A Novel Uncertainty Sampling Algorithm for Cost-sensitive Multiclass Active Learning

Kuan-Hao Huang*, Hsuan-Tien Lin*†
r03922062@csie.ntu.edu.tw, htlin@csie.ntu.edu.tw
*Dept. of Computer Science & Information Engineering, National Taiwan University, Taipei, Taiwan
†Appier Inc., Taipei, Taiwan

***Abstract*—Active learning is a setup that allows the learning algorithm to iteratively and strategically query the labels of some instances for reducing human labeling efforts. One fundamental strategy, called uncertainty sampling, measures the uncertainty of each instance when making querying decisions. Traditional active learning algorithms focus on binary or multiclass classification, but few works have studied active learning for cost-sensitive multiclass classification (CSMCC), which allows charging different costs for different types of misclassification errors. The few works are generally based on calculating the uncertainty of each instance by probability estimation, and can suffer from the inaccuracy of the estimation. In this paper, we propose a novel active learning algorithm that relies on a different way of calculating the uncertainty. The algorithm is based on our newly-proposed cost embedding approach (CE) for CSMCC. CE embeds the cost information in the distance measure of a special hidden space with non-metric multidimensional scaling, and deals with both symmetric and asymmetric cost information by our carefully designed mirroring trick. The embedding allows the proposed algorithm, active learning with cost embedding (ALCE), to define a cost-sensitive uncertainty measure from the distance in the hidden space. Extensive experimental results demonstrate that ALCE selects more useful instances by taking the cost information into account through the embedding and is superior to existing cost-sensitive active learning algorithms.**

I. Introduction

Multiclass classification (MCC) algorithms intend to learn a classifier from numerous instances and their corresponding labels. In many real-world applications, the labels are expensive to obtain. Active learning is thus introduced to reduce the labeling effort [1]. Active learning algorithms iteratively select some instances to be labeled based on some strategies, and aim to achieve high accuracy with a few labeled instances.

Uncertainty sampling is an important and popular family of active learning strategies [2]. The key idea of uncertainty sampling is to select the instances that seem less certain because the labels of such instances usually provide more information to improve the accuracy. The uncertainty of an instance can be defined in different ways, such as the probabilistic measures [3]–[5] and the non-probabilistic ones [6]. In addition to uncertainty sampling, many other active learning strategies take some uncertainty measure as a core part for deciding which instances to select [7]–[9]. It is thus important to consider proper uncertainty measures when designing active learning algorithms.

Most of the existing uncertainty sampling algorithms are designed for "regular" MCC, which means that the *costs* of different kinds of misclassification errors are equal. Nevertheless, regular MCC may not satisfy some needs within real-world applications. For example, consider a three-class classification problem for predicting the state of a patient from {healthy, cold-infected, Zika-infected}. The cost of predicting a Zika-infected patient as healthy shall be remarkably larger than the cost of predicting a healthy patient as cold-infected, because the former may cause more serious public-health troubles. The cost-sensitive MCC (CSMCC) problem matches such needs and has been attracting much research attention in recent years [10]–[13]. CSMCC takes the application-specific misclassification costs into account to learn the classifier and make cost-sensitive predictions.

Although there are many works for CSMCC [10]–[13], only two works focus on active learning for CSMCC—that is, the cost-sensitive multiclass active learning (CSMCAL) problem [14], [15]. Furthermore, both works for CSMCAL are based on uncertainty sampling with the probabilistic measures. It is thus not clear whether better CSMCAL algorithms can be designed with non-probabilistic uncertainty measures.

In this work, we derive a novel non-probabilistic uncertainty sampling algorithm for CSMCAL. We first design a novel CSMCC algorithm called cost embedding (CE), which embeds the cost information in the *distance measure* in a special hidden space by non-metric multidimensional scaling. We further propose a *mirroring trick* to let CE embed the possibly asymmetric cost information in the symmetric distance measure. Then, we define an appropriate cost-sensitive uncertainty measure through CE. The measure forms the backbone of our proposed algorithm, called active learning with cost embedding (ALCE). Extensive experimental results on CSMCC benchmarks demonstrate that ALCE is superior to not only cost-insensitive active learning algorithms but also existing CSMCAL algorithms.

