38.81	37.7	39.6	34.56	43.97	29.33	38.69	41.99
REUT	8.57	49.24	57.55	41.37	34.85	49.26	55.64	40.48	42.96	32.22	45.25	31.21

Time (s)

	PT3				PT4				PT5			
Dataset	NB	BAG	SMO	J48	NB	BAG	SMO	J48	NB	BAG	SMO	J48
20NG	1000	10000	1000	10000	1000	-	10000	10000	1000	10000	1000	10000
ENRN	100	10000	1000	1000	1000	10000	10000	1000	1000	10000	100000	1000
MARX	100	10000	1000	1000	1000	10000	1000	1000	1000	10000	1000	1000
REUT	100	10000	1000	1000	100	1000	1000	1000	100	10000	1000	1000

Table 6: Pachinko Machine - GFS

Global Feature Selection (GFS)

Local Feature Selection (LFS)

Accuracy (%)

	PT3				PT4				PT5			
Dataset	NB	BAG	SMO	J48	NB	BAG	SMO	J48	NB	BAG	SMO	J48
20NG	56.49	58.31	58.83	53.48	43.68	-	52.44	42.03	54.87	40.58	53.37	49.26
ENRN	25.96	29.38	27.73	25.23	15.3	34.99	-	26.26	4.67	25.51	23.59	27.63
MARX	48.49	54.57	42.4	46.84	41.69	38.67	40.34	38.65	46.44	33.59	38.32	41.23

Time (s)

	PT3				PT4				PT5			
Dataset	NB	BAG	SMO	J48	NB	BAG	SMO	J48	NB	BAG	SMO	J48
20NG	444.96	13996.57	1859.72	2023.83	597.19	-	12791.1	7219.44	734.95	13366.3	2405.7	2215.8
ENRN	442.16	1932.06	705.17	603.37	478.82	2265.34	-	657.57	842.75	8655.36	94473.26	1433.52
MARX	115.24	1358.41	486.43	246.04	182.01	1343.64	732.01	666.71	200.98	1878.77	498.72	382.11

Table 7: Pachinko Machine - LFS

6.4.3 The Value of Labeled Examples

6.4.4 Accuracy Over Time

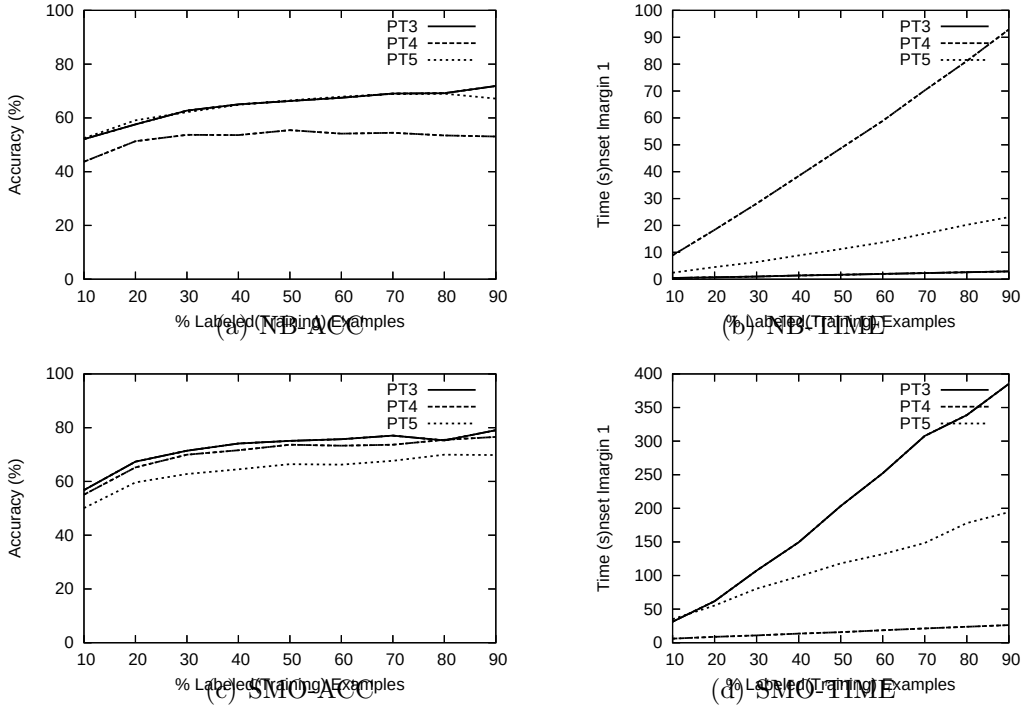
6.5 Conclusions

Results confirm the claim of show that

Interestingly, there is no outright clear method which proves best across all datasets.

