Studying Active Learning in the Cost-Sensitive Framework

Victor S. Sheng
Department of Computer Science
University of Central Arkansas
Conway, AR, USA
ssheng@uca.edu

Abstract
Active learning is a learning paradigm that actively acquires extra information with an “effort” for a certain “gain” when building learning models. This paper unifies the effort and gain by studying active learning in the cost-sensitive framework. The major advantage of studying active cost-sensitive learning aims at the business goal of minimizing the total cost directly, thus the potential applications of the proposed methods are significant. We first study a simple random active learner “buying” additional examples at random in order to reduce the total cost of example acquisition and future misclassifications. Then we propose a novel pool-based cost-sensitive active learner “buying” labels of unlabeled examples in a pool. We evaluate our new cost-sensitive active learning algorithms and compare them to previous active cost-sensitive learning methods. Experiment results show that our pool-based cost-sensitive active learner requires a fewer number of examples yet it produces a smaller total cost compared to the previous methods.

1. Introduction
Active learning, broadly speaking, is a learning program which actively acquires extra information with an effort for a certain gain in building learning models. The extra information is acquired from Oracle. It could be a missing feature value of a labeled example, or a class label for an unlabeled example. The learning program determines which information should be acquired to improve the performance (the “gain”) of the learning model the most. Active learning is widespread in building predictive models for real-world applications, for example, medical diagnosis. When we want to build an automatic medical diagnosis system to classify X-rays, we need to construct the training set first, which contains the series of X-rays and their diagnosis results (positive or negative). Obviously, it takes time for doctors to accurately label each X-ray positive or negative. Suppose there are a great amount of X-rays, we need to determine which X-rays we need their diagnosis results, and how many we need. If we make small efforts to just get a few diagnoses, the learning model built from these cases would perform not well. If we make great efforts to get a great amount of diagnoses, the learning model built from these cases would perform much better. Thus, we need to balance these two factors: “efforts” and “gains”.

Most previous works of active learning treat the “efforts” and “gains” in different forms and do not unify the two. For example, in terms of the “efforts”, the traditional “pool-based” active learner can select to label examples from a pool of unlabeled example. On the other hand, active learners with “feature-value acquisition” may acquire values of selected missing feature values in training or test examples. Similarly, the “gain” from the effort of active learning is also varied. For instance, accuracy and often the learning curves with accuracy have been used to show the usefulness (speedup) in active learning. In other work “deficiency” of active learning has also been explored. Indeed, Baram et al. [1] indicates that “there is no consensus on appropriate performance measures for active learning”.

A few recent works [8] [9] unify the two aspects (effort and gain) of active learning by studying active learning in the cost-sensitive framework (which itself has been extensively studied). That is, both efforts and gains are quantified with cost on the same scale (such as dollars), and active learning becomes the minimization of the total cost. This study further extends traditional active learning literature into cost-sensitive framework. We propose a novel method based on the version space for selecting examples to label. It adopts the total cost of example acquisition and misclassification as the evaluation criterion of the active learning.

Studying active learning in the cost-sensitive framework has many unique and important advantages. First of all, accuracy (as a performance measure) may not be appropriate in many real-world applications, as different kinds of errors (such as false positive and
false negative in binary classification) can cost very differently. Second, the performance of a learning program can be evaluated by many different measures: accuracy, recall, computation time, and so on. However, as long as they can be converted to the cost, they can be included in the total cost of the existing learning system. Third, as active learning is converted to the minimization problem, many optimization techniques (such as gradient descent) can be used. Finally, and most importantly, the ultimate goal of most real-world applications is to minimize the total cost (or maximize the total profit). Studying active learning in the cost-sensitive framework aims at the business goal directly. The potential applications of the proposed methods are significant.

It is true that it may be difficult to provide exact costs for the efforts and gains. However, as “efforts” and “gains” are two opposite measures, any tractable optimization combining the two has to make some explicit or implicit assumptions on the relative magnitude of the efforts and gains. More abstract terms such as time, love, and life may also be assigned dollar values [3]. In addition, we believe that exact cost values are unnecessary. We will show that small variations in cost assignments do not affect results much. We do make some simplifying assumption in this work: the total cost is simply the sum of the two (efforts and gains) in active learning. More complicated forms of cost aggregation will be considered in the future.