This paper is organized as follows. Section II formalizes the CSMCAL problem and reviews the related works. Section III illustrates the proposed ALCE algorithm. We discuss the experimental results in Section IV and conclude in Section V.

II. Preliminary

There are two setups of active learning for multiclass classification: stream-based and pool-based [1]. In the stream-based setup, the instance comes in sequence, and the algorithm has to immediately decide whether to query the label of the instance

or ignore it. The pool-based setup is more flexible in terms of how data can be accessed, and will be considered in this paper. For pool-based multiclass active learning, a labeled pool and an unlabeled pool are presented to the algorithm. In each iteration, the algorithm selects one instance from the unlabeled pool to query its label. More precisely, let $\mathbf{x} \in \mathcal{X} \subseteq \mathbb{R}^d$ denote the instance and $y \in \mathcal{Y} = \{c_1, c_2, ..., c_K\}$ denote the label. Given the labeled pool $\mathcal{D}_l = \{(\mathbf{x}^{(n)}, y^{(n)})\}_{n=1}^{N_l}$ and the unlabeled pool $\mathcal{D}_u = \{\mathbf{x}^{(n)}\}_{n=1}^{N_u}$, the algorithm first learns a classifier $f^{(0)}$ from $\mathcal{D}_l$. For each iteration $t = 1, 2, ..., T$, the algorithm selects an instance $\mathbf{x}_s \in \mathcal{D}_u$ to query its label y_s based on $\mathcal{D}_l$, $\mathcal{D}_u$, and $f^{(t-1)}$. Next, $\mathbf{x}_s$ is removed from the unlabeled pool $\mathcal{D}_u$ and $(\mathbf{x}_s, y_s)$ is added to the labeled pool $\mathcal{D}_l$. The algorithm then learns $f^{(t)}$ based on the updated $\mathcal{D}_l$ and goes to the next iteration. The objective of the algorithm is to make the test accuracy of $f^{(1)}, f^{(2)}, ..., f^{(T)}$ as good as possible.

There are lots of strategies for selecting the instances to query the labels. Uncertainty sampling is an important and popular family of strategies [2]. Uncertainty sampling algorithms assume that the classifier $f^{(t-1)}$ only needs fine-tuning around the decision boundary and hence query the label of the most uncertain instance near the decision boundary of $f^{(t-1)}$. For example, Jing et al. [3] estimate the probabilities of the classes for the unlabeled instances and use *entropy* to evaluate the uncertainty. Some other works use different definitions of the uncertainty based on the probabilities, such as *least confidence* [4] and *margin* [5]. Those strategies rely on an accurate estimate of the probabilities. However, when considering complicated classifiers such as the kernel ones, probability estimation becomes a challenging problem, and hence these strategies may suffer from the inaccurate estimate and weaker performance.

Some other works define the uncertainty without probability estimation. For example, Tong and Koller [6] define the uncertainty by the *distance* between the instance and the decision boundary of SVM [16] for binary active learning (i.e. $K = 2$). Using *distance* as the uncertainty can avoid the challenge of probability estimation. Nevertheless, for multiclass active learning, there can be multiple decision boundaries when the classifier comes from the one-versus-one or one-versus-all reductions. Then, the challenge resides in defining uncertainty by the distance to the *multiple boundaries*.

