From Active to Proactive Learning Methods

Pinar Donmez and Jaime G. Carbonell

Language Technologies Institute
School of Computer Science
Carnegie Mellon University
{pinard,jgc}@cs.cmu.edu

1 Active Learning

1.1 Motivation

In many machine learning tasks, unlabled data abounds, but expert-generated labels are scarce. Consider the process of learning to build a classier for the Sloan Digital Sky Survey (http://www.sdss.org/) so that each astronomical observation may be assigned its class (e.g. "pinwheel galaxy", "globular galaxy", "quasar", "colliding galaxies", "nebula", etc.). The SDSS contains 230 million astronomical objects, among which professional Astronomers have classified manually less than one tenth of 1 percent. Consider classifying web pages into subject-matter based taxonomies, such as the Yahoo taxomy or a Dewy library catalog system. Whereas there are many billions of web pages, less than .001% have reliable topic or subject categories.

In both cases, despite of the massive quantities of unlabeled data, some categories have very few reliably labeled examples, but more are needed to to train machine learning systems to classify with sufficent accuracy. However, obtaining expert labels can prove to be expensive, bounded (e.g. Astronomers will only give so much of their time), or both. Hence, the question becomes one of selecting the most informative instances to label; some labeled examples can be much more useful than others in training classiers to minimize errors. For instance, selecting an example representative of many others, or selecting an example in an unexplored region of the instance space, or selecting an example close to the decision boundary induced from the earlier labeled examples are all better strategies than random selection. Active learning is the process of selecting the best examples to label in order to improve classifer performance, where "best" is typially defined as minimizing the loss function [Miller et al., 1993] of the classifier or otherwise directly minimizing its estimated classification error.

Alternatives to active learning attempt to use unlabeled data directly in helping to training classifiers, without obtaining additional labels. These efforts range from transductive support vector machines [Joachims, 1999] to co-training [Blum and Mitchell, 1998] and other semi-supervised methods [He et al., 2007]. Although effective in some cases, transductive and semi-supervised methods cannot in general rival active learning since they do not take advantage of additional labeled examples. Combining ideas from both semi-supervised and active learning remains a largely unexplored area for future research.

J. Koronacki et al. (Eds.): Advances in Machine Learning I, SCI 262, pp. 97–120.
springerlink.com

1.2 Active Learning Methods in Literature

Although instance selection has been implicitly part of Machine Learning from its early years, such as in Michalski's Star methodology and related work [Michalski et al., 1983; Reinke and Michalski, 1985; Michalski et al., 1986], the systematic study of active learning has flourished and generated considerable interest in the past 15 years, including the emergence of a solid thoeretical foundation for the field [Cohn et al., 1995]. Whereas there have been several studies investigating the use of active learning in classification, rank learning, regression, and function optimization, in this chapter, we mainly focus on the dominant and most studied variant: active learning for classification.

The most straightforward strategy for selecting examples to label is random sampling among unlabeled instances, which is usually adopted as a baseline for comparative analysis since it often outpeforms other naive approaches (such as representative sampling). We group active learning methods into three major categories based on their underlying sampling criteria; namely, 1) uncertainty-based sampling strategies, 2) density-based strategies and 3) ensemble methods. We acknowledge that this is not an exhaustive list, rather a summary grouping of the best performing and most widely used active learning algorithms in the literature. Uncertainty sampling was proposed by [Lewis and Gale, 1994] by choosing to obtain a label for the example on which the current learner is most uncertain. Tong and Koller, 2000 [Tong and Koller, 2000] adopted the same idea and applied it to a support vector machine (SVM) classifier. They provided a theoretical motivation based on shrinking the version space (the margin) as much as possible; in other words, selecting the example that will minimize the maximum expected size of the residual version space. In case of linear discriminant functions, this corresponds to the selection of the unlabeled example with the smallest margin [Tong and Koller, 2000]. Despite the theoretical justification of the version-space reduction methods, their method wastes lableing effort attempting to eliminating areas of the parameter space that have no direct effect on the classification error rate. Hence, these methods are not immune to selecting outliers [McCallum and Nigam, 1998] since they have high uncertainty, but outliner labels seldom help the learner improve its generalization performance. Nevertheless, uncertainty sampling plays a key role in many studies in the literature, including our previous work which is explained in detail in Section 1.3.

The density-based sampling strategies incorporate the underlying data distribution into the selection mechanism, and thus indirectly borrow from semi-supervised methods cited above. Most of these methods propose ways to trade off an uncertainty measure with the density of the instance space in the immediate neighborhood of the sample. Xu et al., 2003 [Xu et al., 2003] uses k-means to cluster the samples lying inside the margin of an SVM classifier trained on the current labeled data. The cluster centroids are then selected for labeling. Shen and Zhai, 2005 [Shen and Zhai, 2005] adopts the same idea in an information retrieval scenario, and uses the k-medoid algorithm for the top relevant examples. Similar active learning schemes are proposed by [Tang et al., 2002] for natural language parsing, and by [Zhang and Chen, 2002] for content-based information retrieval. In [McCallum and Nigam, 1998], a naive Bayes classifier is trained on both labeled and unlabeled data using an EM algorithm. Under the assumption that unlabeled data dominates the labeled data, the training algorithm clusters the full dataset but labeled data is used only to seed the clusers. Clustering information, then, contributes to

the selection such that an uncertainty measure is weighed with the density surrounding the example. Another active learning approach that utilizes combines uncertainty and density criteria is our paired sampling strategy described in [Donmez and Carbonell, 2008]. The difference between our method and the density-based methods introduced above is that our method relies on balanced sampling on both sides of a decision boundary rather than sampling disproportionately on one side. Furthermore, it exploits the natural grouping (clustering) of the data to effectively define a non-Eucledian local distance function, and maximizes a utility-based conditional entropy criterion for sampling. We have shown that this new density-sensitive method yields significantly superior performance over multiple datasets compared to other popular active learning sampling methods [Donmez and Carbonell, 2008].

Nguyen and Smeulders, 2004 [Nguyen and Smeulders, 2004] suggested a probabilistic framework where clustering is combined with a discriminative model. They assume that examples lying on the classification boundary, in other words most uncertain examples, are informative, but using information about the underlying data distribution helps to select better examples. Their method favors higher density examples lying close to the decision boundary. Those examples are assumed to have the largest contribution to the current error. Though their strategy works well in practice, especially by reducing the error quickly, it exhibits very slow additional learning after substantial sampling. On the other hand, standard uncertainty sampling initially has a slower learning rate, but gradually outperforms their method, as shown in our previous work [Donmez et al., 2007]. The reason is that density based methods sample from maximal-density unlabeled regions, and thus help establish the initial classification boundary where it affects the most remaining unlabeled data. On the other hand, uncertainty sampling fine-tunes a decision boundary by sampling the regions where the classifier is least certain, regardless of the distribution of the unlabeled data. Our algorithm, DUAL [Donmez et al., 2007], tackles this issue by proposing a principled ensemble-based approach that selects which sampling method to apply based on estimated residual classification error reduction. DUAL incorporates a robust combination of density weighted and uniformly weighted uncertainty sampling. We have empirically shown that DUAL combines the best of both worlds, and leads to superior performance across various domains [Donmez et al., 2007].

Another ensemble learning method is proposed by [Baram et al., 2003]. Baram et al., 2003 presented an online algorithm (COMB) that selects among three alternative active learning strategies using a variant of the multi-armed bandit algorithm to decide the strategy to be used at each iteration. Although there are similarities between DUAL and COMB, there is a major distinction: COMB aims to select which sampling method is optimal for a given dataset, irrespective of how much the algorithm has already learend, whereas DUAL focuses on selecting the operating range most suitable for the sampling method. Other ensembled active learning methods have appeared in literature. [Melville and Mooney, 2003] extends the query-by-committee algorithm by constructing diverse committees by employing artificial training examples. Co-testing [Muslea et al., 2000] is another ensemble active learning strategy inspired by the multi-view approach in co-training [Blum and Mitchell, 1998]. It utilizes two redundant views of the training data to create an ensemble and selects the unlabeled instance where two classifiers disagree.

1.3 Addressing Performance Barriers

1.3.1 DUAL: A Context-Sensitive Strategy Selection

DUAL [Donmez et al., 2007] is a principled ensemble-based sampling approach for active learning. It combines the uncertainty sampling and density-based sampling by selecting sampling methods based on estimated residual classification error reduction. Almost all active learning strategies that trade-off uncertainty vs. data density aim to balance the uncertainty of the sample with its representativeness, but do so in a fixed manner, rather than by dynamically selecting or reweighing, based on residual error estimation. In other words, prior methods do not estimate the learning rate of the classifier, and therefore have no criterion to switch sampling strategy if one starts to saturate in performance.