In this paper we assume that the extra examples or labels of examples to be acquired are the “effort”, and that reduction in misclassification costs in prediction is the “gain” in active learning. The active learner must attempt to strike a “tradeoff” between the efforts and gains by minimizing the total cost of acquiring examples and misclassifications. No other costs (such as computational resources, which are a part of the future work) are considered in this work.

More specifically we first study a simple case of active learning in the cost-sensitive framework: the random active learner which may “buy” or acquire additional random examples (with labels) at a cost. That is, the learner can decide to “buy” extra training examples given randomly. At a first glance, acquiring training examples at random is usually regarded as “passive learning” (not active learning). However, this can be regarded as a weak active learner who can decide if or not to buy extra examples. Studying this simple case also sets up a baseline to which the pool-based active learning algorithm can be compared.

Next we explore the standard pool-based active learning in the cost-sensitive framework. That is, the learner is given a pool of unlabeled examples, and it may “buy” labels of the selected unlabeled examples in the pool. We propose a novel active learning technique which selects unlabeled examples that can split and reduce the version space [11] most rapidly.

Applications of these two types of active learning are widespread, such as image classification and webpage categorization. An active learner of image classification can find out the right number of extra training examples (images) needed to buy or collect to reduce the total cost. In webpage categorization, the web pages found by a search engine form a pool of unlabeled examples. The active learner can select a small number of “critical” pages to be labeled (at the cost of the editors) in order to minimize the total cost to the search engine company.

2. Review of Previous Work

The most popular type of active learning is called “pool-based” active learning. A pool of unlabeled examples is given, and the learner can choose which ones to label during learning. Many works have been published in recent years on pool-based active learning, including, for instance, [16] [19] [13] [14] [1]1. However, almost all of these previous methods are evaluated by accuracy (or learning curves with accuracy) and “deficiency” of active learning [1], as well as the number of examples acquired. Margineantu [9] published a short paper on pool-based active cost-sensitive learner (called Active-CSL). Its goal is to choose an unlabeled example to label, which can provide the most expected gain in terms of the total cost (labeling and misclassification cost). More specifically, Active-CSL tried to estimate the gain by assigning different class labels to an unlabeled example. For each assigned class label, Active-CSL just summed all the costs obtained under each potential class label and chose the unlabeled example to label that produced maximum gain. We propose a novel, different method for selecting examples to label (based on the version space) and a new evaluation criterion (based on the actual cost reduction). We compare our pool-based active learner with the random active learner and Active-CSL in Section 5.2.

Cost-sensitive learning is also an active research topic in recent years. Turney [18] gives an excellent survey on a variety of costs that may be considered in learning. Most previous works on cost-sensitive

---

1 Our references are surely incomplete. We have only included several typical works published in recent years. See [1] for a good review of active learning approaches.
learning only consider minimizing misclassification costs (e.g., [17] [5] [6] [15]). However, the learning algorithms proposed are not active learners. That is, they assume that a fixed set of training examples is given, and the learners cannot acquire additional information during learning.

Our active cost-sensitive learning algorithm (to be presented in Section 3 and 4) utilizes the cost-sensitive decision tree (CSDT in short) as a base learning algorithm; thus we briefly review it here. CSDT [8] is similar to C4.5 [12], but it uses the total cost of misclassifications, instead of entropy, as the attribute split criterion. (In the original CSDT, there are also attribute costs, which we do not have, and are set to zero here). At each step, CSDT always chooses an attribute from the available attribute set with the maximum cost reduction on training data, similar to maximum entropy reduction, to build decision trees. Cost reduction \((E-E_A)\) is the difference between the expected cost \(E\) before splitting and the sum of the expected cost \(E_A\) after splitting with attribute \(A\). The attribute with the largest cost reduction, if it is greater than 0, is chosen to split the training data (otherwise it is not worth to build the branch further, and a leaf is formed). The same procedure is applied recursively to build subtrees. However, CSDT and C4.5 assume that all examples are available to build the tree.

### 3. Random Cost-Sensitive Active Learning

In this section we will describe our random active learning, called RanCA, using cost-sensitive decision trees as a base learner to establish the evaluation criteria and the baseline for the pool-based active learning.

