A Mixture-of-Experts Framework for Learning from Imbalanced Data Sets

Andrew Estabrooks
IBM Toronto Lab, Office 1B28B (1B/813/1150/TOR),
1150 Eglinton Avenue East, North York, Ontario,
Canada, M3C 1H7
aestabro@ca.ibm.com

Nathalie Japkowicz[*]
SITE, University of Ottawa,
150 Louis Pasteur, P.O. Box 450 Stn. A, Ottawa, Ontario,
Canada K1N 6N5
nat@site.uottawa.ca
Phone: (613) 562-5800 ext. 6693
FAX: (613) 562-5187

[*] Corresponding Author

Abstract

Re-Sampling methods are some of the different types of approaches proposed to deal with the class-imbalance problem. Their advantage over other such approaches is that they are *external* and thus, easily transportable. Although such approaches can be very simple to implement, tuning them most effectively is not an easy task. In particular, it is unclear whether oversampling is more effective than undersampling and which oversampling or undersampling rate should be used. This paper presents an experimental study of these questions and concludes that combining different expressions of the re-sampling approach in a mixture of experts framework is an effective solution to the tuning problem. The proposed combination scheme is evaluated on drastically imbalanced subsets of the Reuters-21578 text collection and is shown to be very effective for these problems.

Introduction

In a concept-learning problem, the data set is said to present a class imbalance if it contains many more examples of one class than the other. Such a situation poses challenges for typical classifiers such as Decision Tree Induction Systems or Multi-Layer Perceptrons that are designed to optimize overall accuracy without taking into account the relative distribution of each class (Japkowicz 2000; Estabrooks 2000). As a result, these classifiers tend to ignore small classes while concentrating on classifying the large ones accurately. Unfortunately, this problem is quite pervasive as many domains are cursed with a class imbalance. This is the case, for example, with text classification tasks whose training sets typically contain much fewer documents of interest to the reader than on irrelevant topics. Other domains suffering from class imbalances include target detection, fault detection, or fraud detection problems, which, again, typically contain much fewer instances of the event of interest than of irrelevant events.

A large number of approaches have previously been proposed to deal with the class imbalance problem.[1] These approaches can be categorized into two groups: the *internal* approaches that create new algorithms or modify existing ones to take the class imbalance problem into consideration (Pazzani et al. 1994; Riddle et al. 1994; Japkowicz et al. 1995; Kubat et al. 1998) and *external* approaches that use un-modified existing algorithms, but re-sample the data presented to these algorithms so as diminish the effect caused by their class imbalance (Lewis & Gale 1994; Kubat & Matwin 1997; Ling & Li 1998). The internal approaches just mentioned may, in certain cases, be quite effective, but they have the

[1] For a full review of these works, please consult (Estabrooks 2000).

disadvantage of being algorithm-specific. This is a problem since data sets presenting different characteristics are better classified by different algorithms (see, for example, (Weiss & Kapouleas 1990)), and it might be quite difficult (if not, sometime impossible) to transport the modification proposed for the class imbalance problem from one classifier to the other. External approaches, on the other hand, are independant of the classifier used and are, thus, more versatile. This is why we chose to focus on these approaches rather than internal ones in this study.

External approaches may, themselves, be divided into two types of categories. First, there are approches that focus on studying what the best *data* for inclusion in the training set are (Lewis & Gale 1994; Kubat & Matwin 1997) and, second, there are approaches that focus on studying what the best *proportion* of positive and negative examples to include in a training set is (Ling & Li 1998). We decided to focus on the second question with the idea that once a good framework for dealing with the proportion question is chosen, this framework can be refined by making "smarter" re-sampling choices as per the first category of external approaches.

In more detail, our study considers the two different categories of re-sampling approaches: methods that *oversample* the small class in order to make it reach a size close to that of the larger class and methods that *undersample* the large class in order to make it reach a size close to that of the smaller class. The purpose of this paper is to find the best way to tune the re-sampling paradigm. In particular, we ask the following three questions:

1. Should we *oversample* or *undersample*?

2. At what *rate* should this oversampling or undersampling take place?

3. Can a *combination* of different expressions of the re-sampling paradigm help improve classification accuracy?

These questions are answered in the context of a decision tree induction system: C5.0, and all re-sampling is done randomly.

The paper is divided into three parts. The first part establishes the problems caused by the class imbalance problem by studying its effect on different artificial domains. In the second part, we conduct an experimental study on some of these artificial data sets in order to explore the first two questions asked above. This study suggests an answer to the third question in the form of a combination scheme that is described and tested on both the artificial data sets and a series of 10 text classification tasks in the third part of the paper.

Part I: The Effects of Class Imbalances

In this part of the paper, we study the effect of class imbalances on data sets representing target concepts of various complexities. In this particular study, the size of the training set is held constant, which means that, as the target concept (represented by the positive class) becomes more complex, the positive class becomes sparser relative to the target concept.[2] This study is relevant since, in real-world data sets, we often encounter situations where the target concept is quite complex, but there are not enough data available to describe it.