DUAL is a context-sensitive sampling method with a primary focus on improving active learning for both the onset of active learning and especially the later portion of the process, rather than traditional methods that concentrate primarily on the initial label selection and do not adapt. DUAL significantly improves upon the work of [Nguyen and Smeulders, 2004] by incorporating a robust combination of density weighted uncertainty sampling and standard (uniform) uncertainty sampling. Nguyen and Smeulders, 2004 [Nguyen and Smeulders, 2004] propose a Density Weighted Uncertainty Sampling criterion, or DWUS for short:

$$s = \arg\max_{i \in I_u} E[(\hat{y}_i - y_i)^2 \mid x_i] p(x_i) \tag{1}$$

where $E[(\hat{y}_i - y_i)^2 \mid x_i]$ and $p(x_i)$ are the expected error and density of a given data point x_i, respectively. I_u is the index for the unlabeled data. This criterion favors points that have the largest contribution to the current classification error. In contrast, one can use an uncertainty-based selection criterion within the same probabilistic framework as illustrated by the following formula:

$$s = \arg\max_{i \in I_u} E[(\hat{y}_i - y_i)^2 \mid x_i] \tag{2}$$

We refer to the above principle as Uncertainty Sampling. Consider Fig. 1, which displays the performance of DWUS and Uncertainty Sampling on two UCI datasets. After rapid initial gains, DWUS exhibits very slow additional learning while uncertainty sampling continues to exhibit more rapid improvement.[1] We have investigated the behaviour of unlabeled data in these two scenarios to explore the underlying reasons for such a pattern. We have observed that at the early iterations, many points are highly uncertain. Thus, DWUS can pick high density points which are lower down in the uncertainty *ranking* but have a high absolute uncertainty score. Later, points with high absolute uncertainty are no longer in dense regions. As a result, DWUS picks points that have moderate density but low uncertainty because such points are scored highly according to the criterion in Equation 1. Hence, it wastes effort picking instances whose selection does not have a large effect on error rate reduction.

In order to address this problem, our algorithm DUAL adopts a dynamically re-weighted mixture of density and uncertainty components. DUAL works as follows:

[1] Although a quick drop in classification error for DWUS is also observed in [Nguyen and Smeulders, 2004], they did not compare with uncertainty sampling.

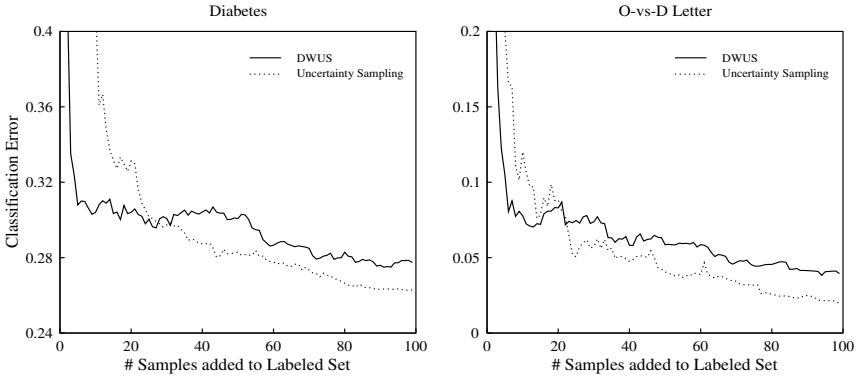

Fig. 1. Comparison of Density Weighted versus (standard) uniformly weighted Uncertainty Sampling on two UCI benchmark datasets

It starts executing DWUS[2] up until it estimates a cross-over point with uncertainty sampling by predicting a low derivative of the expected error, e.g. $\frac{\partial \epsilon(DWUS)}{\partial x_t} \leq \delta$. The derivative estimation need not be exact, requiring only the detection of diminishing returns which we explain soon. Then, it switches to execute a combined strategy of density-based and uncertainty-based sampling. In practice, we do not know the future classification error of DWUS, but we can approximate it by calculating the average expected error of DWUS on the unlabeled data. It will not give us the exact cross-over point, but it will provide a rough estimate of when we should consider switching between methods. The expected error of DWUS on the unlabeled data can be evaluated as follows:

$$\hat{\epsilon_t}(DWUS) = \frac{1}{n_t} \sum_{i \in I_u} E[(\hat{y}_i - y_i)^2 \mid x_i] \tag{3}$$

where $E[(\hat{y}_i - y_i)^2 \mid x_i]$ is the error expectation for a given unlabeled data point x_i:

$$E[(\hat{y}_i - y_i)^2 \mid x_i] = (\hat{y}_i - 1)^2 P(y_i = 1 \mid x_i) + (\hat{y}_i)^2 P(y_i = 0 \mid x_i) \tag{4}$$

Equation 3 is re-calculated at each iteration of active sampling. t is the iteration number, and n_t is the number of unlabeled instances at the t-th iteration and I_u is the set of indices of the unlabeled points. By monitoring the average expected error at every single iteration, we can estimate when DWUS' performance starts to saturate, i.e., $\frac{\partial \hat{\epsilon}(DWUS)}{\partial x_t} \leq \delta$. δ is assigned a fixed small value in our evaluations (see [Donmez et al., 2007] for details). When it is near zero, this is equivalent to detecting when a method is stuck in local minima/plateau in gradient descent methods. In fact, this principle is flexible enough to work with any two active learning methods where one is superior for labeling the initial data and the other is favorable later in the process. It generalizes to N sampling methods by introducing additional estimated

[2] Interested reader is referred to [Nguyen and Smeulders, 2004] for further details on the DWUS algorithm.

switchover points based on estimated derivative of expected error for each additional sampling strategy.

After we estimate the cross-over, we use a mixture model for active sampling:

$$x_s^* = \arg\max_{i \in I_u} \pi_1 * E[(\hat{y}_i - y_i)^2 \mid x_i] + (1 - \pi_1) * p(x_i) \tag{5}$$

It is desirable for the above model to minimize the expected future error. After labeling x_{US}^*, the expected error is:

$$f_{US} = \frac{1}{n} \sum_j E_{L+\{x_{US}^*, y\}}[(\hat{y}_j - y_j)^2 \mid x_j] \tag{6}$$

The subscript $L + \{x_{US}^*, y\}$ indicates that the expectation is calculated from the model trained on the data $L + \{x_{US}^*, y\}$. The appropriate weight in this scenario is inversely related with the expected error of uncertainty sampling. Thus, we can replace the weights by $\pi_1 = 1 - f_{US}$, and $1 - \pi_1 = f_{US}$, and obtain the following model:

$$x_s^* = \arg\max_{i \in I_u} (1 - f_{US}) * E[(\hat{y}_i - y_i)^2 \mid x_i] + f_{US} * p(x_i) \tag{7}$$

Achieving the minimum expected loss is guaranteed only for the extreme case where the expected error, f_{US}, of uncertainty sampling is equal to 0. However, correlating the weight of uncertainty sampling with its generalization performance increases the odds of selecting a better candidate after the cross-over.

In the real world, we do not know the true value of f_{US} so we need to approximate it. But, we approximate $\hat{\epsilon}(US)$ as the average expected error of uncertainty sampling

Algorithm 1. The DUAL Algorithm

Input: Labeled data L, Unlabeled data U, max number of iterations T, and δ.
Output: A set S of actively sampled data points.
Program
Initialize: $t = 0$ and $S = \{\}$.
while while(not switching point) **do**
  Run DWUS algorithm and compute $\frac{\partial \hat{\epsilon}(DWUS)}{\partial x_t}$.
  if $\frac{\partial \hat{\epsilon}(DWUS)}{\partial x_t} > \delta$ **then**
    $x_s^* = \arg\max_{i \in I_u} E[(\hat{y}_i - y_i)^2 \mid x_i] p(x_i)$
    Add the chosen point to set S: $S = S \cup x_s^*$
    t=t+1 (Increment counter t)
  else
    Hit the switching point.
  end if
end while
while while($t < T$) **do**
  Compute $E[(\hat{y} - y)^2 | x]$, $p(x)$ via DWUS, and $\hat{\epsilon_t}(US)$ via uncertainty sampling.
  $x_s^* = \arg\max_{i \in I_u} (1 - \hat{\epsilon_t}(US)) * E[(\hat{y}_i - y_i)^2 \mid x_i] + \hat{\epsilon_t}(US) * p(x_i)$
  Add the chosen point to set S: $S = S \cup x_s^*$
  t=t+1
end while

on the unlabeled portion of the data. This leads us to the following selection criterion for DUAL:

$$x_s^* = \arg\max_{i \in I_u} (1 - \hat{\epsilon}(US)) * E[(\hat{y}_i - y_i)^2 \mid x_i] + \hat{\epsilon}(US) * p(x_i) \tag{8}$$

$\hat{\epsilon}(US)$ is updated at every iteration t after the cross-over. Its calculation is exactly the same as in Equation 3. However, the data to sample from is restricted to the set of already labeled examples by active selection. This set is dynamically updated at each iteration with the actively sampled points. Consequently, in order to calculate the expected error of uncertainty sampling the algorithm never requests the label of a point that has not already been sampled during the active learning process. Such a restriction will prevent an exact estimate of the expected error. But, it is a reasonable alternative, and introduces no additional cost of labeling. The pseudo-code for the DUAL algorithm is given in Algorithm 1.