#### 3.1. RanCA with Cost-Sensitive Decision Trees

The general idea of the RanCA is quite simple, and it can use any cost-sensitive learning algorithm as the base learner (as long as it produces probability estimates of the predictions; we use cost-sensitive decision tree CSDT reviewed in Section 2 in RanCA). The framework of RanCA is an incremental learner. That is, it acquires random examples at a cost gradually while monitoring an evaluation criterion (see Section 3.2.) until it is met. The evaluation computes the sum of the acquisition cost and misclassification cost of test examples to see when it reaches the minimum.

More specifically, each time, the RanCA acquires one or \(m\) examples (adding one example each time often has too little effect so \(m\) can be greater than 1) randomly, and includes them in the training set. Then a new cost-sensitive decision tree is built from the expanded training set, and is evaluated to see if the total cost of the example acquisition and misclassification of test examples is reduced. If it is, then the process repeats (more examples are acquired); if not, the learner stops acquiring more examples, and the current decision tree is produced.

Note that even though RanCA calls for an evaluation process for each example acquired, it is more “conservative” and accurate than estimating the examples needed, as it uses the actual cost obtained so far directly. In addition, the base learner is an efficient decision tree algorithm, and computation resource is not part of the cost in our paper.

We will discuss the evaluation method in the next subsection.

### 3.2. Evaluation Methods

To evaluate the trees built in the random (and later pool-based) active learners for future test performance, we can use a (small) part of training examples as (future) test examples. These test examples are not used in training. However, from the active learning perspective which attempts to minimize the total cost of acquiring training examples and future misclassification on test examples, holding out some examples for testing excludes them from building the model, making some acquiring examples wasted. To reduce the waste of acquired training examples used for testing, we use leave-one-out cross-validation (LOO in short) to evaluate the learned model, so only one example is “wasted”. As the decision tree learning algorithm is quite efficient, LOO does not pose a major concern in computation time. For example, each iteration of example acquisition via LOO on a typical dataset (such as Thyroid; see Section 5) takes only 71 seconds on average on a modern PC. (Further, the computational cost is not considered in the total cost in this paper.)

The following procedure describes details on how to estimate the misclassification cost of one test example (in LOO). For binary classification (used in this paper), we use the following notations: \(TP\) is the cost of true positive, \(TN\) is the cost of true negative, \(tp\) and \(fn\) are the number of positive examples, and \(tn\) and \(fp\) are the number of true negative examples. Both \(TP\) and \(TN\) usually are negative values (benefits of correct classification; usually set as 0); \(FP\) and \(FN\) are positive values (misclassification costs). For a leaf in the cost-sensitive decision tree, \(C_p = tp\times TP + fp\times FP\) is the total misclassification cost of being a positive leaf, and \(C_n = tn\times TN + fn\times FN\) is the total misclassification cost of being a negative leaf. Then the probability of a
leaf being positive is estimated by the relative cost of \( C_P \) and \( C_N \); the smaller the cost, the larger the probability (as minimum cost is sought). Thus, the probability of the leaf being positive is thus: 
\[
1 - \frac{C_P}{C_P + C_N} = \frac{C_N}{C_P + C_N}.
\]
Similarly, the probability of a leaf being a negative is 
\[
\frac{C_P}{C_P + C_N}.
\]
However, these probabilities are not used directly in estimating misclassification costs, because the number of training examples in leaves is usually very small, especially at the beginning of example acquisition. To reduce the effect of extreme probability estimations, we apply the Laplace correction \([7] [4]\) to smooth probability estimates. 

For example, acquisition. To reduce the effect of extreme probability estimates, we apply the Laplace correction \([7] [4]\) to smooth probability estimates. The original Laplace correction for accuracy can be expressed as 
\[
\frac{n_C + 1}{n + n_C},
\]
where \( n_C \) is the number of examples which belong to class \( C \), \( n \) is the number of training examples, and \( n_C \) is the number of classes. As we consider misclassification costs now, the Laplace-corrected probability of a leaf being positive is 
\[
\frac{C_P + \lambda}{C_N + C_P + k},
\]
where \( \lambda = |FP - FN| \) and \( k = FP + FN \). Similarly, the probability of a leaf being negative is 
\[
\frac{C_P + \lambda}{C_N + C_P + k}.
\]
Thus, the expected misclassification cost of a true negative example is 
\[
\frac{C_N + \lambda}{C_P + C_N + k} \times FN,
\]
and the expected misclassification cost of a true positive example is 
\[
\frac{C_P + \lambda}{C_P + C_N + k} \times FP.
\]