In order to investigate the performance of induced decision trees on balanced and imbalanced data sets, eight sets of training and testing data of increasing complexities were created. All the experiments are conducted on artificial data sets defined over the domain

[2] A similar but more thorough study relating different degrees of imbalance ratios, training set sizes and concept difficulty was conducted by Japkowicz (2000) (this study uses neural networks rather than decion tree induction systems). However,this study falls beyond the scope of this paper.

of DNF expressions. DNF expressions were specifically chosen because of their simplicity as well as their similarity to text data whose classification accuracy we are ultimately interested in improving. In particular, like in the case of text-classification, DNF concepts of interest are, generally, represented by much fewer examples than there are counter-examples of these concepts, especially when 1) the concept at hand is fairly specific; 2) the number of disjuncts and literals per disjunct grows larger; and 3) the values assumed by the literals are drawn from a large alphabet. Furthermore, an important aspect of concept complexity can be expressed in similar ways in DNF and textual concepts since adding a new subtopic to a textual concept corresponds to adding a new disjunct to a DNF concept.

The target concepts in the data sets were made to vary in concept complexity by increasing the number of disjuncts in the expresion to be learned, while keeping the number of conjunctions in each disjunct constant. In particular, expressions of complexity c= 4x2, 4x3, 4x4, 4x5, 4x6, 4x7 and 4x8 were created where the first number represents the number of literals present in each disjunct and the second represents the number of disjuncts in each concept. We used an alphabet of size 50. For each concept, we first created a training set containing 6,000 positive and 6,000 negative examples. We then 1) randomly removed 4,800 positive examples from the training set, thus creating a 1:5 class imbalance in favour of the negative class and 2) randomly removed 960 extra examples from the training set, thus creating a 1:25 class imbalance in favour of the negative class.[3] In all three cases (no class imbalance, a 1:5 class imbalance and a 1:25 class imbalance), we tested the classifier on 1,200 positive and 1,200 negative examples. For each expression, the results of C5.0 were

[3] Imbalanced ratios greater than 1:25 were not tried on this particular problem since we did not want to confuse the imbalance problem for the small sample problem.

averaged over 10 runs on different domains of the same complexity.

Figure 1 shows the results obtained in this set of experiments over the positive testing set. The error obtained over the negative training set was negligeable since it always remained below 1.5% for an imbalance ratio of 1:1 and never rose over 0.1% for the larger imbalance ratios.

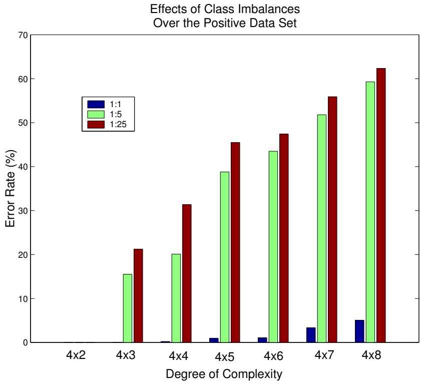


Figure 1: The effect of class imbalance on various target concept complexities over the positive testing set (it is negligeable over the negative testing set)

The results show clearly that a class imbalance causes a sharp decrease in accuracy given a single target concept complexity. Furthermore, it shows that the more complex the target concept the more negative effect class imbalances have. Because in real world domains, the target concept is often quite complex and the class imbalance problem real, this study suggests that the class imbalance problem desperately needs to be addressed in order to maintain an acceptable error rate on such domains.

Part II: Over-Sampling versus Under-Sampling

In this part of the paper, we study the effects of oversampling versus undersampling and oversampling or undersampling at different rates.[4] The part is divided into three sections.

[4] Throughout this work, we consider a fixed imbalance ratio, a fixed number of training examples and a fixed degree of concept complexity.

In the first section, we study the effect of over-sampling versus under-sampling when both methods keep on re-sampling until the imbalance has completely vanished. The second section investigates the effect of the two re-sampling methods further by asking whether the two methods yield C5.0 to use the same or different learning strategies for a same problem. The third section considers the question of re-sampling at different rates rather than until the two classes get balanced.

2.1 Over-Sampling and Under-Sampling to Full Balance

The purpose of this section is to explain the effects of full oversampling and undersampling on imbalanced domains. In order to illustrate these effects, a complex and imbalanced class of domains was used: the class of 4x7 DNF concepts designed with a 1:25 class imbalance in favour of the negative class. Similar results on different concept complexities are reported in (Estabrooks 2000).

Three sets of experiments were conducted. First, we trained and tested C5.0 on the 4x7 data sets, left untouched. Second, we randomly oversampled the positive class, until its size reached the size of the negative class, i.e., 6,000 examples. The added examples were straight copies of the data in the original positive class, with no noise added. Finally, we undersampled the negative class by randomly eliminating data points from the negative class until it reached the size of the positive class or, 240 data points. Here again, we used a straightforward random approach for selecting the points to be eliminated. Each experiment was repeated 50 times on 4x7 DNF concepts and using different oversampled or removed examples.[5] After each training session, C5.0 was tested on separate testing sets containing

[5] Note that, throughout the paper, the experiments on artificial domains are ran on different numbers of trials. This was done to maintain an acceptable level of reliability in our results. If the variance in trial results was high, greater numbers of trials were used than if their variance was low.