Experiments. To evaluate the performance of DUAL, we ran experiments on UCI benchmarks: splice, diabetes, image and letter. For the letter recognition dataset, we

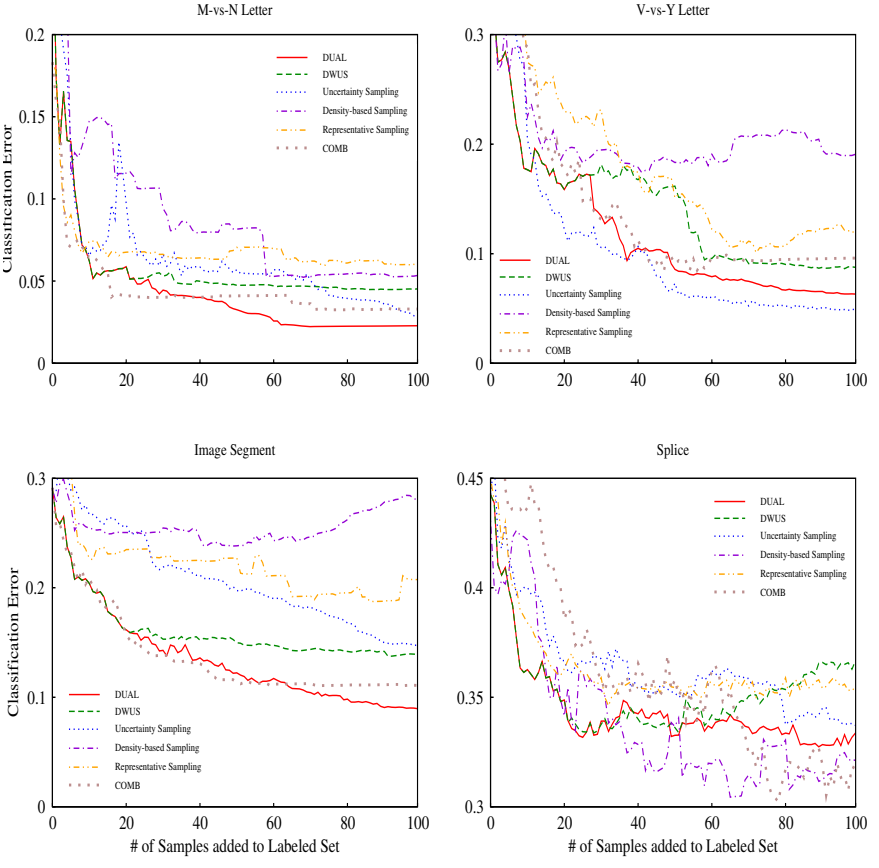

Fig. 2. Results on 4 different UCI benchmark datasets

picked three pairs of letters (M-vs-N,O-vs-D,V-vs-Y) that are most likely to be confused with each other. We compared the performance of DUAL with that of DWUS, uncertainty sampling, representative sampling [Xu et al., 2003], density-based sampling and the COMB method of [Baram et al., 2003]. Density-based sampling adopts the same probabilistic framework as DWUS but uses only the density information for active data selection: $x_s^* = \arg\max_{i \in I_u} p(x_i)$. COMB uses an ensemble of uncertainty sampling, sampling method of [Roy and McCallum, 2001], and a distance-based strategy choosing the unlabeled instance that is farthest from the current labeled set. COMB uses SVM with Gaussian kernel as the base classifier. For further implementation details on COMB, see [Baram et al., 2003].

Figure 2 presents the improvement in error reduction using DUAL over the other methods. We only display results on 4 datasets (See `www.cs.cmu.edu/~pinard/DUAL` for more thorough comparison). DUAL outperforms DWUS and representative sampling both with $p < 0.0001$ significance. DUAL outperforms COMB with $p < 0.0001$ significance on 4 out of 6 datasets, and with $p < 0.05$ on Image and M-vs-N data sets. We also calculate the error reduction of DUAL compared to the strong baseline DWUS. For instance, at the point in each graph after 3/4 of the sampling iterations after cross-over occurs, we observe 40% relative error reduction on O-vs-D data, 30% on Image, 50% on M-vs-N, 27% on V-vs-Y, 10% on Splice, and 6% on Diabetes dataset. These results are significant both statistically and also with respect to the magnitude reduction in relative residual error. DUAL is superior to Uncertainty sampling ($p < 0.001$) on 5 out of 6 datasets.

1.3.2 Density-Sensitive Paired Sampling

Another active learning algorithm we developed is called density-sensitive paired sampling [Donmez and Carbonell, 2008]. It relies on maximizing the likelihood of straddling the decision boundary with paired samples that potentially have the maximum information for the current learner. In order to sample points that are likely to be maximally informative to an active learner, we first seek to maximize the chance that we will sample on both sides of a decision boundary – sampling disproportionately on either side will not optimize boundary placement in the learning process. Maximizing the distance between two points is a step in the right direction, but Euclidean distance may not be the optimal measure; instead we investigate density-sensitive distance functions. The function we adopted in [Donmez and Carbonell, 2008] was proposed by [Chapelle and Zien, 2005] for clustering. The idea is to transform the data into a new space such that points in separate clusters are assigned low similarities, and vice versa. Then, the density-sensitive distance between any two points is approximated by the longest distance edge along each path that connects these two points. For details, see [Chapelle and Zien, 2005].

In order to maximize the likelihood of straddling the decision boundary, and to halve the computational time, we sample a pair of points to label at a time, in contrast to the typical active learning methods that select one point at each iteration. Figure 3 illustrates the motivation for paired sampling in active learning. Here we assume for simplicity the data is linearly separable. The dashed line shows the current decision boundary while the two solid lines define the region where the true boundary is expected to lie; namely the version space. The left figure in Figure 3 is an example of sampling a pair

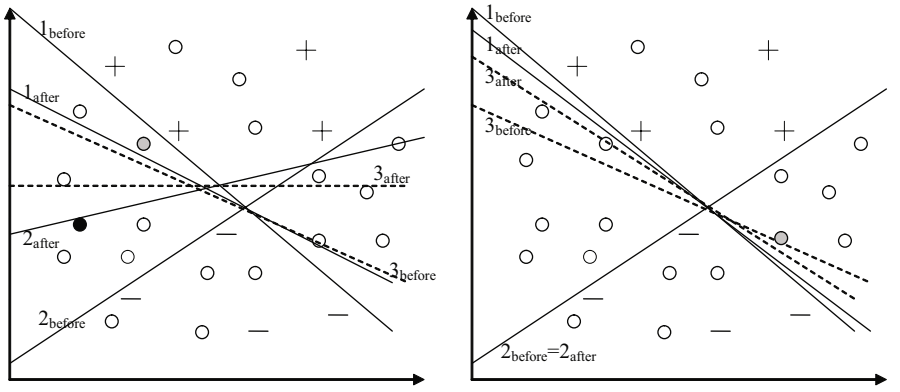

Fig. 3. Illustrative Example: The plus (minus) sign and circles indicate the positively (negatively) labeled points and unlabeled data, respectively. x_{after} and x_{before} indicate the line before and after data is sampled for labeling. The selected points are labeled either positive (shown in grey) or negative (shown in black). This example illustrates our motivation to sample two points with opposite labels at a time instead of a single point.

for labeling from opposite sides of the current boundary. It greatly reduces the version space since both points affect how the version space will be bounded. The figure on the right shows that only a single point is sampled for labeling. Clearly, the left figure has bigger potential to converge to the true hypothesis. These two scenarios illustrate why it is more advantageous to straddle the decision boundary in order to reduce the set of candidate hypotheses rapidly. With this goal in mind, we strive to sample two points with opposite class labels in a binary classification model. In multi-class scenarios, this is equivalent to sampling as many points as the number of classes at each iteration of active learning, seeking to maximize the chance of sampling each class once per round. Since the labels of the unlabeled data are unknown, we need to approximate the likelihood that any two points have opposite class labels, $P(y_i \neq y_j \mid x_i, x_j)$, for all $i, j \in I_u$ where I_u is the set of indices of the unlabeled points in the data. By the cluster assumption [Chapelle and Zien, 2005], points in different clusters are likely to have different labels. In the new representation of the data, points in different clusters are assigned low similarity. It is then reasonable to define $P(y_i \neq y_j \mid x_i, x_j)$ as proportional to the distance between x_i and x_j, i.e. $P(y_i \neq y_j \mid x_i, x_j) \propto \|x_i - x_j\|^2$. For an empirical analysis justifying this claim, see [Donmez and Carbonell, 2008].