The next question is how to integrate the misclassification cost of one test example (obtained above) with the cost of training examples acquired. A simple sum of the two, as in \([9]\), would not be reasonable, as it depends on how many future test examples (or how often) the model will be used to predict. Clearly, if the model built will be seldom used (only once or twice), we can reduce the total cost through building a rough model with acquiring only a few examples. On the other hand, if the model built will be used very frequently (for instance, millions of times), it would be worthwhile to acquire more examples to build a highly accurate model with very low misclassification cost. As we may not know the number of future test examples during the model building process, we introduce a variable \( t \) to represent the number of future test examples. Thus, the total cost on test examples is the misclassification cost of one test example obtained by LOO multiplied by \( t \). The total cost is simply the sum of the total cost on \( t \) test examples and the cost of the so-far acquired examples.

The next question is how to use the total cost as a stopping criterion for the incremental cost-sensitive active learners. Ideally, we can expect the total cost to decrease initially, and it would then reach a minimum before it goes up. This is because the gain in reduced misclassification cost would tail off as more and more examples or labels are acquired (with a constant cost for each). Thus, we can obtain a learning curve in terms of the total cost and the number of examples required (see Figure 2). If the curve is smooth, then indeed the learning algorithm has found the optimal number of training examples needed. However, as we will see in Section 5, the curve of the total cost may not be smooth, and the local minimum may not be the global one. Thus it is necessary to “look ahead” and acquire a few more (units of) examples to ensure that the local minimum is a global one. The extra examples in LOO and look-ahead will be extra (or wasted) on top of the minimum total cost found. We will describe the look-ahead strategy obtained empirically in Section 5.1.

4. Pool-based Cost-Sensitive Active Learning

In this section we propose a novel pool-based cost-sensitive active learner, called PoolCA, which selects examples from the pool of unlabeled examples for labeling at a cost. The basic framework of PoolCA is the same as RanCA: it is also an incremental learner that uses the same evaluation criterion. However, PoolCA chooses example(s) to be labeled based on the version space \([11]\), which consists of all possible hypotheses (trees) consistent with labeled examples acquired so far. More specifically, PoolCA will attempt to split (and reduce) the version space by half (as rapidly as possible) after examples are labeled. The basic idea has been proposed and implemented with SVM \([16]\) but not with decision trees. The main advantage of using decision trees is its much higher efficiency in the tree building compared to the SVM learning.

For decision trees, the version space is the set of all possible cost-sensitive decision trees built on the available (labeled) training examples. However, as the version space may contain a very large number of trees, we “sample” the version space by building a small committee of trees. In this paper, we build 50 trees as a committee of trees to represent the version space.

To make the trees built be representative of small cost-sensitive trees, CSDT is modified to have some random component in attribute selection. While the
original CSDT chooses the attribute with the maximum cost reduction deterministically, the modified version chooses an attribute with a probability proportional to the cost reduction (at every stage of the tree building). This way, less favorable attributes may still be chosen in the tree building, and a set (or committee) of different but small cost-sensitive decision trees are built from the same training set.

After the committee of 50 trees is built, PoolCA uses them to predict the class label (with probability estimates) of each unlabeled example. Then each unlabeled example is assigned an uncertainty score, based on the predictions of the 50 trees. Specially, if \( P_i \) is the probability of being class 1 (estimated by one of the 50 trees), then weighted probability of being class 1 of this example is \( P = \frac{\sum P_i}{50} \). Then the uncertainty score of this example is: \( 1 - \max\{P, 1 - P\} \). Thus, when \( P \) is 0.5, the prediction of the example by the committee is most uncertain, and the uncertainty score is the largest.

Then PoolCA chooses \( m \) examples from unlabeled set according to the uncertainty scores. Instead of selecting the \( m \) examples with the highest uncertainty scores, sampling is used to introduce some varieties in the chosen examples. That is, examples are repeatedly sampled according to the probability proportional to their uncertain scores, until \( m \) examples are obtained.