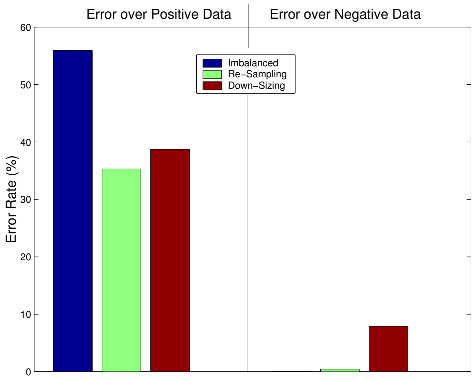


Figure 2: Re-Sampling versus Downsizing

1,200 positive and 1,200 negative examples. The average accuracy results are reported in Figure 2. The left side of the figure shows the results obtained on the positive testing set while its right side shows the results obtained on the negative testing set.

The results show that the number of false negatives (results over the positive class) is a lot higher than the number of false positives (results over the negative class). This is not surprising given the fact that the data set is sharply imbalanced in favour of the negative class. As well, the results suggest that both full oversampling and undersampling are quite helpful for reducing the error caused by the class imbalance on this problem although oversampling appears more accurate than undersampling.[6]

2.2. Is OverSampling equivalent to UnderSampling?

The results of the previous section suggest that both the full oversampling and undersampling methods used are helpful for reducing the error caused by a class imbalance although

[6] Note that the usefulness of oversampling versus undersampling is problem dependent. Domingos (1999), for example, finds that in the majority of data sets he tested, undersampling is more effective than oversampling. In all our experiments on artificial domains, however, the opposite can be observed. We believe that the nature of the negative data set has a lot to do with whether undersampling or oversampling will be more effective. If the negative examples provided are very different from the positive ones, then they are not particularly useful and the negative class can be undersampled. If, on the other hand, the negative examples are closely related to the positive class and help define it, then they shouldn't be eliminated. In addition, the robustness of the classifier to large training sets also needs to be taken into consideration prior to choosing an oversampling strategy.

oversampling appears more effective than undersampling on the 4x7 problems. If a single classifier is to be used along with a simple re-sampling approach on similar domains, this result, therefore, suggest that oversampling is probably a better choice than undersampling. However, if several classifiers are to be used and combined and other types of domains are to be classified, the question that needs to be answered is:

> Are the two approaches solving the problem using the same strategy more or less successfully or are they going about solving the problem using different strategies altogether?

If the former hypothesis is the correct one, then there is no real point in combining the two approaches. If, however, the second hypothesis is correct, then there might be ways to combine the two approaches that will make the overall system benefit from the strengths of both strategies and, hopefully, avoid their weaknesses.

In order to answer this question, we decided to look at the rule set learned by C5.0 in both the undersampling and oversampling cases. Because oversampling is based on a much larger negative training set than undersampling (in oversampling there are 6,000 distinct negative examples while there are only 240 in undersampling) but the number of *distinct* positive examples is the same in both cases (even though there are 6,000 positive examples in the case of oversampling, there are only 240 distinct such examples), we looked only at the positive rule set. The data we gathered on the positive rule sets of both strategies and for different concept complexities are summarized in Table 1. The results reported in the table represent the averages of 1) the size of the rule set learned by C5.0 for a given concept class in terms of number of disjuncts per rule and 2) the number of rules per set learned by

C5.0 for that concept class. These averages were obtained over 30 different runs of C5.0.

Expr. Comp.	**Under** *R.Size*	**Under** *R.Num.*	**Over** *R.Size*	**Over** *R.Num.*
4x2	4.0	2.0	4.0	2.0
4x4	3.8	5.6	4.0	4.0
4x6	4.7	13.5	4.0	6.0
4x8	4.9	15.4	8.3	36.2
4x10	5.0	18.6	8.5	43.7

Table 1: The size of C5.0-oversampled and C5.0-undersampled rule sets. “Under” corresponds to Undersampling, “Over” corresponds to Oversampling, “R.Size” corresponds to the average size of the rules generated by C5.0 and “R.Num.” corresponds to the number of rules generated by C5.0.

The results reported in Table 1 show that oversampling and undersampling go about classifying the data set using different strategies. In particular, the two approaches seem to fit the concept perfectly for the concept class of size 4x2. While oversampling keeps on doing so for the concept classes of size 4x4 and 4x6 undersampling appears to overfit the positive data since the number of rules generated by the undersampling approach grows beyond the number of disjuncts in the concept class. For concept classes of size 4x8 and 4x10, both approaches seem to overfit the positive data but oversampling overfits them much more than undersampling. Indeeed, while undersampling generates a number of rules reaching about twice the number of disjuncts present in the corresponding concept classes, oversampling generates between 4 and 5 times more rules than there are disjuncts present in these concept classes. Oversampling also generates rules that are, on average, larger than those generated by undersampling, and both are beyond the number of conjuncts present in the training concepts. This suggests that the two methods use a different bias to classify the data and that there may be an advantage to combining both approaches rather than choosing one over the other.