In order to obtain a faster learning rate we need to select two points that are likely to have opposite labels *and* high uncertainty:

$$\begin{aligned} \{i^*, j^*\} &= \arg \max_{i \neq j \in I_u} P(y_i \neq y_j \mid x_i, x_j) * U(i,j) \\ &= \arg \max_{i \neq j \in I_u} \|x_i - x_j\|^2 * U(i,j) \end{aligned} \tag{9}$$

where $U(i,j)$ is a complex utility score. To compute $U(i,j)$, we focus on sampling the most uncertain points in high density regions. Hence, we propose an uncertainty weighted density measure:

$$\hat{p}(\boldsymbol{x}) = \sum_{k \in N_x} \exp(-\|\boldsymbol{x} - \boldsymbol{x}_k\|^2) * \min_{y_k \in \{-1,+1\}} \{P(y_k \mid \boldsymbol{x_k}, \hat{\boldsymbol{w}})\} \tag{10}$$

where N_x is the close neighborhood of the data point x, i.e. the set of indices of unlabeled data points whose distance to x is less than a small threshold. $\hat{p}(\boldsymbol{x})$ is a function that measures both the local density of a point and also the information content of its neighbors. The local density depends on the number of close neighbors as well as their proximity. Moreover, each neighbor's contribution to the density score is weighed by its uncertainty; hence, the effect of the neighbors with least uncertainty is reduced. Formally, we define the utility $U(i, j)$ of a pair of points as the sum of the density estimate for each point. By the definition of N_x, it includes the point $\boldsymbol{x}$ in consideration. Hence, Equation 10 includes the uncertainty of the point itself, $\min_{y \in \{-1,+1\}}\{P(y \mid \boldsymbol{x}, \boldsymbol{w})\}$, as a summand with weight equals to $\exp(-\|\boldsymbol{x} - \boldsymbol{x}\|^2) = 1$. We propose to give more flexibility to that uncertainty term by introducing a regularization coefficient. It quantifies a trade-off of the information content of an instance with the proximity weighted information content of its neighbors. This allows us to define the utility function as follows:

$$\begin{aligned} U(i,j) &= \log\{\hat{p}(\boldsymbol{x_i}) + \hat{p}(\boldsymbol{x_j})\} \qquad (11)\\ &= \log\Bigg\{ \sum_{k \neq i \in N_{x_i}} \exp(-\|\boldsymbol{x}_i - \boldsymbol{x}_k\|^2) * \min_{y_k \in \{\pm 1\}} \{P(y_k \mid \boldsymbol{x_k}, \hat{\boldsymbol{w}})\} + \\ &\quad \sum_{r \neq j \in N_{x_j}} \exp(-\|\boldsymbol{x}_j - \boldsymbol{x}_r\|^2) * \min_{y_r \in \{\pm 1\}} \{P(y_r \mid \boldsymbol{x_r}, \hat{\boldsymbol{w}})\} \\ &\quad + s * \Big(\min_{y_i \in \{\pm 1\}} \{P(y_i \mid \boldsymbol{x_i}, \hat{\boldsymbol{w}})\} + \min_{y_j \in \{\pm 1\}} \{P(y_j \mid \boldsymbol{x_j}, \hat{\boldsymbol{w}})\} \Big) \Bigg\} \end{aligned}$$

Experiments. We conducted a set of experiments in order to evaluate our method on 5 real datasets and one artificial dataset, details of which can be found in [Donmez and Carbonell, 2008]. We compared our paired sampling approach with four other strategies: most uncertain, density-only, representative sampling [Xu et al., 2003], and random sampling. Figure 4 shows the results on the UCI Breast data comparing five methods. Our method has the steepest decrease as well as the lowest final error rate which is very close to the optimal achieved using the entire training data. A more thorough comparison of all methods on six datasets is given in Table 1. The first column in Table 1 shows the dataset and the corresponding iteration at which the error rates are compared. The percentage error reduction against the random sampling baseline is given in parenthesis. Lowest error rates are given in bold. Our method wins on the majority of the cases. Whenever it loses, there is only a slight difference between our method and the winner so our method is still comparable on cases where it is not the best.

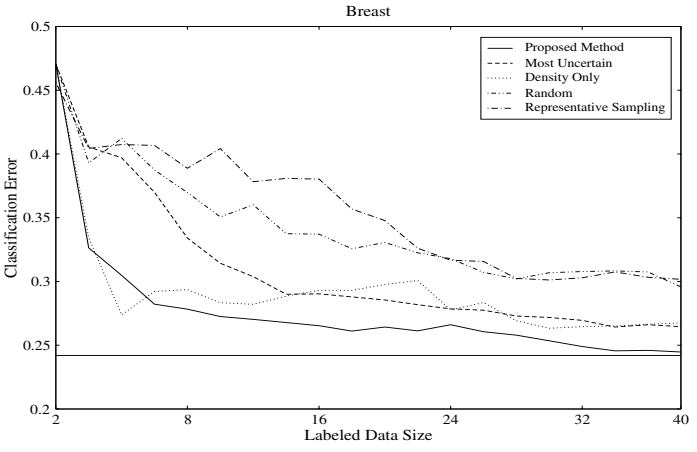

Fig. 4. Results on UCI Breast data. The solid horizontal line indicates the 10-fold cross-validation error using the entire data as the training data.

Table 1. Comparison of five different active learners on all datasets

Data	Proposed Method	Most Uncertain	Density Only	Representative	Random
Breast 5	**0.278 (-24.6%)**	0.334 (-9.04%)	0.293 (-20.5%)	0.380 (+2.9%)	0.369
Breast 11	**0.264 (-20%)**	0.285 (-13.6%)	0.297 (-10%)	0.347 (+5.1%)	0.330
Breast 17	**0.249 (-18.8%)**	0.269 (-12.3%)	0.264 (-14%)	0.302 (-1.6%)	0.307
Heart 5	**0.213 (-18.3%)**	0.245 (-6.1%)	0.220 (-15.7%)	0.216 (-17.2%)	0.261
Heart 11	**0.198 (-4.3%)**	0.208 (+0.4%)	0.220 (+6.2%)	0.205 (-0.9%)	0.207
Heart 17	0.166 (-13.5%)	**0.164 (-14.5%)**	0.219 (+14%)	0.20 (+4.1%)	0.192
Flare 5	0.465 (+5.2%)	0.454 (+2.7%)	0.454 (+2.7%)	0.478 (+8.1%)	**0.442**
Flare 11	**0.394 (-1.6%)**	0.451 (+10%)	0.422 (+2.9%)	0.417 (+1.7%)	0.410
Flare 17	**0.366 (-8.7%)**	0.449 (+11.9%)	0.401 (0%)	0.393 (-1.9%)	0.401
Face 5	0.350 (-1.9%)	0.468 (+31%)	0.420 (+17.6%)	**0.313 (-12.3%)**	0.357
Face 11	**0.210 (-23.3%)**	0.312 (+13.8%)	0.287 (+4.7%)	0.252 (-8%)	0.274
Face 17	**0.151 (-32.5%)**	0.196 (-12.5%)	0.189 (-15.6%)	0.202 (-9.8%)	0.224
Glass2 5	0.339 (-11%)	0.442 (+16%)	0.392 (+2.8%)	**0.326 (-14.4%)**	0.381
Glass2 11	0.317 (-7%)	0.341 (0%)	0.324 (-4.9%)	**0.31 (-9%)**	0.341
Glass2 17	**0.266 (-8.9%)**	0.292 (0%)	0.275 (-5.8%)	0.30 (+2.7%)	0.292
g50c 5	**0.169 (-46.3%)**	0.242 (-23.1%)	0.187 (-40.6%)	0.241 (-23.4%)	0.315
g50c 11	**0.110 (-37.8%)**	0.136 (-23.1%)	0.128 (-27.6%)	0.168 (-5%)	0.177
g50c 17	**0.079 (-34.1%)**	0.094 (-21.6%)	0.102 (-15%)	0.139 (+15.8%)	0.120

2 Proactive Learning

2.1 Why Proactively Learn?

Active learning, for all its popularity, operates in an idealized framework. For instance, we assume that there exists an oracle who provides accurate answers to all questions (error-free labeling), who never gets tired, and who charges uniformly regardless of question difficulty. In virtually all applications these assuptions break down: Human

experts occasionally err; they can be reluctant to answer if tired or faced with difficult questions; different experts could charge differently for their services, perhaps based on quesion difficulty. If we use lab experiments to obtain labels, rather than human experts, these too can be fallible, producing inaccurate results, failing to produce results on occasion, and exhibiting different costs depending on the experimental setup.

Consider, for instance, proteomic classification [Cheng et al., 2005; Wang et al., 2007], where the task is to label proteins as to functional and structural family and subfamily membership based on their primary amino-acid sequence and other features (e.g. locus, density, homology, etc.). Although about three million proteins have been sequenced, only about 1.5% have been structurally resolved with certainty, and these are predominantly soluble proteins which are easier to crystalize and thefore ameanble to X-ray diffraction crystalography – the primary method for obtaining the structure of large proteins. If we need to obtain a structural label for a new protein, we have several options: 1) Ask a human expert, who will venture a educated guess if he or she can, or say "sorry" otherwise; 2) Check if the protein has very high sequence homology to a structurally resolved one, and if so infer an identical structure label, but this process will only provide labels for very sequence-proximate instances at best; or 3) Perform a lab experiment such as X-ray Crystallography – very expensive and uncertain to yield results, or NMR imaging for small proteins – less expensive but its reliability plummets with increasing protein size. The problem we have is not just to choose an instance to label, but also choose a labeling method (an "oracle") most appropriate for the instance. And, if we have a fixed budget, we must also calculate the cost of failure – experiments that do not yield results still incur costs. Hence, we cannot select the information-value best instance without regards to probability of successfully obtaining a label and without regard to cost. Ideally we want to jointly optimize the selection of instance and labeling method (i.e. oracle), which might yield a more certain or inexpensive choice over the information-value-only optimum.