We see that Naive Bayes preforms unquestionably superior in a flattened corpus and reigns at the top of all the accuracy reported by in the *20 newsgroup* data. We also note that it runs significantly faster than other algorithms.

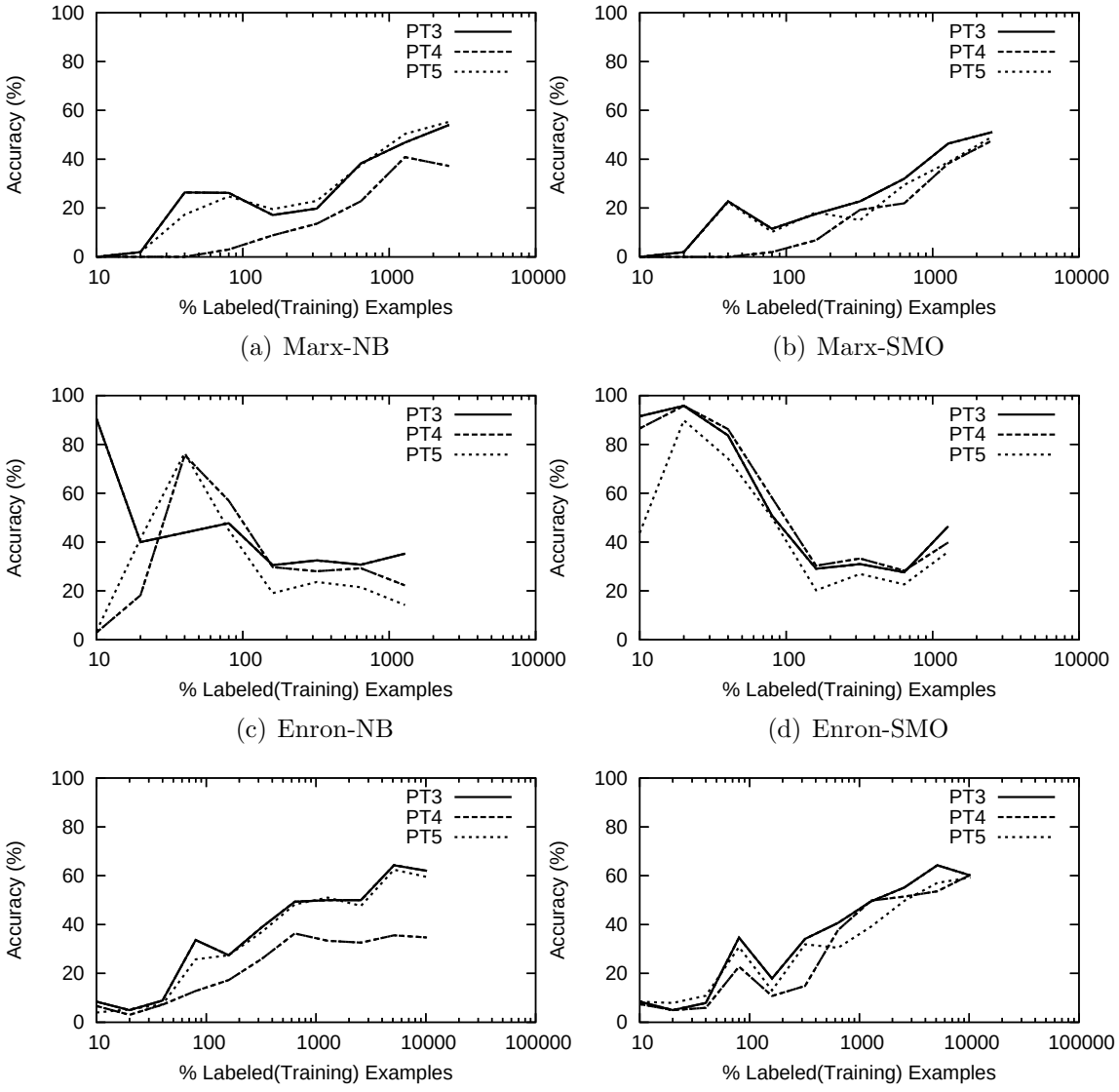
Nicolò Cesa-Biancha and colleges [7] also state the effectiveness of a hierarchy of SVMs, which is confirmed by the results listed in section **??** : where SMO certainly performed better than any other algorithm under every problem transformation method and generally as well as the flat algorithm. This supports the reasoning behind attempts to combine Naïve Bayes and SMO[7, 20].