2.3. Re-Sampling and Down-Sizing at various Rates

The purpose of this section is to find out what happens when different oversampling or undersampling rates are used, and whether the effect of using different re-sampling rates is the same for different domains. In order to illustrate our answer to these questions, we considered the domains of concept size 4x7 previously used in Section 2.1 and we added another class of domains, those of concept size 4x5. For this experiment, we assumed that, despite the imbalance in the training set, the cost of misclassifying instances of the positive class is the same as that of misclassifying instances of the negative class and we added the two errors together.[7]

Rather than simply oversampling and undersampling our domains by equalizing the size of the positive and the negative training sets, our experiments consisted of oversampling and undersampling them at different rates. In particular, we divided the difference between the size of the positive and negative training sets by 10 and used this value as an increment in our oversampling and undersampling experiments. We chose to make the 100% oversampling rate correspond to the fully oversampled data sets of section 2.1 but to make the 90% undersampled rate correspond to its fully undersampled data sets.[8] For example, data sets with a 10% oversampling rate contain $240 + (6,000 - 240)/10 = 816$ positive examples and 6,000 negative examples. Conversely, data sets with a 0% undersampling rate contain 240

[7] We made this assumption in order to simplify our analysis: it would have been more tedious to analyze the results obtained on the positive and the negative class separately. Note, however, that although the assumption we made is correct in certain real-world domains, it is not always so. We could, of course, have chosen another assumption as well, but our purpose here, is not to try every possible setting, but simply to demonstrate what may happen, given a chosen assumption, on two different domains re-sampled at different rates.

[8] This was done so that no classifier was duplicated in our combination scheme. (See Section 3.1)

positive examples and 6,000 negative ones while data sets with a 10% undersampling rate contain 240 positive examples and $6,000 - (6,000 - 240)/10 = 5424$ negative examples. A 0% oversampling rate and a 90% undersampling rate correspond to the fully imbalanced data sets designed in section 2.1 while a 100% undersampling rate corresponds to the case where no negative examples are present in the training set.

Once again, and for each oversampling and undersampling rate, the rules learned by C5.0 on the training sets were tested on testing sets containing 1,200 positive and 1,200 negative examples. The results of our experiments are displayed in Figure 3 for the case of oversampling and undersampling, respectively, on both the 4x5 and the 4x7 concepts. They represent the averages of 50 trials.

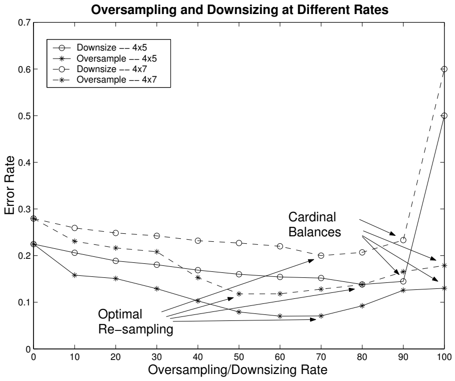


Figure 3: Oversampling and Downsizing at Different Rates

The results shown in this figure suggest that different sampling rates have different effects on the accuracy of C5.0 on imbalanced data sets for both the oversampling and the undersampling method. Furthermore, the effect is different for different domains. In particular, the following observations can be made:

- Oversampling or undersampling until a cardinal balance of the two classes is reached is not necessarily the best strategy: best accuracies are reached before the two sets are cardinally balanced.

- The optimal oversampling and undersampling rates vary as a function of the domain. In particular, they are different for domains including concepts of varying complexity.

In more detail, the first observation comes from the fact that in both the oversampling and undersampling curves of Figure 3 the optimal accuracy is not obtained when the positive and the negative classes have the same size. In the oversampling curves where class equality is reached at the 100% oversampling rate, the average errors obtained on the 4x5 and 4x7 concepts at that point are 13% and 17.8% respectively, whereas the optimal average error rate obtained in these two cases are 7.08% and 11.8%, respectively. Similarly, although less significantly, in the undersampling curves, where class equality is reached at the 90% undersampling rate, the average error rates obtained on the 4x5 and 4x7 concepts at that point are 14.48% and 23.23% respectively, whereas the optimal average error rates obtained in these two cases are 13.81% and 20%, respectively.[9]

The second observation comes from the fact that in both the oversampling and undersampling curves, the optimal average sampling rates are not the same for the 4x5 and 4x7 concepts. In the oversampling case, the optimal oversampling rate for the 4x5 concept is 60% while it is 50% for the 4x7 concept. Similarly, the optimal undersampling rate is 80% in the 4x5 case while it is 70% in the 4x7 case. Further results for other concept classes can be found in (Estabrooks 2000).

[9] Note that the sharp increase in error rate taking place at the 100% undersampling point is caused by the fact that at this point, no negative examples are present in the training set.

Part III: A Mixture-of-Experts Scheme

The results obtained in the previous part of the paper suggest that it might be useful to combine oversampling and undersampling versions of C5.0 sampled at different rates. On the one hand, the combination of the oversampling and undersampling strategies may be useful given the fact that the two approaches are both useful in the presence of imbalanced data sets and appear to learn concepts in different ways (cf. results of Section 2.1 and 2.2). On the other hand, the combination of classifiers using different oversampling and undersampling rates may be useful since optimal sampling rates are different in different domains and we may not be able to predict, in advance, which rate is optimal given a new domain (cf. results of Section 2.3). We will now describe the combination scheme we designed to deal with the class imbalance problem. This combination scheme is first tested on some artificial domains and it is then tested on a series of imbalanced subsets of the Reuters-21578 text classification domain.