There are many real learning problems with characteristics similar to protein labeling, perhaps more so than for the idealized oracle of active learning. Thus, we need to extend active learning to include a utility-based deision-theoretic approach that encompasses oracle errors, oracle reluctance, and variable costs. We call this extension *proactive learning*. In summary, proactive learning extends active learning as in Table 2.

Table 2. Active versus Proactive Learning

Oracle Property	Active Learning	Proactive Learning
Number	Individual (one oracle)	Plural (different oracles)
Accuracy	Infallible (always right)	Probabilistic (may err)
Reluctance	Indefatigable (always answers)	Reluctant (may refuse to answer)
Cost	Invariant (free or constant)	Variable (per oracle, per instance)

Proactive learning is a naescent field of investigation; the remainder of this paper provides a substantial exploratory foray, but much research remains to be done.

2.2 A Case-Study: Optimizing Proactive Learning with Two Oracles

In this section, we aim to show that a decision-theoretic approach [DeGroot, 2004] is beneficial for cost-effective sampling in cases with imperfect oracles. We mainly focus

on three model scenarios embodying the notion of oracles with differing properties and costs. In each scenario, we target a single oracle property. In "Scenario 1", we assume there exist one reliable oracle and one reluctant oracle. The reliable oracle gives an answer every time it is invoked with a query, and the answer is always correct. The reluctant oracle, on the other hand, does not always provide an answer, but when it answers it does so correctly. We assume the probability of getting an answer from the reluctant oracle depends on the difficulty of the classification task. Not surprisingly, they charge different fees: the reliable oracle is more expensive than the potentially reluctant one.

Rather than fixing the number of instances to classify, as in standard active learning, proactive learning fixes a maximum budget envelope since instances and oracles may have variable costs. We formulate the problem as a joint optimization of which instance(s) to sample and which oracle to use to purchase their label(s), where the objective is to maximize the cumulative value of the information given a pre-defined budget:

$$\text{maximize } E[V(S)] - \lambda(\sum_k t_k * C_k) \quad \text{s.t. } \sum_k t_k * C_k = B \text{ , } \sum_k t_k = |S|$$

where B is the budget, S is the set of instances to be sampled, and $E[V(S)]$ is the expected value of information of the sampled data to the learning algorithm. $V(S)$ is a value function that can be replaced with any active selection criterion. In our experiments, we adopt a density weighted uncertainty score proposed in [Donmez and Carbonell, 2008], which significantly outperforms other strong baselines. $k \in K$ denotes the chosen oracle from the set of oracles, K, and λ is a parameter controlling the relative importance of maximizing the information and minimizing the cost. For simplicity, we assumed $\lambda = 1$ in this work. C_k and t_k indicate the cost of the chosen oracle and the number of times it is invoked, respectively. UL is the set of unlabeled examples, $|S|$ is the total size of the sampled set[3]. The drawback of this approach is that it is difficult to optimize directly due to the fact that the maximization is over the entire set of potential sampling sequences, an exponentially large number. Thus, we cannot decide all the points to be sampled at once. A tractable alternative is a greedy approximation that will select the locally-optimal strategy at each round according to a joint utility function of the oracle-instance pair:

$$(x^*, k^*) = \arg \max_{x \in UL, k \in K} U(x,k) = \arg \max_{x \in UL, k \in K} \frac{P(ans \mid x,k) * V(x)}{C_k} \text{ where } k \in K \tag{12}$$

where the utility $U(x,k)$ is defined as the expected information value of that example at unit cost.

We need to simulate reliability and reluctance since there do not exist real-world datasets that have ground truth information on it. We assume the reliable (perfect) oracle resembles a system that has been trained on the entire dataset and has zero training error. Unlike the reliable oracle, a reluctant oracle has access only to a small portion of the data; therefore, it is not knowledgable for every point. We train a classifier on a small random subset of the entire data to obtain a posterior class distribution $P(y \mid x)$. We

[3] The extension of this formulation to more than two oracles is straightforward.

assume that the chance of obtaining an answer from the reluctant oracle is low when the uncertainty $\min_{y \in \mathcal{Y}} P(y \mid x)$ is high and vice versa.

We further assume that the reluctant oracle's knowledge level or response characteristics are unknown apriori. Hence, we estimate these properties in an exploration phase. We hypothesize that the oracle shows similar response characteristics for similar data instances since we model reluctance as a function of uncertainty which is similar for similar instances. Therefore, it is reasonable to estimate the answer probability of the reluctant oracle by inquiring the labels of the cluster centroids and propagating the information to nearby points. We cluster the unlabeled data using k-means clustering [Hartigan and Wong,]. The number of clusters depends on a pre-defined budget available for this phase and the cost of the reluctant oracle. For each cluster, if we obtain the label of the centroid, then we increase the answer probability of the points in this cluster. Similarly, we decrease the answer probability of the points in the clusters whose centroids we did not obtain the labels of. Initially, we assume the answer probability for each unlabeled point is 0.5, which indicates a random guess. Then, we update the answer probabilities of the reluctant (unreliable) oracle so that it changes as a function of the proximity of the point to the cluster centroid and oracle responsiveness:

$$\hat{P}(ans \mid x, reluctant) = \frac{0.5}{Z} * exp\left(\frac{h(x_{c_t}, y_{c_t})}{2} \ln \frac{max_d - \|x_{c_t} - x\|}{\|x_{c_t} - x\|} \right), \forall x \in C_t \tag{13}$$

where Z is a normalization factor. x_{c_t} is the centroid of the cluster C_t that includes x. $h(x_c, y_c) \in \{1, -1\}$ is an indicator function which is equal to 1 when we receive the label y_c for the centroid x_c, and -1 otherwise. $\|x_c - x\|$ is the Euclidean distance between the cluster centroid x_c and the point x and $max_d := \max_{x_{\acute{c}}, x} \|x_{\acute{c}} - x\|$ is the maximum distance between any cluster centroid and data point.

We substitute the estimated answer probability into the utility function, i.e. $\hat{U}(x,k) = \frac{\hat{P}(ans|x,k) * V(x)}{C_k}$. The joint sampling of the oracle-example pair can now be performed as shown in Algorithm 2. The algorithm works in rounds till the budget is exhausted. Each round corresponds to a single label acquisiton attempt where sampling persists until obtaining a label. However, we need to restrain from spending too much on a single attempt by adaptively penalizing the reluctant oracle every time it refuses to answer. At any given round, if the algorithm chooses the reluctant oracle and does not receive an answer, the utility of remaining examples with respect to this oracle decreases by the amount spent thus far at this round:

$$\hat{U}(x, reluctant) = \frac{\hat{P}(ans \mid x, reluctant) * V(x)}{C_{round}}$$

where C_{round} is the amount spent thus far in the given round. This penalization only applies to the reluctant oracle since the reliable oracle always provides the label. Algorithm 2 selects the maximum utility examples. This framework leads to an incrementally optimal solution in the sense that the most useful data is sampled at the minimum cost.

Algorithm 2. Proactive Learning: Scenario 1

Input: a classifier f, labeled data L, unlabeled data UL, entire budget B, clustering budget $B_C < B$, two oracles, each with a cost C_k, $k \in K = \{reliable, reluctant\}$
Output: f
- Cluster UL into $p = B_C / C_{reluctant}$ clusters
- Let x_{c_t} be the data point closest to its cluster centroid, $\forall t = 1, ..., p$
- Query the label y_{c_t} for each cluster centroid x_{c_t}
- Identify $\{x_{c_1}, ..., x_{c_g}\}$ for which we obtain the labels[4]
- Estimate $\hat{P}(ans \mid x, reluctant)$ via Equation 13
- Update $L = L \cup \{x_{c_t}, y_{c_t}\}_{t=1}^{g}$, $UL = UL \setminus \{x_{c_t}, y_{c_t}\}_{t=1}^{g}$
- cost spent so far $C_T = B_C$
while $C_T < B$ **do**
  - Train f on L
  - Initialize the cost of this round $C_{round} = 0$ and the set of queried examples $Q = \{\}$
  - $\forall k \in K, x \in UL$ estimate utility $\hat{U}(x, k)$
  repeat
    1. Choose $k^* = \arg \max_{k \in K} \max_{x \in UL \setminus Q} \{\hat{U}(x, k)\}$
    2. Choose $x^* = \arg \max_{x \in UL \setminus Q} \{\hat{U}(x, k^*)\}$
    3. Update $C_{round} = C_{round} + C_{k^*}$
    4. $Q = Q \cup \{x^*\}$
    5. Query the label y^* with probability $P(ans \mid x^*, k^*)$
  until label y^* is obtained
  - Update $C_T = C_T + C_{round}$
  - Update $L = L \cup (x^*, y^*)$ and $UL = UL \setminus (x^*, y^*)$
end while

In real applications, such as protein structure classification, there might also be fallible oracles which always answer, but the credibility of the answer is questionable[5]. We simulate this setting in "Scenario 2", where we assume two oracles; one reliable and one unreliable oracle. The reliable oracle is the perfect oracle that always provides the correct answer to any query. The unreliable oracle in this scenario is fallible that it may provide a random label for the queried example x with probability $1 - P(correct \mid x, k)$. We assume the probability of correct classification approaches 0.5 (random guess) if an example approaches the decision boundary. Similar to the previous scenario, we request the labels of the cluster centroids. Unlike the reluctant oracle, the fallible oracle provides the label together with its confidence[6]. The confidence is its posterior class probability for the provided label, $P(y \mid x)$. If the class posterior is within an uncertainty range, then we decide not to use the provided label since it is likely to be noisy. We decrease the correctness probability for the points in the cluster whose centroid has a class posterior in the uncertainty range and vice versa;

[5] Wrong answers might also be due to the noise in the data. But, we for now we assume the data is noise-free, and fallibility is only a property of the oracle.