Another popular family of strategies is representative sampling, which considers both the uncertainty and the representativeness to select the instances. For example, Settles and Craven [7] use *information density* to estimate the representativeness and give the instances different weights to calculate the uncertainty. Huang et al. [8] measure the representativeness by estimating the possible label for the unlabeled instances. *Clustering* and *hierarchical clustering* are also used for measuring the representativeness [9], [17]. Although the representative sampling algorithms are named after the representativeness, most of them still extend or adopt the concept of uncertainty sampling [7]–[9]. Thus, designing a good uncertainty sampling algorithm is arguably an important task for active learning.

Most of the existing uncertainty sampling algorithms are designed for the regular MCC, which means that the *costs* of different kinds of misclassification errors are equal. However, in some real-world applications, the costs shall be different. Therefore, it is important to study cost-sensitive multiclass classification (CSMCC) [10], [11] and its active learning setup, cost-sensitive multiclass active learning (CSMCAL). Given the different misclassification costs, CSMCAL considers the costs when selecting the instances and learning the classifiers.

In this paper, we follows the CSMCAL setup introduced by Chen and Lin [14]. Let the cost matrix $\mathbf{C}$ be an $K \times K$ matrix with $\mathbf{C}_{i,j}$ representing the cost when c_i is the ground truth and c_j is the prediction. It is natural to assume that $\mathbf{C}$ is non-negative and zero-diagonal. The objective of the CSMCAL algorithm is to select the useful instance in each iteration such that the test cost $\mathbf{C}_{y,f^{(t)}(\mathbf{x})}$ is low with respect to the distribution that generates $(\mathbf{x}, y)$ for $f^{(1)}, f^{(2)}, ..., f^{(T)}$.

Chen and Lin [14] extend the probabilistic uncertainty, *least confidence* and *margin*, to cost-sensitive versions by Bayesian inference. Agarwal [15] proposes a cost-sensitive algorithm based on *margin* with theoretical guarantees, but the algorithm is for stream-based active learning rather than the pool-based active learning. It is worth noting that the existing works for CSMCAL are all the extensions of uncertainty sampling. In addition, they all define the cost-sensitive uncertainty by probabilities. Therefore, these algorithms encounter the same challenging problem as the regular uncertainty sampling: probability estimation. Motivated by this, we study the design of non-probabilistic uncertainty sampling algorithms for pool-based CSMCAL. We define the cost-sensitive uncertainty by the *distance measure* in a special hidden space, and conquer the difficulty in combining the distances from the multiple decision boundaries by applying a totally different embedding view for CSMCC.

III. Proposed Algorithm

In this section, we first propose a novel CSMCC algorithm which takes the cost matrix $\mathbf{C}$ into account when learning the classifier f from the labeled pool $\mathcal{D}_l = \{(\mathbf{x}^{(n)}, y^{(n)})\}_{n=1}^{N_l}$. The proposed algorithm relies on a totally different embedding view of CSMCC. Then, we propose an algorithm for CSMCAL by defining a cost-sensitive uncertainty based on the newly-proposed CSMCC algorithm.

A. Embedding View for CSMCC

Unlike general CSMCC algorithms, our classifier f is not directly learned from the labeled pool $\mathcal{D}_l$ and the cost matrix $\mathbf{C}$. Alternately, we construct a hidden structure which embeds the cost information of $\mathbf{C}$, and then learn the classifier f through the hidden structure. In particular, in the training stage, for classes $c_1, c_2, ..., c_K$, we determine K corresponding hidden points $\mathbf{z}_1, \mathbf{z}_2, ..., \mathbf{z}_K$ in a M-dimensional hidden space $\mathcal{Z}$ to preserve the cost information of $\mathbf{C}$. Then, we learn a multi-target regressor g from $\{\mathbf{x}^{(n)}, \mathbf{z}^{(n)}\}_{n=1}^{N_l}$, where $\mathbf{z}^{(n)}$ is the corresponding hidden point to $y^{(n)}$. In the predicting