After the chosen examples are labeled (at a cost) and added into the training set, the same evaluation criterion (Section 3.2.) will be applied. The pseudo code of PoolCA is presented in Figure 1.

**Algorithm PoolCA**

**Input:** an initial labeled training set \( T \) (can be empty), an unlabeled set \( U \), the cost-sensitive learner CSDT, and the sample size \( m \).

**Repeat**  
1. Apply the randomized CSDT on the training set \( T \), build 50 cost-sensitive decision trees.  
2. Compute uncertainty scores for all examples in \( U \).  
3. Probabilistically sample \( m \) examples according to the uncertainty scores from \( U \).  
4. Request labels, add them into \( T \), delete them from \( U \).

**Until** the stopping criterion is met

**Output:** the final cost-sensitive decision tree

---

**Table 1. Summary of the differences among the compared models (RanCA, Active-CSL, and PoolCA).**

<table>
<thead>
<tr>
<th>Differences</th>
<th>RanCA</th>
<th>Active-CSL</th>
<th>PoolCA</th>
</tr>
</thead>
<tbody>
<tr>
<td>Base learning algorithm</td>
<td>Cost-sensitive decision tree (CSDT)</td>
<td>Bagged probability estimation trees</td>
<td>Cost-sensitive decision tree (CSDT)</td>
</tr>
<tr>
<td>Criterion of choosing examples to acquire labels</td>
<td>Uncertainty Scoring based on single CSDT</td>
<td>Expected high risk based on</td>
<td>Uncertainty Scoring given by a committee of CSDTs (version space)</td>
</tr>
<tr>
<td>Time complexity</td>
<td>Fastest: building one decision tree model for one selection</td>
<td>Very slow: building ( 2 \times</td>
<td>U</td>
</tr>
</tbody>
</table>

---

5. **Experiments**

We conduct experiments with our cost-sensitive active learning algorithms on 10 real-world datasets from the UCI Machine Learning Repository [2]. These datasets are chosen because they have at least some discrete attributes and binary class. In addition, these datasets are also used in our previous work.

The datasets are listed in Table 2. As misclassification costs of these datasets are not available, we assign values in a reasonable range as costs, following [5] [8] [20]. This is fair and reasonable as all experimental comparisons are conducted with the same cost assignments. In this paper we assign random numbers from 500 to 2000 as the cost of a random example or the label of an example in the pool, and \( FP/FN = 2000/6000 \) (we assume that \( TP=TN=0 \)). We have tried other cost assignments and results are very similar.
Table 2. Features of the 10 Datasets used in the experiments.

<table>
<thead>
<tr>
<th>#Attributes</th>
<th>#Examples</th>
<th>Class dist.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ecoli</td>
<td>6</td>
<td>332 230/102</td>
</tr>
<tr>
<td>Breast</td>
<td>9</td>
<td>683 444/239</td>
</tr>
<tr>
<td>Heart</td>
<td>8</td>
<td>161 98/163</td>
</tr>
<tr>
<td>Thyroid</td>
<td>24</td>
<td>2000 1762/238</td>
</tr>
<tr>
<td>Australia</td>
<td>15</td>
<td>653 296/357</td>
</tr>
<tr>
<td>Tic-tac-toe</td>
<td>9</td>
<td>958 332/626</td>
</tr>
<tr>
<td>Mushroom</td>
<td>21</td>
<td>8124 4208/3916</td>
</tr>
<tr>
<td>Kr-vs-kp</td>
<td>36</td>
<td>3196 1527/1669</td>
</tr>
<tr>
<td>Voting</td>
<td>16</td>
<td>232 108/124</td>
</tr>
<tr>
<td>Cars</td>
<td>6</td>
<td>446 328/118</td>
</tr>
</tbody>
</table>

5.1. Experiments with RanCA

We first conduct experiments for the random active learner RanCA (Section 3.1.). The experiment results are shown in Figure 2 which displays the learning curve for a typical dataset (Thyroid). The unit $m$ is chosen to be 10 in all experiments. The horizon axis is the number of acquired examples, and the vertical axis is the total cost, which is the sum of the cost of acquisition examples and the average misclassification cost of $t$ future test examples, assuming $t$ is 1,000 here (other values of $t$ have been tried and resulted in similar curves).