[6] This could be a rather strong assumption. Section 2.3 discusses a more general case without requiring such information.

i.e. $\hat{P}(correct \mid x, fallible) = \frac{0.5}{Z} * exp\left(\frac{\tilde{h}(x_{c_t}, y_{c_t})}{2} \ln \frac{max_d - \|x_{c_t} - x\|}{\|x_{c_t} - x\|}\right) \forall x \in C_t$ where $\tilde{h}(x_{c_t}, y_{c_t}) = -1$ if $P(y \mid x_{c_t})$ is in the uncertainty range, and 1 otherwise. The pseudocode of the algorithm is given in Algorithm 3, where h denotes the number of high confident centroids.

Algorithm 3. Proactive Learning: Scenario 2

Input: a classifier f, labeled data L, unlabeled data UL, entire budget B, clustering budget $B_C < B$, two oracles, each with a cost C_k, $k \in K = \{reliable, fallible\}$
Output: f
- Cluster UL into $p = B_C / C_{fallible}$ clusters
- Let x_{c_t} be the data point closest to its cluster centroid, $\forall t = 1, ..., p$
- Query the label y_{c_t} for each cluster centroid x_{c_t}
- Identify $\{x_{c_1}, ..., x_{c_h}\}$ for which the fallible oracle has high confidence
- Estimate $\hat{P}(correct \mid x, fallible)$ as in Equation 13
- Update $L = L \cup \{x_{c_t}, y_{c_t}\}_{t=1}^{h}$, $UL = UL \setminus \{x_{c_t}, y_{c_t}\}_{t=1}^{h}$
- cost spent so far $C_T = B_C$
while $C_T < B$ **do**
1. Train f on L
2. $\forall k \in K, x \in UL$ estimate $\hat{U}(x, k)$
3. Choose $k^* = \arg \max_{k \in K} \max_{x \in UL} \{\hat{U}(x, k)\}$
4. Choose $x^* = \arg \max_{x \in UL} \{\hat{U}(x, k^*)\}$
5. Update $C_T = C_T + C_{k^*}$
6. Update $L = L \cup (x^*, y^*)$ and $UL = UL \setminus (x^*, y^*)$ where y^* is the correct label with probability $P(correct \mid x^*, k^*)$
end while

Thus far, we have only considered the settings where a uniform fee is charged for every query by an oracle, although each oracle may charge differently. Fraud detection in banking transactions is a good example for this setting. The customer records are saved in the bank database so it takes the same amount of time and effort, hence the same cost, to look up any entry in the database. However, it is possible that the costs are distributed non-uniformly over the set of instances. For instance in text categorization, it might be relatively easy for an annotator to categorize a short web page; hence the cost is modest. On the other hand, it is more time-consuming to assign a book into a category, which incurs a considerable reading time and therefore cost. We already discussed variable costs in protein classification, where less expensive NMR imaging may apply to smaller proteins. Another example for a non-uniform cost scenario is medical diagnosis. Some diseases such as herpes are easy to diagnose. Such diagnoses are not costly since there is usually a major definitive symptom, i.e. outbreak of blisters on the skin. On the other hand, diagnosing hepatitis can be more costly since it may require blood and urine tests, CT scans, or even a liver biopsy. In "Scenario 3", we explore the problem of deciding which instances to query for the labels when label acquisition cost varies with the instance. We assume two oracles one of which has a uniform and fixed cost for each query whereas the other charges according to the task

Algorithm 4. Proactive Learning: Scenario 3

Input: a classifier f, labeled data L, unlabeled data UL, entire budget B, two oracles, each with a cost C_k, $k \in K = \{unif, non-unif\}$
Output: f
cost spent so far $C_T = 0$
while $C_T < B$ **do**
1. Train f on L
2. $\forall k \in K, x \in UL$ calculate $\hat{U}(x,k)$ via Equation 14.
3. Choose $k^* = \arg\max_{k \in K} \max_{x \in UL} \{\hat{U}(x,k)\}$
4. Choose $x^* = \arg\max_{x \in UL} \{\hat{U}(x,k^*)\}$
5. Update $C_T = C_T + C_{k^*}$
6. Update $L = L \cup (x^*, y^*)$, $UL = UL \setminus (x^*, y^*)$

end while

difficulty. We further assume that these oracles always provide an answer and both are perfectly reliable in their answers.

We model the cost of each example x as a function of the posterior class distribution $P(y \mid x)$ so that there is a positive correlation between the classification difficulty and the cost. We use the class posterior calculated similarly in the previous scenarios. The non-uniform cost $C_{non-unif}(x)$ per instance is then defined as follows:

$$C_{non-unif}(x) = 1 - \frac{\max_{y \in \mathcal{Y}} P(y \mid x) - 1/|\mathcal{Y}|}{1 - 1/|\mathcal{Y}|}$$

The cost increases as the instance approaches the decision boundary and vice versa. In other words, the oracle charges based on how difficult it is to classify, and how valuable it might potentially be to the learner. This sets up a challenging decision in terms of the utility-cost trade-off. The utility score in this scenario is calculated as the difference between the information value and the cost instead of the information value per unit cost[7]. This is to avoid infinitely large utility scores as a result of the division by small ϵ-cost. Thus, the revised utility score per oracle is given as follows:

$$\begin{aligned} \hat{U}(x, unif) &= V(x) - C_{unif} \\ \hat{U}(x, non-unif) &= V(x) - C_{non-unif}(x) \end{aligned} \tag{14}$$

where C_{unif} is the fixed cost of the uniform-cost oracle. The pseudocode of the algorithm is given in Algorithm 4. There is no clustering phase in Algorithm 4 since we assume we know the cost of every instance, which is realistic for many real-world applications.

Problem Setup. In order to simulate the reliability of the labeling source (oracle), we assume that a perfectly reliable oracle resembles by a strong classifier trained on the

[7] In general, if the cost and information value are not assessed in the same units, then they are normalized into the same range.

entire data, and having zero training error. An unreliable oracle, on the other hand, resembles a classifier trained on only a small subset of the entire data; hence a weak one. We randomly sampled a small subset from each dataset and trained a logistic regression classifier on this sample to output a posterior class distribution. Then, we identified the instances whose class posterior falls into the uncertainty range, i.e. $\min_y P(y \mid x) \in [0.45, 0.5]$. This range is used to filter the instances that the reluctant oracle does not answer or the fallible oracle outputs a random label. We varied the percentage of instances that fall into the uncertainty range $[.45, .5]$. The second column in Table 3 shows the different percentages used in our experiments. There is also a tradeoff between cost and unreliability to have an incentive to choose between oracles rather than exploiting a single one. The higher cost ratios for the fallibility scenario indicate more penalization for receiving a noisy label. See Table 3 for details.

Table 3. Oracle Properties and Costs. B_C is the clustering budget, B is the entire budget. Uncertain % is the percentage of the uncertain data points. Cost Ratio is the ratio of the cost of the reliable oracle to the cost of the unreliable one.

Scenario	Uncertain %	Cost Ratio	B_C	B
Scenario 1	45-50%	1:3	20	300
	50-55%	1:4	30	
	65-70%	1:5	50	
Scenario 2	45-50%	1:5	20	300
	50-55%	1:6	30	
	65-70%	1:7	50	

The other case we need to simulate involves the uniform and non-uniform cost oracles. The cost of each instance for the variable-cost oracle is defined as a function of the class posterior obtained on the randomly chosen subset. The cost of the uniform-cost oracle is chosen within the range of instance costs for the variable-cost oracle. We varied the fixed cost such that there is always an incentive to choose between oracles instead of fully exploiting a single one.

All the results are averaged over 10 runs. At each run, we start with one randomly chosen labeled example from each class which are assumed to be known a-prori, and therefore free. The rest of the data is considered unlabeled. The learner selects one example at each iteration to be labeled, and the learning function is tested on the remaining unlabeled set once the label is obtained. The learner pays the cost of each queried example regardless of whether the label is obtained. The budget is fixed at 300 in Scenario 1 and 2, and at 20 in Scenario 3[8]. A small budget is enough for the latter since the cost of individual instances can be very small depending on the posterior probability. The clustering budget, on the other hand, varies according to the unreliability, but is the same for each baseline under the same scenario (See Table 3).