(a) NB-ACC (b) NB-TIME

(c) SMO-ACC (d) SMO-TIME

Figure 4: The Value of Labeled Examples

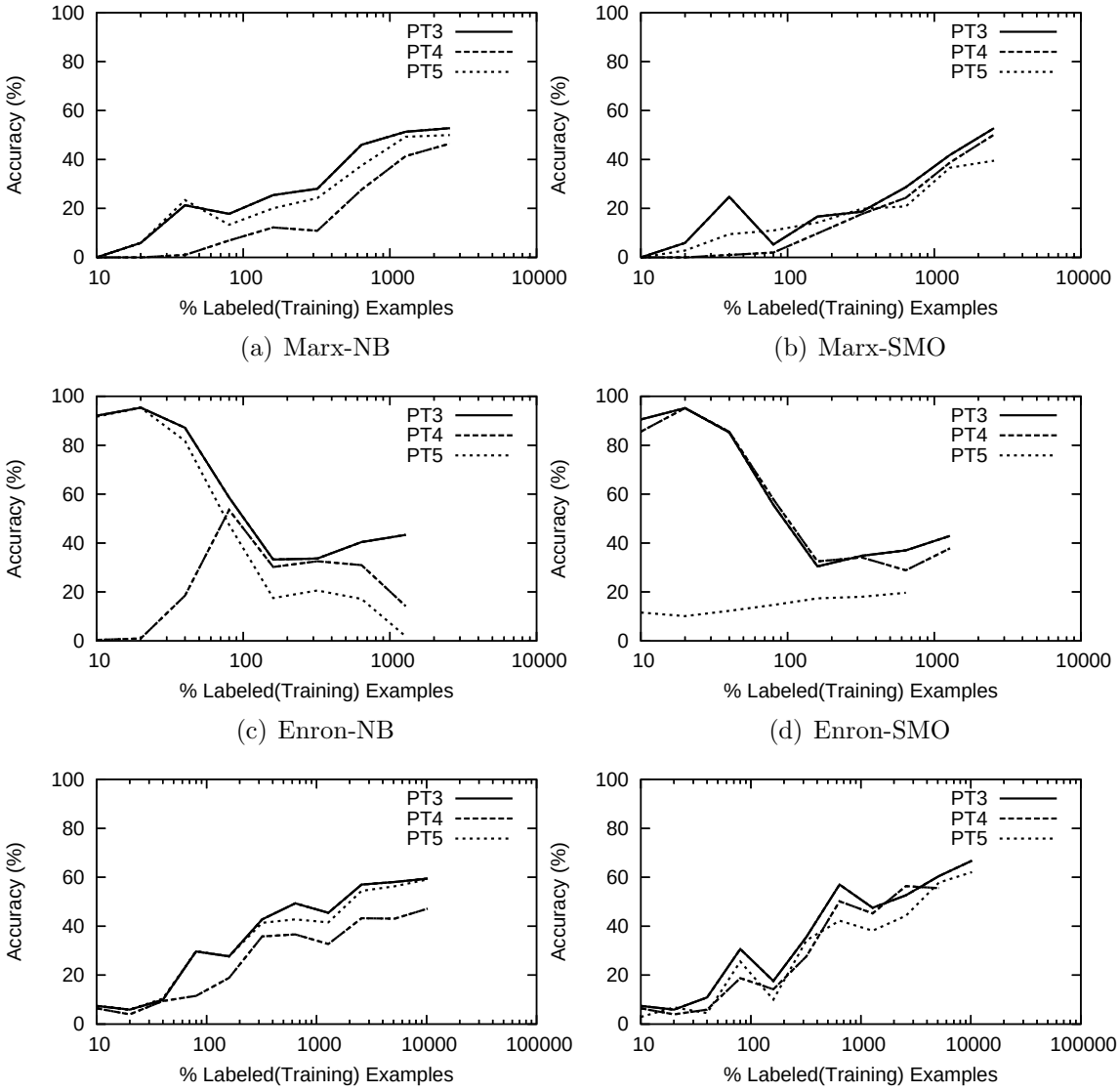


(a) Marx-NB (b) Marx-SMO

(c) Enron-NB (d) Enron-SMO

(e) NG-NB (f) NG-SMO

Figure 5: Accuracy Over Time - Problem Transformation Methods



(a) Marx-NB (b) Marx-SMO

(c) Enron-NB (d) Enron-SMO

(e) NG-NB (f) NG-SMO

Figure 6: Accuracy Over Time – Local Hierarchical Methods

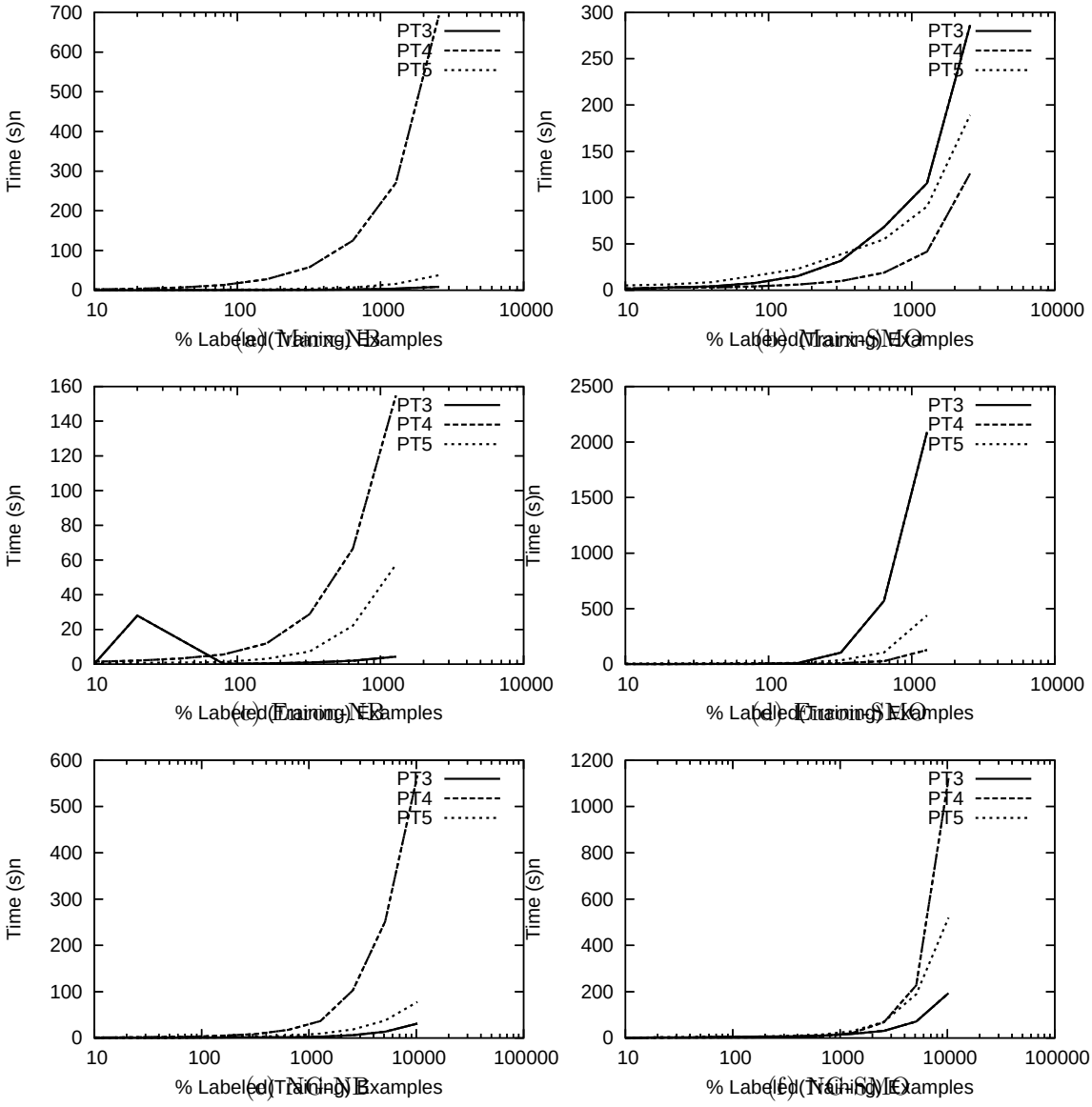


Figure 7: Accuracy Over Time - Problem Transformation Methods

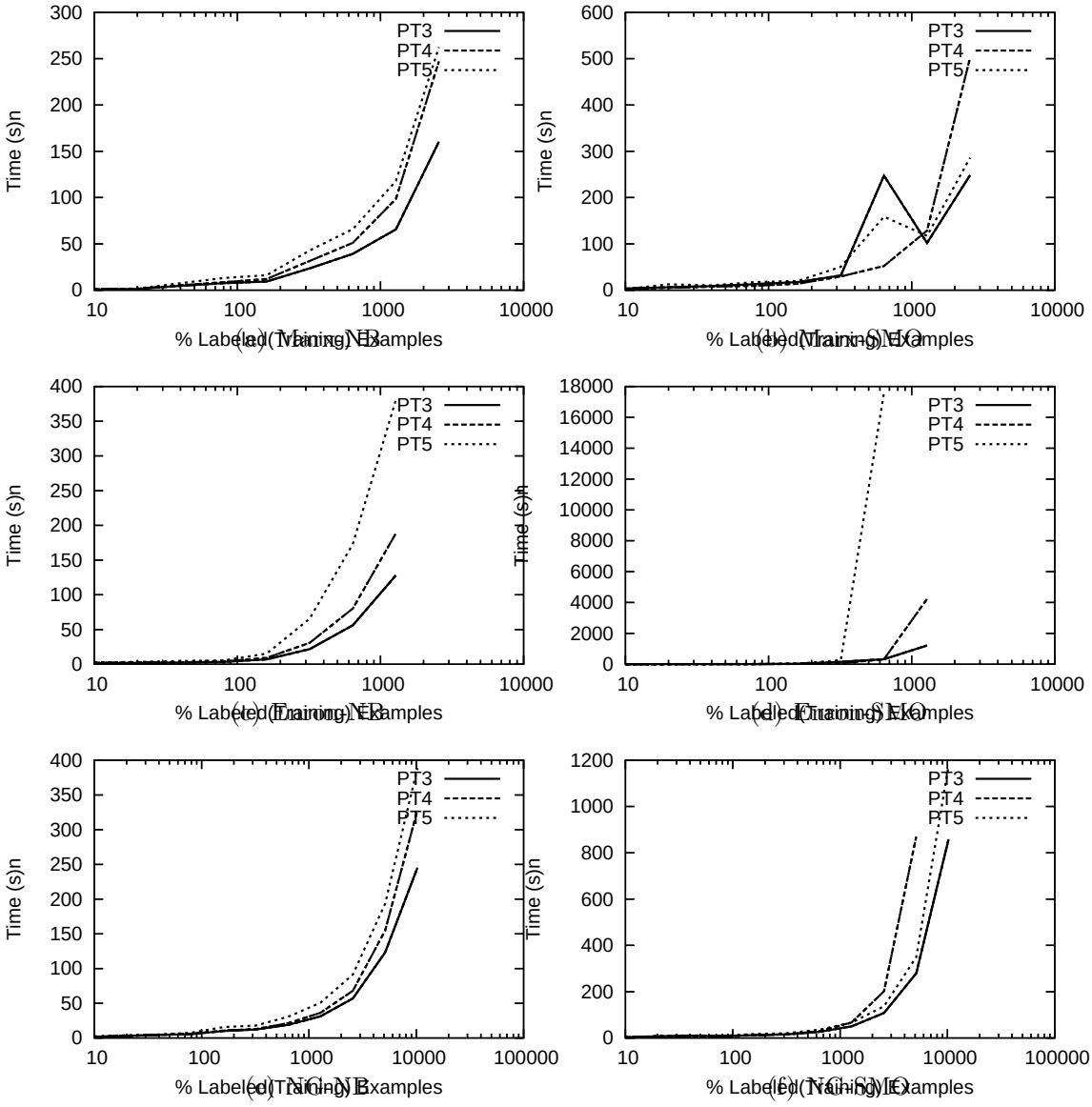


Figure 8: Accuracy Over Time – Local Hierarchical Methods

Overhead of LFS filtering, meaning Flat method superior to LHMC with Bayes – both accuracy and time. with some datasets. Not the case with complex hierarchies like Marx (because of smaller nodes). SMO more dependent on No. instances – so overhead doesn't outweigh anything

We can see from the results that the PT3-EM extension with naïve Bayes generally performs one or two percentage points better than with plain PT3., and in a subset of a very large corpus such as the *20newsgroups* actually the most superior of all algorithms. We also notice that this comes at a cost of time complexity. While, EM is not limited to naïve Bayes, it certainly performed more reliably better with that algorithm.

The best results with EM are had when using a massive percentage of unlabeled data, thou clearly unlabeled examples have a much weaker value than labeled examples.

Finally and most importantly in the semi-supervised discussion is that methods which use unlabeled examples such as EM are not inherently suited to stream based data. Clearly if we are going to train on a lot of unlabeled examples, a point in time t to a later point in time t' we have to assume that within this ΔT window, the label set will not have changed. This is an assumption we not be able to make all of the time.

This is unrealistic in an on-line active learning setting without even mentioning the additional computational overhead. EM will be left alone.