From Figure 2, we can see clearly that the LOO curve produced by RanCA has a desirable trend: it goes down at the beginning (after a few units of training examples are acquired), and then it goes up after a certain point, forming a global minimum (the minimal total cost). All datasets produce similar curves as shown in Figure 2, except Cars, which has no global minimum – the curve keeps on going lower. This can happen when the available training set is not large enough to witness the global minimum to happen.

![Figure 2. Total cost curve with RanCA for a typical dataset Thyroid.](image)

5.2. Experiments with PoolCA

We conduct experiments for PoolCA and compare its results with RanCA (the baseline) and Active-CSL [9]. The results are shown in Figure 3. Note that the segments of the curves after the global minima are computed and plotted for demonstrating the incensement of the total cost only. Note that the segment of Active-CSL after its global minima is not computed and plotted completely for the four datasets (Breast, Tic-tac-toe, Mushroom, and Kr-vs-kp), just because Active-CSL is very slow. From Figure 3, we can see clearly that the cost curves of all cost-sensitive active learners have a similar trend: they go down at the beginning after a few units of training examples are acquired, and then they go up, forming global minima. Overall, PoolCA has the lowest (and best) total cost performance. Active-CSL does not always perform better than the random active learner (RanCA). It performs worse than RanCA on the two datasets Thyroid and Australia.

The minimum total costs and the number of units of examples acquired for the 10 datasets are summarized in Table 4. Again the total costs shown have included the 2 units of look-ahead in search of the global minima.

From Table 4, we can draw several interesting and surprising conclusions. First of all, we can see that PoolCA reaches the global minimum with fewer units of examples on average than RanCA (14.0 vs. 16.1). It also requires fewer units of examples than Active-CSL (14.0 vs. 16.1). However, PoolCA produces a lower average total cost than RanCA and Active-CSL. This indicates that by choosing fewer “critical” examples to label, PoolCA can produce better results (lower total costs) than the random learner RanCA with more examples. It is a good indication that noisy examples

Table 3. The minimum length of look-ahead needed for ACTree to find the true global minimum in the LOO curve for each dataset.

<table>
<thead>
<tr>
<th>LOO</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Ecoli</td>
<td>2</td>
</tr>
<tr>
<td>Breast</td>
<td>2</td>
</tr>
<tr>
<td>Heart</td>
<td>2</td>
</tr>
<tr>
<td>Thyroid</td>
<td>1</td>
</tr>
<tr>
<td>Australia</td>
<td>2</td>
</tr>
<tr>
<td>Tic-tac-toe</td>
<td>2</td>
</tr>
<tr>
<td>Mushroom</td>
<td>2</td>
</tr>
<tr>
<td>Kr-vs-kp</td>
<td>3</td>
</tr>
<tr>
<td>Voting</td>
<td>2</td>
</tr>
<tr>
<td>Cars</td>
<td>1</td>
</tr>
</tbody>
</table>
were not chosen by PoolCA (it is our future research to verify this).

<table>
<thead>
<tr>
<th></th>
<th>RanCA</th>
<th>Active-CSL</th>
<th>PoolCA</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ecoli</td>
<td>326.9/16</td>
<td>322.9/14</td>
<td>306.4/13</td>
</tr>
<tr>
<td>Breast</td>
<td>604.0/22</td>
<td>570.7/27</td>
<td>576.5/16</td>
</tr>
<tr>
<td>Heart</td>
<td>1074.2/13</td>
<td>1102.4/17</td>
<td>1091.9/17</td>
</tr>
<tr>
<td>Thyroid</td>
<td>529.1/7</td>
<td>576.4/8</td>
<td>474.6/10</td>
</tr>
<tr>
<td>Australia</td>
<td>876.4/16</td>
<td>898.4/11</td>
<td>865.0/10</td>
</tr>
<tr>
<td>Tic-tac-toe</td>
<td>1251.3/20</td>
<td>1217/26</td>
<td>1135.1/23</td>
</tr>
<tr>
<td>Mushroom</td>
<td>516.5/17</td>
<td>492.9/13</td>
<td>492.1/15</td>
</tr>
<tr>
<td>Kr-vs-kp</td>
<td>738.4/14</td>
<td>722.8/13</td>
<td>714.1/7</td>
</tr>
<tr>
<td>Voting</td>
<td>375.7/8</td>
<td>377.5/7</td>
<td>378.9/11</td>
</tr>
<tr>
<td>Car</td>
<td>876.2/28</td>
<td>866.7/25</td>
<td>865.7/18</td>
</tr>
<tr>
<td>Average</td>
<td>716.9/16.1</td>
<td>714.8/16.1</td>
<td>688.6/14.0</td>
</tr>
</tbody>
</table>