3.1. Description of the Combination Scheme

A combination scheme for inductive learning consists of two parts. On the one hand, we must decide *which* classifiers will be combined and on the other hand, we must decide *how* these classifiers will be combined. We begin our discussion with a description of the architecture of our mixture of experts scheme. This discussion explains which classifiers are combined and gives a general idea of how they are combined. The specifics of our combination scheme are motivated and explained in the subsequent part of the discussion.

3.1.1 Architecture

In order for a combination method to be effective, it is necessary for the various classifiers that constitute the combination to make different decisions (Hansen 1990). The previous part of our study suggests that undersampling and oversampling will produce classifiers able to make different decisions. Furthermore, different sampling rates will allow us to "hit" an optimal rate which could not be predicted in advance. This suggests a 3-level hierarchical combination approach consisting of the *output level*, which combines the results of the oversampling and undersampling experts located at the *expert level*, which themselves each combine the results of 10 classifiers located at the *classifier level* and trained on data sets sampled at different rates. In particular, the 10 oversampling classifiers oversample the data at rates 10%, 20%, ... 100% (the positive class is oversampled until the two classes are of the same size) and the 10 undersampling classifiers undersample the negative class at rate 0% (no re-sampling), 10%, ..., 90% (the negative class is undersampled until the two classes are of the same size). Figure 4 illustrates the architecture of this combination scheme that was motivated by Shimshoni (1998)'s Integrated Classification Machine.[10]

3.1.2. Detailed Combination Scheme

Our combination scheme is based on two different assumptions/observations:

Assumption #1: Within a single testing set, different testing points could be best classified by different single classifiers.

Observation #2: In class imbalanced domains for which the positive training set is small

[10] However, (Shimshoni 1998) presents a general architecture. It was not tuned to the imbalance problem, nor did it take into consideration the use of oversampling and undersampling to inject principled variance into the different classifiers.

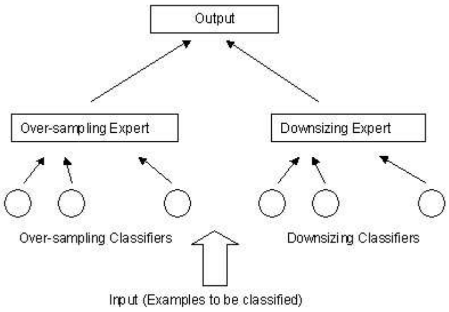


Figure 4: The mixture-of-experts architecture

and the negative training set is large, classifiers tend to make many false-negative errors (see, Figure 2, for example).

In order to deal with the first assumption, we decided not to average the outcome of different classifiers by letting them vote on a given testing point, but rather to let a single "good enough" classifier make a decision on that point. The classifier selected for a single data point needs not be the same as the one selected for a different data point. In general, letting a single, rather than several classifiers decide on a data point is based on the assumption that the instance space may be divided into non-overlapping areas, each best classified by a different expert. In such a case, averaging the result of different classifiers may not yield the best solution. We, thus, created a combination scheme that allowed single but different classifiers to make a decision for each point.

Of course, such an approach is dangerous given that if the single classifier chosen to make a decision on a data point is not reliable, the result for this data point has a good chance of being unreliable as well. In order to prevent such a problem, we designed an elimination procedure geared at preventing any unfit classifier present at our architecture's classification level from participating in the decision-making process. This elimination program relies on our second observation in that it invalidates any classifier labeling too many examples as positive. Since the classifiers of the combination scheme have a tendency of being naturally biased towards classifying the examples as negative, we assume that a classifier making too many positive decision is probably doing so unreliably.

In more detail, our combination scheme consists of

- a combination scheme applied to each expert at the expert level
- a combination scheme applied at the output level
- an elimination scheme applied to the classifier level

The expert and output level combination schemes use the same very simple heuristic: if one of the non-eliminated classifiers decides that an example is positive, so does the expert to which this classifier belongs. Similarly, if one of the two experts decides (based on its classifiers' decision) that an example is positive, so does the output level, and thus, the example is classified as positive by the overall system.

We designed two different elimination schemes depending on whether labeled data are available for the process or only unlabeled data can be found. In the case where labeled data are available, the elimination scheme consists of testing each classifier of the combination on a negative "weighing" data set (different from the negative testing set used to test the entire combination) and retaining only those that misclassified at most T examples of that set. T, the threshold can be set to different values and, after several cross-validation experiments, it was set to 0 for non-complex concepts of size 4x5 and below and to the number of positive training examples in the case of complex concepts (of size 4x8, 4x10 and 4x12).

In the case where only unlabeled data are available, the elimination scheme used at the classifier level uses the following heuristic: the first (most imbalanced) and the last (most balanced) classifiers of each expert are tested on an unlabeled data set. The number of positive classifications each classifier makes on the unlabeled data set is recorded and averaged and this average is taken as the threshold that none of the expert's classifiers must cross. In other words, any classifier that classifies more unlabeled data points as positive than the threshold established for the expert to which this classifier belongs needs to be discarded.[11]

It is important to note that, at the expert and output level, our combination scheme is heavily biased towards the positive under-represented class. This was done as a way to compensate for the natural bias against the positive class embodied by the individual classifiers trained on the class imbalanced domain. This heavy positive bias, however, is mitigated by our elimination scheme which strenuously eliminates any classifier believed to be too biased towards the positive class.