It is little surprise that the accuracies of PT4 are generally significantly inferior to the PT3 and PT5 methods on most datasets with the exception of the MEDI dataset – the explanation being that MEDI is a non-hierarchical dataset, and the independence of labels is the same assumption that PT4

makes.

might perform better if ranked

'degree of wrongness' eg. US House Prices under North America -¿ US

Although results of PT5 deteriorate rapidly for the ENRON corpus, evidence suggests that the chosen threshold is the cause. We see that with a carefully chosen threshold.... The ENRON corpus performs poorly in general because of branches 3 and 4, which evidently are not compatible with multi-label learning given the limited data supplied by emails.

better degrees of certainty with the ranking problem– could be useful

remembering that we get partially correct classifications which may be better than totally wrong

Local hierarchical learning proves undeniably more efficient than flat and global learning, demonstrated clearly just below in figure **??** which is based on the table data of **??**. This is particular the case with SMO which failed to perform in some instances for lack of memory.

PT3 keeps going up - as it learns combinations, but aside from that – accuracy tapers off sharply and even begins to decrease. This shows that the value of labeled examples decreases dramatically.

PT5 / Probabilistic wPT5/4 – selecting quality thresholds

Intuition Important -side point marxist.com languags / theory

Efficiency Important to a degree, but can be offset by using less training data

7 Plans for Future Work

Preliminary results have shown that the transformation method, PT5, while least documented, proves very promising. PT5 could benefit from some of the procedures used elsewhere such as EM and stacking algorithms. Most importantly the threshold of PT5 could be fine tuned. Research will proceed down the branch of local hierarchical approaches, mainly as preliminary results have confirmed the

- reduced computational complexity associated with this approach which is essential for an online context;
- its suitability for use with Naïve Bayes, an easily update-able algorithm, implying adaptability;
- the potential for improvement via the application of techniques such as stacking and EM;
- the potential gain and especially work with it's threshold;
- the lack of prior work in this area.

8 Schedule

8.1 Completed Work

March 2007 to August 2007

- Extensive literature review of previous work

- Gathered and created a variety of datasets
- Re-implemented all known problem transformation methods
- Re-implemented a generalised EM method
- Re-implemented a generalised stacking method
- Re-implemented a generalised boosting method
- Experimented with different forms of feature selection
- Re-implemented local hierarchical pachinko machine approaches with both local and global feature selection
- Re-implemented a local hierarchical probabilistic approach
- Empirically compared and evaluated the above methods, in batch learning and on-line settings
- Conclusions drawn and research focus area tightened
- Wrote final proposal

8.2 Scheduled Work

September 2007 to February 2008

- Literature review to keep up to date with current related work
- To concentrate on local hierarchical methods in a on-line setting, working in depth with:

  - preprocessing strategies;
  - different algorithms and parameter combinations;
  - feature selection techniques;
  - adaptive learning strategies;

  and different combinations of these, on the different datasets available.

- Development of visualisation tools to explore datasets, their hierarchical structure and the effect on classification.
- Empirical experimental analysis of local hierarchical methods under these various strategies.

March 2008 to August 2008

- Literature review to keep up to date with current related work
- Development of a user-driven practical application to demonstrate the utility of the research
- Testing the application with real-world time-based on-line data in a real world scenario
- Drawing conclusions and determining the implications for development
- Formulation and development of novel approaches following those implications

September 2008 to February 2009

- Development cycles of:

1. Empirical experimentation with novel framework and algorithms,
2. analysis and evaluation of experiment results against those of other prior works, and
3. reformulation of novel work to improve the framework.

March 2009 to August 2009

- Additional improvements and tuning to preprocessing techniques, algorithms and parameter combinations
- Extensive empirical testing of the novel framework and statistical analysis of results

September 2009 to February 2010

- Thesis writing

March 2010

- Thesis submission

9 Thesis Outline

1. Introduction
2. Problem Definition
3. Prior Work
4. Multi-label Classification

5. Hierarchical Classification
6. Feature Selection
7. On-line Classification
8. Active Learning
9. Presentation of Novel Work
10. Experiments and Evaluation
11. Conclusions
12. Future Work
13. Appendices

10 Resource Requirements

The only resource requirements for this research are totally satisfied by the Computer Science department's machine learning laboratory

11 Ethical Consent

There is no need to use personal or sensitive data for this project due to the widely available public non-sensitive data which can be gathered, hence there are no ethical issues that need to be dealt with before proceeding with this work.

References

[1] *A k-nearest neighbor based algorithm for multi-label classification*, volume 2, 2005.

[2] Zafer Barutcuoglu, Robert E. Schapire, and Olga G. Troyanskaya. Hierarchical multi-label prediction of gene function. *Bioinformatics*, 22(7):830–836, April 2006.