Second, if we look at individual datasets, PoolCA produces smaller total costs in 8 out of 10 datasets compared to RanCA, and in 7 out of 10 datasets compared to Active-CSL. On the datasets in which PoolCA loses, the difference in total costs is usually small. Third, on average, Active-CSL does perform slightly better than RanCA; it performs better than RanCA (in the total cost) in 6 out of 10 datasets. To sum, we can conclude that PoolCA performs much better than random active learner (RanCA) and the previous method of Active-CSL.

5.3. Varying the Committee Size Sampling the Version Space

In previous subsection, we conduct experiments on PoolCA with 50 trees as a committee to sample the version space. In this subsection, we investigate the performance of PoolCA under different committee sizes on a typical data (Thyroid here). The committee size (CS in short) varies from 20 to 200, and the experimental results are shown in Figure 4. From Figure 4, we can see that PoolCA performs better when the committee size increases from 20 to 50. However, the performance becomes very close when the committee size further increases. As PoolCA will become slower when the committee size increases, we choose 50 as the optimal committee size for the efficiency and effectiveness, as used in PoolCA.

![Figure 4: Varied committee sizes of version space.](image)
Table 5 and 6 respectively, and summarized in Figure 5 and 6. We can make the following conclusions.

**Table 5. Summary of the global minima and the number of iterations (displayed in each cell separated by /) for each dataset under the three active learning methods with the assumption the number of future test examples t=500.**

<table>
<thead>
<tr>
<th></th>
<th>RanCA</th>
<th>Active-</th>
<th>PoolCA</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ecoli</td>
<td>396.3/11</td>
<td>389.5/12</td>
<td>373.2/13</td>
</tr>
<tr>
<td>Breast</td>
<td>730.9/14</td>
<td>700.9/16</td>
<td>682.9/13</td>
</tr>
<tr>
<td>Heart</td>
<td>1158.6/13</td>
<td>1202.1/17</td>
<td>1160.0/10</td>
</tr>
<tr>
<td>Thyroid</td>
<td>685.2/6</td>
<td>678.2/6</td>
<td>580.3/6</td>
</tr>
<tr>
<td>Australia</td>
<td>988.3/10</td>
<td>1018.2/11</td>
<td>955.5/6</td>
</tr>
<tr>
<td>Tic-tac-toe</td>
<td>1386.7/18</td>
<td>1383.5/12</td>
<td>1287.2/20</td>
</tr>
<tr>
<td>Mushroom</td>
<td>633.1/7</td>
<td>632.8/9</td>
<td>580.3/6</td>
</tr>
<tr>
<td>Kr-vs-kp</td>
<td>968.0/6</td>
<td>986.3/12</td>
<td>852.0/7</td>
</tr>
<tr>
<td>Voting</td>
<td>458.2/6</td>
<td>443.6/4</td>
<td>458.5/5</td>
</tr>
<tr>
<td>Car</td>
<td>966.5/13</td>
<td>960.8/11</td>
<td>958.2/18</td>
</tr>
<tr>
<td>Average</td>
<td>837.2/10.4</td>
<td>839.6/11.0</td>
<td>793.7/10.6</td>
</tr>
</tbody>
</table>

**Table 6. Summary of the global minima and the number of iterations (displayed in each cell separated by /) for each dataset under the three active learning methods with the assumption the number of future test examples t=2,000.**