3.2 Results on the Artificial Domains

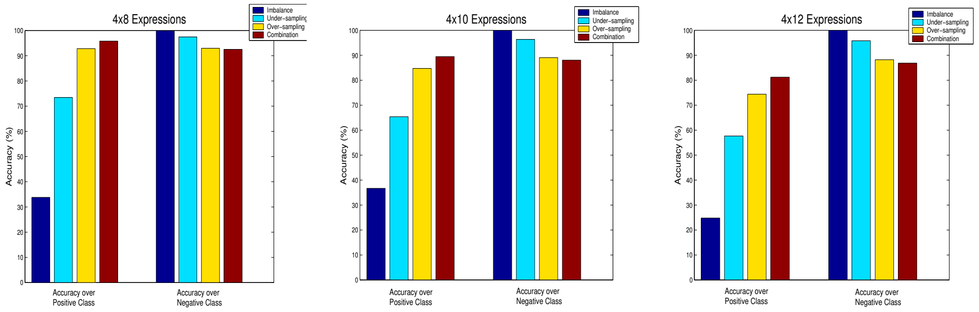
We began our system's evaluation by testing the combination scheme on a series of artificial domains. The results we obtained are reported here. The next section will report the results we obtained on real world domains, namely, on a series of text classification tasks. As mentioned above two different threshold settings of the elimination procedure were used on

[11] Because no labels are present, this technique constitutes an educated guess of what an appropriate threshold should be. The heuristic was tested in (Estabrooks 2000) and was shown to improve the system (over the combination scheme not using this heuristic) on the text classification data set by 3.2% when measured according to the F_1 measure, 0.36% when measured according to the F_2 measure, and 5.73% when measured according to the $F_{0.5}$ measure. See below, for a definition of the F_B measures, but note that the higher the F_B value, the better.

the artificial domains. For domains of small complexities (concepts of size smaller than 4x5), experiments revealed that varying the threshold only a small amount did not significantly alter the performance of the combination. As a matter of fact, requiring that all classifiers classify all the negative data of the weighing set accurately (setting the threshold to T=0) was able to achieve an accuracy of 100% on both the positive and the negative class. This 100% accuracy came as a result of the variance among the classifiers. In other words, although any given classifier in the undersampling expert missed a number of positive examples, at least one of the other classifier in the combination picked them up and classified them accurately as positive. In the meantime, no negative example got misclassified.

For domains of greater complexity (e.g., domains of concept size 4x8, 4x10 and 4x12), the threshold was established at the size of the positive training set, T = 240, with the idea that if a classifier misclassifies more than that number of examples as positive rather than negative, then it has less than 50% confidence over the positive class and is, thus, unacceptable.

We now report on our experiments on the more complex domains. In the first place, domains of concept size 4x8, 4x10 and 4x12, respectively, were generated, each containing 240 positive training examples, 12,000 negative examples, 1,200 positive testing examples and 1,200 negative testing examples. The negative training set was divided into 2 sets of 6,000 examples each. The first set was used along the positive training set for training the classifiers of the combination scheme and the second was used by the elimination scheme for "weighing" these trained classifiers. For each concept size, each experiment was ran 30 times on different domains belonging to the same concept class and the results correspond to the average obtained over these different runs. In each domain, 4 different classifiers



4x8 **4x10** **4x12**

Figure 5: A Comparison of learning on an imbalanced domain, under-sampling, over-sampling, and the combination scheme of domains of concept complexity 4x8, 4x10 and 4x12

were tested: C5.0 ran on the imbalanced data set (once again, the imbalance is of a ratio of 1:25 in favour of the negative class); the under-sampling expert (comprising 10 classifiers undersampling the training set at different rates); the over-sampling expert (comprising 10 classifiers oversampling the training set at different rates); and the combination of the two experts (20 classifiers undersampling and oversampling the training set at different rates).

As illustrated by Figure 5, testing the systems revealed that, a great improvement is achieved every time the under-sampling expert or the over-sampling expert or the combination of both is used. When comparing the performance of each expert to each other's and to that of the complete combination scheme, we found that, on average, the over-sampling expert performed better than the undersampling expert on the positive data sets. Conversely, the undersampling expert performed better than the oversampling expert on the negative data sets. In addition, we observed that combining both experts improved the performance on the positive data sets, but at the expense of the performance on the negative data sets. Nonetheless, in all cases, the gains in performance on the positive class were greater than its losses on the negative class with respect to both experts (this can be seen in Figure 5).

3.3 Results on a Text Classification Task

Since our combination scheme was shown to help increase classification accuracy on several classes of artificial domains, we also decided to test it on a real-world domain. In particular, we chose to test it on a subset of the ten largest categories of the the Reuters-21578 Data Set. We first present an overview of the data, followed by the results obtained by our scheme on these data.

3.3.1 The Reuters-21578 Data

The ten largest categories of the Reuters-21578 data set consist of the documents included in the classes of financial topics listed in Table 2:

Class.	**Document Count**
Earn	3987
ACQ	2448
MoneyFx	801
Grain	628
Crude	634
Trade	551
Interest	513
Wheat	306
Ship	305
Corn	254

Table 2: The top 10 Reuters-21578 categories

Several typical pre-processing steps were taken to prepare the data for classification. First, the data was divided according to the ModApte split which consists of considering all labelled documents published before 04/07/87 as training data (9603 documents, altogether) and all labelled documents published on or after 04/07/87 as testing data (3299 documents altogether). The unlabelled documents represent 8676 documents and were used during the classifier elimination step.