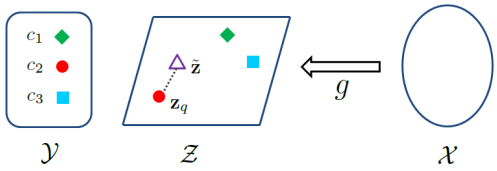


Fig. 1. Embedding view for CSMCC

stage, for any testing instance $\mathbf{x}$, we first obtain the predicted hidden point $\tilde{\mathbf{z}} = g(\mathbf{x})$. Next, we find the nearest hidden point of $\tilde{\mathbf{z}}$ from $\{\mathbf{z}_1, \mathbf{z}_2, ..., \mathbf{z}_K\}$, which is denoted as $\mathbf{z}_q$. The final prediction $\tilde{y}$ is set as c_q, which is the class corresponding to $\mathbf{z}_q$. In other words, our classifier $f = \phi \circ g$, where ϕ is the nearest neighbor function $\phi(\cdot) = \operatorname{argmin}_k d(\mathbf{z}_k, \cdot)$. Figure 1 illustrates the embedding view for CSMCC.

B. Properties of Hidden Points

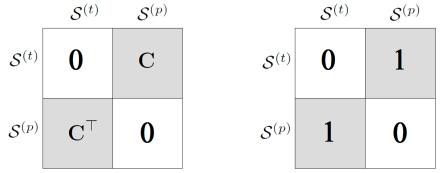
Now, we discuss how the hidden points preserve the cost information. For a testing instance $\mathbf{x}$ and its label $y = c_r$, we assume that c_i is the better prediction than c_j ($\mathbf{C}_{r,i} < \mathbf{C}_{r,j}$). Recall that the prediction $\tilde{y} = c_q$ is decided by $\mathbf{z}_q$, the nearest hidden point of $\tilde{\mathbf{z}} = g(\mathbf{x})$. If unfortunately, $\tilde{\mathbf{z}}$ is inaccurate, then the nearest hidden point $\mathbf{z}_q \neq \mathbf{z}_r$ and we make the wrong prediction ($\tilde{y} = c_q \neq c_r$). In this case, to reduce the cost, we prefer predict c_i rather than c_j (since $\mathbf{C}_{r,i} < \mathbf{C}_{r,j}$). This motivates us to let $d(\mathbf{z}_r, \mathbf{z}_i) < d(\mathbf{z}_r, \mathbf{z}_j)$, where d represents the Euclidean distance in the hidden space. Because if $\mathbf{z}_r$ is closer to $\mathbf{z}_i$ than $\mathbf{z}_j$, we are more likely to predict c_i than c_j according to the nearest neighbor decision.

Based on this idea, we would like to determine the hidden points $\mathbf{z}_1, \mathbf{z}_2, ..., \mathbf{z}_K$ such that $d(\mathbf{z}_i, \mathbf{z}_j) < d(\mathbf{z}_{i'}, \mathbf{z}_{j'})$ iff $\mathbf{C}_{i,j} < \mathbf{C}_{i',j'}$. In other words, the magnitude-order of the costs is embedded in the distance between the hidden points, and the larger (smaller) distance implies the higher (lower) cost.

C. Non-metric Multidimensional Scaling

In this section, we introduce non-metric multidimensional scaling (NMDS) [18], which is helpful to determine the desired hidden points. NMDS is one variant of multidimensional scaling (MDS) [19], which is a classic manifold learning approach to identify the hidden structure of L given objects with non-metric dissimilarities. More specifically, let $\mathbf{\Delta}$ be an $L \times L$ matrix, where $\mathbf{\Delta}_{i,j}$ represents the *symmetric* dissimilarity of the i-th object and j-th object. NMDS attempts to finds L target points $\mathbf{u}_1, \mathbf{u}_2, ..., \mathbf{u}_L$ in the M-dimensional target space such that $d(\mathbf{u}_i, \mathbf{u}_j) < d(\mathbf{u}_{i'}, \mathbf{u}_{j'})$ iff $\mathbf{\Delta}_{i,j} < \mathbf{\Delta}_{i',j'}$. In other words, the magnitude-order of the dissimilarities is preserved in the distance between the target points. We can adjust the weight of each pair (i, j) by defining an $L \times L$ weight matrix with $\mathbf{W}_{i,j}$ being the weight. Note that both $\mathbf{\Delta}$ and $\mathbf{W}$ are limited to be symmetric, non-negative, and zero-diagonal.