[8] At each run, the total number of iterations to spend the entire budget may differ depending on how the budget is allocated between oracles. We rely on the minimum number of iterations attained over 10 runs for each experiment. This results in reporting different maximum elicitation costs smaller than the budget for different experiments

Evaluation. We conducted a thorough analysis to examine the performance of our method under various conditions. Due to the lack of existing work on cost-sensitive active learning with multiple oracles, we compared our method against active sampling with randomly chosen oracles and active sampling with a single oracle. We denote our method by *Joint*, the random sampling of oracles by *Random*. *Reliable*, *Reluctant*, and *Fallible* refer to the corresponding single oracle baseline.

We evaluated the performance of the proposed methods on various real-world benchmark datasets, but here we only report a representative subset of results. Figure 5 shows the results for the reluctance scenario on the Spambase dataset. Each plot indicates a different cost ratio. Our method outperforms the others on every case while the performance gap increases with the cost ratio. This is largely because the oracle differences leave more room for improvement via oracle selection in the latter case. When the unreliability gets higher, the reluctant oracle tend to spend almost the entire budget on a single label acquisition attempt. This leads to acquiring only a small amount of labeled data; hence, poor performance. As a result, we do not report the reluctant oracle baseline except for the $1:3$ cost ratio.

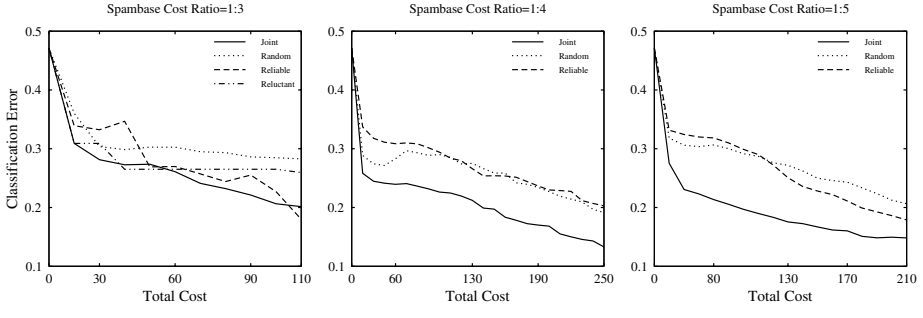

Fig. 5. Performance Comparison for Scenario 1 (Reluctance) on the Spambase dataset. The cost ratio is indicated above each plot.

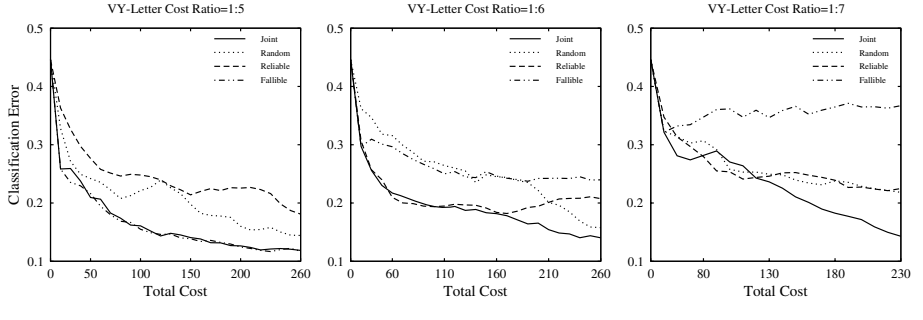

Fig. 6. Performance Comparison for Scenario 2 (Fallible) on the VY-Letter dataset. The cost ratio is indicated above each plot.

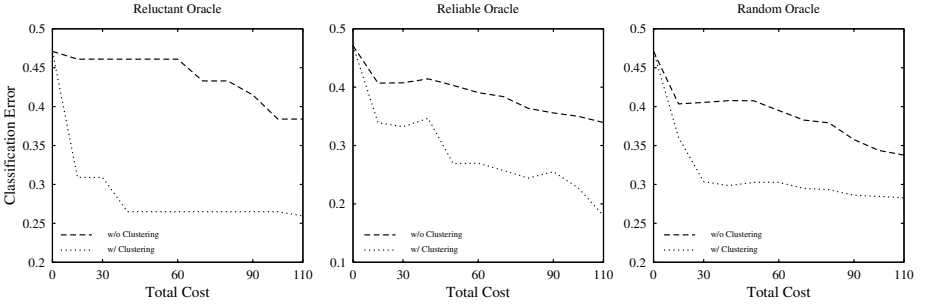

Fig. 7. Change in performance of each baseline with and without clustering on Spambase. The type of baseline is given in the title. The cost ratio is 1:3.

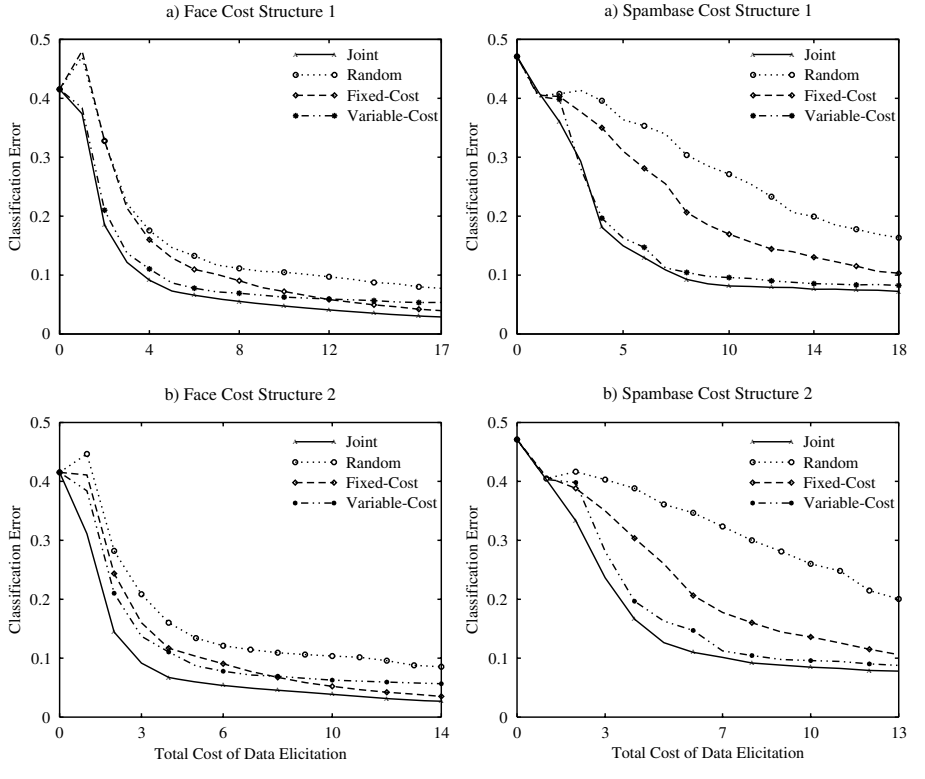

Fig. 8. Comparison of different algorithms under non-uniform cost structures (Scenario 3) on Face (left panel) and Spambase (right panel) datasets. a) Cost Structure 1: Fixed-Cost oracle has Cost1 b) Cost Structure 2: Fixed-Cost oracle has Cost2.

Figure 6 compares the performances for Scenario 2 on the VY-Letter dataset. The fallible oracle in this scenario performs poorly as unreliability increases since the labels are increasingly likely to be noisy. This pattern is especially evident in Figure 6 for the cost ratio $1:7$. On the other hand, *Joint* strategy is quite effective for reducing the error

in this scenario, indicating that it is capable of reducing the risk of introducing noisy data through strategic selection between oracles.

Figure 8 presents the evaluation results when the cost varies non-uniformly across the set of instances. We experimented with different assignments of the fixed cost, but we present two representative assignments for each dataset. The remaining assignments are not included since they are similar to those reported here. *Joint* is the best performer throughout the full operating range. The performance difference between *Joint* and each baseline is also statistically significant based on a paired two-sided t-test ($p < 0.01$).

In order to investigate if the initial clustering phase helps all the baselines, we re-ran each baseline excluding the clustering step. (See Figure 7). Every baseline significantly benefits from clustering, with the biggest boost in improvement occuring for the reluctant oracle. Hence, both the baselines and the ''Joint'' strategy benefit from the diversity-based sampling via clustering in their initial steps. Without pre-clustering, the reluctant oracle is prone to spend too much on a single elicitation attempt due to unsuccessful labeling requests. It can, however, maximize the chance of receiving a label through diversity sampling during the clustering step instead of getting stuck in one round for a single label.