Second, the documents were transformed into feature vecors in several steps. Specifically, all the punctuation and numbers were removed and the documents were filtered through a stop word list[12]. The words in each document were then stemmed using the Lovins stemmer[13] and the 500 most frequently occurring features were used as the dictionnary for the bag-of-word vectors representing each documents.[14] Finally, the data set was divided into 10 concept learning problems where each problem consisted of a positive class containing 100 examples sampled from a single top 10 Reuters topic class and a negative class containing the union of all the examples contained in the other 9 top 10 Reuters classes. Dividing the Reuters multi-class data set into a series of two-class problems is typically done because considering the problem as a straight multiclass classification problem causes difficulties due to the high class overlapping rate of the documents, i.e., it is not uncommon for a document to belong to several classes simultaneously. Furthermore, although the Reuters Data set contains more than 100 examples in each of its top 10 categories (see Table 2), we found it more realistic to use a restricted number of positive examples.[15] Having restricted the number of positive examples in each problem, it is interesting to note that the class imbalances in these problems is very high since it ranges from an imbalance ratio of 1:60 to one of 1:100 in favour of the negative class.

[12] The stop word list was obtained at: http://www.dcs.gla.ac.uk/idom/it_resources/linguistic_utils/stop-words.

[13] The Lovins stemmer was obtained from: ftp://n106.isitokushima-u.ac.ip/pub/IR/Iterated-Lovins-stemmer

[14] A dictionary of 500 words is smaller than the typical number of words used (see, for example, (Scott & Matwin 1999)), however, it was shown that this restricted size did not affect the results too negatively while it did reduce processing time quite significantly (see (Estabrooks 2000)).

[15] Indeed, very often in practical situations, we only have access to a small number of articles labeled "of interest" whereas huge number of documents "of no interest" are available

3.3.2 Results

The results obtained by our scheme on these data were pitted against those of C5.0. However, since we decided that it was not fair to compare the effectiveness of a system of 20 classifiers to that of a single classifier, we also ran C5.0 with the Ada-boost option combining 20 classifiers.[16] The results of these experiments are reported in Figure 6 as a function of the micro-averaged (over the 10 different classification problems) F_1, F_2 and $F_{0.5}$ measures.[17]

In more detail, the F_B-measure is defined as:

$$F_B = \frac{(B^2+1)\times P\times R}{B^2\times P+R}$$

where the precision, P, and the Recall, R, are defined, respectively, as follows:

$$P = \frac{TruePositives}{TruePositives+FalsePositives}$$

$$R = \frac{TruePositives}{TruePositives+FalseNegatives}$$

The precision corresponds to the proportion of examples classified as positive that are truly positive; the recall corresponds to the proportion of truly positive examples that were classified as positive; the F_B-measure combines the precision and recall by a ratio specified by B. If $B = 1$, then precision and recall are considered as being of equal importance. If $B = 2$, then recall is considered to be twice as important as precision. If $B = 0.5$, then precision is considered to be twice as important as recall.

Because 10 different results are obtained for each value of B and each combination system (1 result per classification problem), these results had to be averaged in order to be presented

[16] C5.0 was shown in (Estabrooks 2000) to obtain results close to those obtained by state-of-the-art classifiers designed for text classification. We expected Adaboost to obtain even better results than C5.0 given that it is currently considered one of the best general-purpose classification algorithm (Breiman 1998).

[17] We chose to use these measures over measures of accuracy because text classification tasks are more concerned with issues of precision and recall (see below) than false positive and false negative rates. Furthermore, similarly to ROC curve and ROC hull reports, varying the B coefficient in the F_B formula allows us to evaluate a system under different performance requirements.

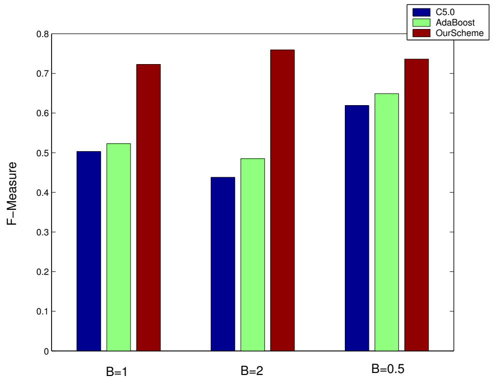


Figure 6: Micro-Averaged results obtained by C5.0, C5.0 with the Ada-Boost option and the Mixture-of-Experts scheme on 10 text classification problems. The results are reported in function of the F_1, F_2 and $F_{0.5}$ measures.

in a single graph. Micro-averaging consists of a straight average of the F-Measure obtained in all the problems, by each combination system, and for each value of B. Using Micro-averaging gives each problem the same weight, independently of the number of positive examples they contain.