[3] Dendrik Blockeel, Leander Schietgat, Jan Struyf, Amanda Clare, and Saso Dzeroski. Hierarchical multilabel classification trees for gene function prediction (extended abstract).

[4] Janez Brank, Dunjaj Mlademic, and Marko Grobelnik. Hierarchical text categorization using coding matrices. In *SiKDD*, October 2006.

[5] Klaus Brinker. *On Active Learning in Multi-label Classification*. 2006.

[6] Klaus Brinker, Johannes Frnkranz, and Eyke Huellermeier. A unified model for multilabel classifcation and ranking. In *The 17th European Conference on Artificial Intelligence*.

[7] Nicolò Cesa-Bianchi, Claudio Gentile, and Luca Zaniboni. Hierarchical classification: combining bayes with svm. In *ICML '06: Proceedings of the 23rd international conference on Machine learning*, pages 177–184, New York, NY, USA, 2006. ACM Press.

[8] Allen Chan and Alex A. Freitas. A new ant colony algorithm for multi-label classification with applications in bioinfomatics. In *GECCO '06:*

Proceedings of the 8th annual conference on Genetic and evolutionary computation, pages 27–34, New York, NY, USA, 2006. ACM Press.

[9] Yu H. Chang. Email filtering: Machine learning techniques and an implementation for the unix pine mail system.

[10] Chun H. Cheng, Jian Tang, Ada W. Chee, and Irwin King. Hierarchical classification of documents with error control. In David Cheung, Qing Li, and Graham Williams, editors, *Proceedings of PAKDD-01, 5th Pacific-Asia Conferenece on Knowledge Discovery and Data Mining*, pages 433–443, Hong Kong, CN, 2001. Springer Verlag, Heidelberg, DE.

[11] Ed H. Chi and Todd Mytkowicz. Understanding navigability of social tagging systems. 2007.

[12] Wesley T. Chuang, Asok Tiyyagura, Jihoon Yang, and Giovanni Giuffrida. A fast algorithm for hierarchical text classification. In Yahiko Kambayashi, Mukesh Mohania, and Min A. Tjoa, editors, *Proceedings of DaWaK-00, 2nd International Conference on Data Warehousing and Knowledge Discovery*, pages 409–418, London, UK, 2000. Springer Verlag, Heidelberg, DE.

[13] Gao Cong, Wee S. Lee, Haoran Wu, and Bing Liu. Semi-supervised text classification using partitioned em.

[14] Domonkos T. Department. Experiment with a hierarchical text categorization method on the wipo-alpha.

[15] Susan T. Dumais and Hao Chen. Hierarchical classification of web content. In Nicholas J. Belkin, Peter Ingwersen, and Mun K. Leong, editors, *Proceedings of SIGIR-00, 23rd ACM International Conference on Research and Development in Information Retrieval*, pages 256–263, Athens, GR, 2000. ACM Press, New York, US.

[16] Andr Elisseeff and Jason Weston. A kernel method for multi-labelled classification.

[17] Y. Freund and R. Schapire. A short introduction to boosting, 1999.

[18] Nadia Ghamrawi and Andrew Mccallum. Collective multi-label classification. In *CIKM '05: Proceedings of the 14th ACM international conference on Information and knowledge management*, pages 195–200, New York, NY, USA, 2005. ACM Press.

[19] Rayid Ghani. Using error-correcting codes for text classification. In Pat Langley, editor, *Proceedings of ICML-00, 17th International Conference on Machine Learning*, pages 303–310, Stanford, US, 2000. Morgan Kaufmann Publishers, San Francisco, US.

[20] S. Godbole, S. Sarawagi, and S. Chakrabarti. Scaling multi-class support vector machines using inter-class confusion, 2002.

[21] Shantanu Godbole and Sunita Sarawagi. Discriminative methods for multi-labeled classification. In *8th Pacific-Asia Conference on Knowledge Discovery and Data Mining*, 2004.

[22] Elizaboth L. Hohman and David J. Marchette. A dynamic graph model for analyzing streaming news documents. In *2007 IEEE Symposium on Computational Intelligence and Data Mining*, pages 462–469, 2007.

[23] Ian. *Data Mining: Practical machine learning tools and techniques*. Morgan Kaufmann, San Francisco, second edition, 2005.

[24] Julia Itskevitch. Automatic hierarchical e-mail classification using association rules.

[25] Svetlana Kiritchenko. *Hierarchical Text Categorization and its Application to Bioinformatics*. PhD thesis, 2005.