2.3 Future Directions

Thus far, we explored simple cases each of which addresses a single oracle property with one perfect and one imperfect oracle. A natural next step is to extend it to multiple oracles with differing levels of reliability. With multiple oracles, the goal is to select the oracle-example pair with the maximum utility, but the estimation of the oracle properties becomes more challenging. For instance, oracles may be fallible at differing levels. A simple scenario is to assume that each oracle has a fixed probability of making a labeling mistake independent of the data instance to be classified. The goal, then, is to estimate each oracle's noise probability and use the estimates to select the least noisy oracle(s) for label acquisition. As before, noise estimation requires an exploration phase whereas querying the least noisy oracles is the exploitation phase. Hence, there is an exploration vs. exploitation tradeoff. This problem is analogous in many ways to multi-armed bandit problem.

The multi-armed bandit problem for a gambler is to decide the arm to pull in a K-slot machine to maximize her total reward in a series of trials. Each action does not just bring in reward, but also helps discover new knowledge about the slot machine that can be used to assist future actions. The reward of each lever is drawn from a specific distribution which is unknown to the gambler. Thus, she tries to acquire knowledge through repeated trials via balancing the exploration vs. exploitation tradeoff. There is quite a lot of work done in the literature to solve the multi-armed bandit problem. One popular bandit algorithm is the exponential weight algorithm introduced by [Auer et al., 2002]. The probability of choosing the lever k at round t is defined by:

$$p_k(t) = (1-\gamma)\frac{w_k(t)}{\sum_K^{j=1} w_j(t)} + \frac{\gamma}{K}$$

where $w_j(t+1) = w_j(t)exp(\gamma\frac{r_j(t)}{p_j(t)K})$ if the lever j has been pulled at time t with the observed reward $r_j(t)$, otherwise $w_j(t+1) = w_j(t)$. It has a theoretical guarantee that

the regret of the algorithm is $O(\sqrt{KTlog(K)})$ with γ decreasing over time, where regret is defined the difference between the reward sum of the optimal strategy and the sum of the collected rewards. This strategy can be adopted to the proactive learning setting with multiple fallible oracles with the appropriate definitions of the reward function. In a completely unsupervised setting (absence of gold standard label information), the reward for an oracle output can be determined based on the majority vote from all and/or a subset of oracles. The majority voting principle has been used by [Sheng et al., 2008] in a simple context where each oracle has the same noise level; hence, there is no need for estimation.

More complexity arises when the error probability of each oracle is dependent on the data instance. This is probably a more realistic assumption than independence on the data. One possible solution is to leverage the fact that similar instances have similar labels. In other words, if an oracle makes a misprediction for some instance then it is likely to make mispredictions on similar instances. Then, it becomes important to query the oracles for instances that are most representative of the entire input space. A similar approach to out belief propagation framework can be quite effective to estimate the oracles' error probabilities without intensive labeling.

Another interesting problem occurs when each oracle has partial knowledge at only a certain region of the data distribution and its cost depends on its knowledge. The goal is to select the oracle who is an expert at the region of interest. Therefore, each oracle needs an estimate about the region at which it is knowledgeable. One possible way to attack this problem is to adopt a top-down strategy where each oracle is assumed to be an expert for the entire data space. The knon expertise region of each oracle shrinks over time based on inaccurate responses to labeling queries. How to calculate the shrinkage without massively repeated querrying remains an open question.

The journey to proactive machine learning is a long and exciting one. We have explored certain roads and not others. For instance, one could simplify the problem by assuming that oracle properties are fully known a-priori, or one could take detours into investigations such as referal networks among oracles, if the one queried does not know the answer with sufficient accuracy – but that would require a more complex cost model with referral fees and mode complex oracle network learning models.

References

[Auer et al., 2002] Auer, P., Cesa-Bianchi, N., Freund, Y., Schapire, R.E.: The nonstochastic multiarmed bandit problem. SIAM Journal on Computing 32(1), 48–77 (2002)

[Baram et al., 2003] Baram, Y., El-Yaniv, R., Luz, K.: Online choice of active learning algorithms. In: ICML 2003, pp. 19–26 (2003)

[Blum and Mitchell, 1998] Blum, A., Mitchell, T.: Combining labeled and unlabeled data with co-training. In: COLT 1998, pp. 92–100 (1998)

[Chapelle and Zien, 2005] Chapelle, O., Zien, A.: Semi-supervised classification by low density separation. In: Proceedings of the Tenth International Workshop on Artificial Intelligence and Statistics 2005 (2005)

[Cheng et al., 2005] Cheng, B.Y., Carbonell, J.G., Klein-Seetharaman, J.: Protein classification based on text document classification techniques. Proteins: Structure, Function and Bioinformatics 58(4), 955–970 (2005)

[Cohn et al., 1995] Cohn, D.A., Ghahramani, Z., Jordan, M.I.: Active learning with statistical models. In: Advances in Neural Information Processing Systems, vol. 7, pp. 705–712 (1995)

[DeGroot, 2004] DeGroot, M.H.: Optimal Statistical Decisions. Wiley Classics Library, Chichester (2004)

[Donmez and Carbonell, 2008] Donmez, P., Carbonell, J.G.: Paired sampling in density-sensitive active learning. In: International Symposium on Artificial Intelligence and Mathematics (2008)

[Donmez et al., 2007] Donmez, P., Carbonell, J.G., Bennett, P.N.: Dual strategy active learning. In: Kok, J.N., Koronacki, J., Lopez de Mantaras, R., Matwin, S., Mladenič, D., Skowron, A. (eds.) ECML 2007. LNCS (LNAI), vol. 4701, pp. 116–127. Springer, Heidelberg (2007)

[Hartigan and Wong,] Hartigan, J.A., Wong, M.A.: A k-means clustering algorithm. Applied Statistics, 28

[He et al., 2007] He, J., Carbonell, J., Liu, Y.: Graph-based semi-supervised learning as a generative model. In: IJCAI 2007 (2007)

[Joachims, 1999] Joachims, T.: Transductive inference for text classification using support vector machines. In: ICML 1999 (1999)

[Lewis and Gale, 1994] Lewis, D., Gale, W.: A sequential algorithm for training text classifiers. In: SIGIR 1994, pp. 3–12 (1994)

[McCallum and Nigam, 1998] McCallum, A., Nigam, K.: Employing em and pool-based active learning for text classification. In: ICML 1998, pp. 359–367 (1998)

[Melville and Mooney, 2003] Melville, P., Mooney, R.J.: Constructing diverse classifier ensembles using artificial training examples. In: IJCAI 2003, pp. 505–510 (2003)

[Michalski et al., 1983] Michalski, R.S., Carbonell, J.G., Mitchell, T.M.(eds.): Machine Learning: An Artificial Intelligence Approach. TIOGA Publishing Co. (1983)

[Michalski et al., 1986] Michalski, R.S., Carbonell, J.G., Mitchell, T.M. (eds.): Machine Learning: An Artificial Intelligence Approach, vol. II. Morgan Kaufmann Publishers, Inc., San Francisco (1986)

[Miller et al., 1993] Miller, J.W., Goodman, R., Smyth, P.: On loss functions which minimize to conditional expected values and posterior probabilities. IEEE Transactions on Information Theory 39(4), 1404–1408 (1993)

[Muslea et al., 2000] Muslea, I., Minton, S., Knoblock, C.: Selective sampling with naive co-testing: preliminary results. In: The ECAI 2000 workshop on Machine Learning for information extraction (2000)

[Nguyen and Smeulders, 2004] Nguyen, H.T., Smeulders, A.: Active learning with pre-clustering. In: ICML 2004, pp. 623–630 (2004)

[Reinke and Michalski, 1985] Reinke, R., Michalski, R.S.: Incremental learning of decision rules: A method and experimental results. Presented at the Machine Intelligence Workshop II (1985)

[Roy and McCallum, 2001] Roy, N., McCallum, A.: Toward optimal active learning through sampling estimation of error reduction. In: ICML 2001, pp. 441–448 (2001)

[Shen and Zhai, 2005] Shen, X., Zhai, C.: Active feedback in ad hoc information retrieval. In: SIGIR 2005, pp. 59–66 (2005)

[Sheng et al., 2008] Sheng, V.S., Provost, F., Ipeirotis, P.G.: Get another label? improving data quality and data mining using multiple, noisy labelers. In: KDD 2008, pp. 614–622 (2008)

[Tang et al., 2002] Tang, M., Luo, X., Roukos, S.: Active learning for statistical natural language parsing. In: ACL 2002 (2002)

[Tong and Koller, 2000] Tong, S., Koller, D.: Support vector machine active learning with applications to text classification. In: Proceedings of International Conference on Machine Learning, pp. 999–1006 (2000)

[Wang et al., 2007] Wang, Y., Wu, L.-Y., Chen, L., Zhang, X.-S.: Supervised classification of protein structures based on convex hull representation. International Journal of Bioinformatics Research and Applications 3(2), 123–144 (2007)

[Xu et al., 2003] Xu, Z., Yu, K., Tresp, V., Xu, X., Wang, J.: Representative sampling for text classification using support vector machines. In: Sebastiani, F. (ed.) ECIR 2003. LNCS, vol. 2633, pp. 393–407. Springer, Heidelberg (2003)

[Zhang and Chen, 2002] Zhang, C., Chen, T.: An active learning framework for content-based information retrieval. IEEE Trans. on Multimedia 4, 260–268 (2002)