The results in Figure 6 show that our combination scheme is much more effective than C5.0 and C5.0 with the Adaboost option on *both* recall and precision. Indeed, C5.0 and C5.0-Adaboost get F_1 measures of 50.3% and 52.3%, respectively, on the data set while our combination scheme gets an F_1 measure of 72.25%. If recall is considered as twice more important than precision, the results are even better. Indeed, the mixture-of-experts scheme gets an F_2-measure of 75.9% while C5.0 and C5.0-Adaboost obtain F_2-measures of 43.8% and 48.5%, respectively. On the other hand, if precision is considered as twice more important than recall, then the combination scheme is still effective, but not as effective with respect to C5.0 and C5.0-Adaboost since it brings the $F_{0.5}$-measure on the data set to only 73.61%, whereas C5.0's and C5.0-AdaBoost's performance amount to 61.9% and 64.9%, respectively.

The generally better performance displayed by our proposed system when evaluated using the F_2-measure and its generally worse performance when evaluated using the $F_{0.5}$-measure are not surprising, since we biased our system so that it classifies more data points as positive. In other words, it is expected that our system will correctly discover new positive examples that were not discovered by C5.0 and C5.0-Adaboost, but will incorrectly label as positive examples that are not positive. Overall, however, the results of our approach are quite positive with respect to both precision and recall. Furthermore, it is important to note that this method is not particularly computationally intensive. In particular, its computation costs are comparable to those of commonly used combination methods, such as AdaBoost.

Conclusion and Future Work

This paper presented an approach for dealing with the class-imbalance problem that consisted of combining different expressions of re-sampling based classifiers in an informed fashion. In particular, our combination system was built so as to bias the classifiers towards the positive set in order to counteract the negative bias typically developed by classifiers facing a higher proportion of negative than positive examples. The positive bias we included was carefully regulated by an elimination strategy designed to prevent unreliable classifiers to participate in the process. The technique was shown to be very effective on a drastically imbalanced version of a subset of the Reuters text classification task.

There are different ways in which this study could be expanded in the future. First, our technique was used in the context of a very naive oversampling and undersampling scheme. It would be useful to apply our scheme to more sophisticated re-sampling approaches such as that of (Kubat & Matwin 1997). Second, it would be interesting to find out whether our combination approach could also improve on cost-sensitive techniques previously designed. Finally, we would like to test our technique on other domains presenting a large class imbalance.

Acknowledgements

The authors would like to thank Rob Holte and Chris Drummond for their valuable comments. This research was funded in part by an NSERC grant. The work described in this paper was conducted at Dalhousie University.

References

Breiman, L. (1998): Combining Predictors, *Technical Report, Statistics Department, 1998.*

Domingos, Pedro (1999): Metacost: A general method for making classifiers cost sensitive, *Proceedings of the Fifth International Conference on Knowledge Discovery and Data Mining*, 155–164.

Estabrooks, Andrew (2000): *A Combination Scheme for Inductive Learning from Imbalanced Data Sets*, MCS Thesis, Faculty of Computer Science, Dalhousie University.

Hansen, L. K. and Salamon, P. (1990): Neural Network Ensembles, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(10), 993–1001.

Japkowicz, Nathalie (2000): The Class Imbalance Problem: Significance and Strategies, *Proceedings of the 2000 International Conference on Artificial Intelligence (IC-AI'2000),* 111–117.

Japkowicz, Nathalie, Myers, Catherine and Gluck, Mark (1995): A Novelty Detection Approach to Classification, *Proceedings of the Fourteenth Joint Conference on Artificial Intelligence*, 518–523.

Kubat, Miroslav and Matwin, Stan (1997): Addressing the Curse of Imbalanced Data Sets: One-Sided Sampling, *Proceedings of the Fourteenth International Conference on Machine Learning*, 179–186.

Kubat, Miroslav, Holte, Robert and Matwin, Stan (1997): Machine Learning for the Detection of Oil Spills in Satellite Radar Images, *Machine Learning, Volume 30*, 195–215.

Lewis, D. and Gale, W. (1994): Training Text Classifiers by Uncertainty Sampling, *Proceedings of the Seventh Annual International ACM SIGIR Conference on Research and Development in Information Retrieval.*

Ling, C. and Li, C. Data Mining for Direct Marketing: Problems and Solutions, *Proceedings of KDD-98.*

Pazzani, M., Merz, C., Murphy, P., Ali, K., Hume, T. and Brunk, C. (1994): Reducing Misclassification Costs, *Proceedings of the Eleventh International Conference on*

Machine Learning, 217–225.

Riddle, P., Secal, R. and Etzioni, O. (1991): Representation Design and Brute-Force Induction in a Boeing Manufacturing Domain, *Applied Artificial Intelligence, Volume 8*, 125–147.

Scott, Sam and Matwin, Stan (1999): Feature Engineering for Text Classification, *Proceedings of the Sixteenth International Conference on Machine Learning*, 379–388.

Shimshoni, Y. and Intrator, N. (1998): Classifying Seismic Signals by Integrating Ensembles of Neural Networks, *IEEE Transactions On Signal Processing, Special issue on NN.*

Weiss, S. and Kapouleas, I. (1990): An empirical comparison of pattern recognition, neural nets and machine learning methods, *Readings in Machine Learning*, J.W Shavlik and T.G. Dietterich (editors), Morgan Kauffman.

List of Figures

List of Tables