[26] Svetlana Kiritchenko, Stan Matwin, Richard Nock, and Fazel Famili. Learning and evaluation in the presence of class hierarchies: Application to text categorization. In Springer and S. Lcn, editors, *Proc. of the 19th Canadian Conference on Artificial Intelligence*, volume 4013, pages 395–406, 2006.

[27] J. Kleinberg. Bursty and hierarchical structure in streams. In *Proc. 8th ACM SIGKDD Intl. Conf. on Knowledge Discovery and Data Mining*, 2002.

[28] D. Koller and M. Sahami. Hierarchically classifying documents using very few words, 1997.

[29] Ray Liere and Prasad Tadepalli. The use of active learning in text categorization.

[30] Xiao Luo and Nur A. Zincir-Heywood. *Evaluation of Two Systems on Multi-class Multi-label Document Classification*. 2005.

[31] A. Mccallum, R. Rosenfeld, T. Mitchell, and A. Ng. Improving text classification by shrinkage in a hierarchy of classes, 1998.

[32] Andrew K. Mccallum. Multi-label text classification with a mixture model trained by em.

[33] Andrew K. Mccallum and Kamal Nigam. Employing em and pool-based active learning for text classification. In *Proc. 15th International Conf. on Machine Learning*, pages 350–358. Morgan Kaufmann, San Francisco, CA, 1998.

[34] D. Mladenic and M. Grobelnik. Feature selection for classification based on text hierarchy.

[35] Kamal Nigam, Andrew Mccallum, and Tom M. Mitchell. Semi-supervised text classification using em. 2000.

[36] Kamal Nigam, Andrew K. Mccallum, Sebastian Thrun, and Tom M. Mitchell. Text classification from labeled and unlabeled documents using em. *Machine Learning*, 39(2/3):103–134, 2000.

[37] World A. On. Stream-based active learning for data selection in a real.

[38] Juho Rousu, Craig Saunders, and Et. On maximum margin hierarchical multilabel classification.

[39] Miguel E. Ruiz and Padmini Srinivasan. Hierarchical text categorization using neural networks. *Information Retrieval*, 5(1):87–118, 2002.

[40] R. Schapire and Y. Singer. Boostexter: A system for multiclass multi-label text categorization, 1998.

[41] Greg Schohn and David Cohn. Less is more: Active learning with support vector machines. In *Proc. 17th International Conf. on Machine Learning*, pages 839–846. Morgan Kaufmann, San Francisco, CA, 2000.

[42] J. Struyf, S. Dzeroski, H. Blockeel, and A. Clare. Hierarchical multi-classification with predictive clustering trees in functional genomics, 2005.

[43] D. Tikk, J. Yang, and S. Bang. Hierarchical text categorization using fuzzy relational thesaurus.

[44] Domonkos Tikk, Gyrgy Bir, and Jae D. Yang. A hierarchical text categorization approach and its application to frt expansion.

[45] Simon Tong and Daphne Koller. Support vector machine active learning with applications to text classification. In Pat Langley, editor, *Proceedings of ICML-00, 17th International Conference on Machine Learning*, pages 999–1006, Stanford, US, 2000. Morgan Kaufmann Publishers, San Francisco, US.

[46] G. Tsoumakas and I. Katakis. Multi label classification: An overview. *International Journal of Data Warehousing and Mining*, 3(3), 2007.

[47] David Vilar and María José. Multi-label text classification using multinomial models. 2004.

[48] Ke Wang, Senquiang Zhou, and Shiang C. Liew. Building hierarchical classifiers using class proximity. In Malcolm P. Atkinson, Maria E. Orlowska, Patrick Valduriez, Stanley B. Zdonik, and Michael L. Brodie, editors, *Proceedings of VLDB-99, 25th International Conference on Very Large Data Bases*, pages 363–374, Edinburgh, UK, 1999. Morgan Kaufmann Publishers, San Francisco, US.

[49] Wahyu Wibowo and Hugh E. Williams. Simple and accurate feature selection for hierarchical categorisation. In *DocEng '02: Proceedings of the 2002 ACM symposium on Document engineering*, pages 111–118, New York, NY, USA, 2002. ACM Press.

[50] Wahyu Wibowo and Hugh E. Williams. Strategies for minimising errors in hierarchical web categorisation. In *CIKM '02: Proceedings of the eleventh international conference on Information and knowledge management*, pages 525–531, New York, NY, USA, 2002. ACM Press.

[51] S. Zhong. Efficient streaming text clustering. *Neural Netw*, 18(5-6):790–798, 2005.

[52] Shenghuo Zhu, Xiang Ji, Wei Xu, and Yihong Gong. Multi-labelled classification using maximum entropy method.