<table>
<thead>
<tr>
<th></th>
<th>RanCA</th>
<th>Active-</th>
<th>PoolCA</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ecoli</td>
<td>280.7/20</td>
<td>278.6/20</td>
<td>265.8/17</td>
</tr>
<tr>
<td>Breast</td>
<td>526.1/32</td>
<td>474.7/27</td>
<td>483.1/26</td>
</tr>
<tr>
<td>Heart</td>
<td>1032.1/13</td>
<td>1047.2/17</td>
<td>1036.7/17</td>
</tr>
<tr>
<td>Thyroid</td>
<td>463.7/14</td>
<td>483.6/16</td>
<td>402.0/10</td>
</tr>
<tr>
<td>Australia</td>
<td>789.3/16</td>
<td>826.0/15</td>
<td>796.2/10</td>
</tr>
<tr>
<td>Tic-tac-toe</td>
<td>1116.8/49</td>
<td>1124.6/26</td>
<td>1053.4/23</td>
</tr>
<tr>
<td>Mushroom</td>
<td>390.4/25</td>
<td>374.4/23</td>
<td>374.6/21</td>
</tr>
<tr>
<td>Kr-vs-kp</td>
<td>584.1/25</td>
<td>563.8/17</td>
<td>589.9/13</td>
</tr>
<tr>
<td>Voting</td>
<td>327.4/10</td>
<td>319.9/11</td>
<td>314.5/11</td>
</tr>
<tr>
<td>Car</td>
<td>801.6/31</td>
<td>797.5/30</td>
<td>799.8/29</td>
</tr>
<tr>
<td>Average</td>
<td>631.2/23.5</td>
<td>629.0/20.2</td>
<td>611.6/17.7</td>
</tr>
</tbody>
</table>

**Figure 5. Summarization of minimum total costs under different number of future test examples.**

From Figure 5, we can conclude that the minimum average total cost of the three methods decreases when the size of future test set increases. This is expected, as the share of the cost of acquiring training examples, which each test example burdens, decreases when the number of future test examples increases.

**Figure 6. Summarization of the number of iterations needed to achieve the minima under different number of future test examples.**

Figure 6 shows that the number of training examples acquired to achieve the minimum total cost increases when the size of future test set increases. This is also expected, as the more the number future test examples, the more accurate model should be built. Thus, more training examples are required.

From Figure 5 and 6 above, we can also conclude that PoolCA is best, followed by Active-CSL, followed by RanCA. Furthermore, in the aspect of the minimum total cost, the difference between PoolCA and Active-CSL increases when the size of future test set decreases. Active-CSL is very close to RanCA. In the aspect of the number of training examples acquired PoolCA acquires fewer examples than Active-CSL and RanCA. And the difference of acquired examples by PoolCA and Active-CSL increases when the size of future test set increases. However, Active-CSL acquires fewer examples than RanCA.

**5.5. Varying Cost Assignments**

As mentioned in the Introduction, one possible criticism of our work is that costs of various types must be assumed or obtained. However, we believe that accurate cost assignments are not necessary, as small variations of costs will not affect the discrete decision (the number of units of examples) dramatically. To investigate this, we conduct experiments using RanCA under the same setting, except that we inject a small variation (5%) to the cost of a random example, false positive cost ($FP$), and false negative cost ($FN$). The results on a typical dataset (Thyroid here) are shown in Figure 7. Three curves in the figure are from the cost variations of each cost ($example cost, FP, FN$) independently, and the “All” curve is from injecting noise to all three costs simultaneously. We can see clearly that all the curves have a very similar trend. In addition, they all reach their global minimum after acquiring a very similar number of examples. The
global minimums are also very close to each other. We can thus conclude that small variations in the exact values of cost assignments do not affect much of the results.

![Figure 7. Varying cost assignments in a typical dataset Thyroid.](image)

6. Conclusions and Future Work

In this paper we study active learning in the cost-sensitive learning framework to achieve a single goal: minimizing the sum of the cost of “effort” and “gain”. More specifically, we study several active learners which may acquire additional examples randomly at a cost, or class labels from the pool of unlabeled examples at a cost, to minimize the total cost of example acquisition and future misclassifications. We compare these new strategies with previous work and conclude that the pool-based active learner (PoolCA) achieves a smaller total cost with fewer numbers of examples than the random active learner RanCA and the previous work Active-CSL.

In our future work, we plan to study other forms of active learning (such as stream-based and membership query) in the cost-sensitive learning framework. We also plan to study the line search optimization method to speed up the search for the optimal number of examples to acquire for the minimal total cost.

7. Acknowledgement

We thank the anonymous reviewers for the valuable comments. The work was supported by the National Science Foundation (IIS-1115417).

8. References