There are many algorithms available in the literature to solve NMDS. A representative one is Scaling by MAjorizing a COmplicated Function (SMACOF) [20], which trains the isotonic regressors and find the target points iteratively. In general, the complexity of SMACOF is $\mathcal{O}(L^3)$, but there is often room for speeding up with special weight matrices $\mathbf{W}$.



(a) $\mathbf{\Delta}$ (b) $\mathbf{W}$

Fig. 2. Constructions of $\mathbf{\Delta}$ and $\mathbf{W}$

D. Determining Hidden Points by Solving NMDS

The objective of NMDS is to find the target points to preserve the dissimilarities of the objects, which is similar to our objective to determine the hidden points to embed the costs of the classes. It is natural to take the classes as the objects and take the cost matrix $\mathbf{C}$ as the dissimilarities matrix $\mathbf{\Delta}$. Then the target points obtained by NMDS can be the desired hidden points. Nevertheless, the dissimilarity matrix $\mathbf{\Delta}$ needs to be symmetric while the cost matrix $\mathbf{C}$ can be asymmetric, that is, $\mathbf{C}_{i,j} \neq \mathbf{C}_{j,i}$. To resolve this difficulty, we propose a *mirroring trick* to deal with the asymmetric cost matrix.

The asymmetric cost matrix $\mathbf{C}$ implies that each class has two roles: as the ground truth and as the prediction. The cost is different given that the class serves different roles. For the class c_i, we use the notation $c_i^{(t)}$ and $c_i^{(p)}$ when we view c_i as the ground truth and the prediction respectively, and use the notation $\mathcal{S}^{(t)}$ and $\mathcal{S}^{(p)}$ to denote the sets $\{c_i^{(t)}\}_{i=1}^K$ and $\{c_i^{(p)}\}_{i=1}^K$. Note that the two mirrored classes $c_i^{(t)}$ and $c_i^{(p)}$ are in fact the same, but they have different meanings. We consider $2K$ objects with the first K objects being the elements in $\mathcal{S}^{(t)}$ and the last K objects being the elements in $\mathcal{S}^{(p)}$. Now, $\mathbf{C}_{i,j}$ can be viewed as the dissimilarity between $c_i^{(t)}$ and $c_j^{(p)}$, which is symmetric for them. Similarly, $\mathbf{C}_{j,i}$ can be viewed as the symmetric dissimilarity between $c_i^{(p)}$ and $c_j^{(t)}$. In other words, the costs can be viewed as the dissimilarities between the elements in $\mathcal{S}^{(t)}$ and the elements in $\mathcal{S}^{(p)}$.

On the basis of this idea, we construct $\mathbf{\Delta}$ and $\mathbf{W}$ as follows (Figure 2). Let $\mathbf{\Delta}$ and $\mathbf{W}$ be the $2K \times 2K$ matrices. Given that we are concerned only about the dissimilarities between the elements in $\mathcal{S}^{(t)}$ and $\mathcal{S}^{(p)}$, we set the top-right part and the bottom-left part of weight matrix $\mathbf{W}$ to ones, and set the top-left and the bottom-right parts of $\mathbf{W}$ to zeros (and the corresponding parts of $\mathbf{\Delta}$ conveniently to zeros as well). Then, we set the top-right part and the bottom-left part of $\mathbf{\Delta}$ to be the corresponding cost. That is,

$$\mathbf{W}_{i,j} = \begin{cases} 1 & \text{if } (i,j) \text{ in the top-right part} \\ 1 & \text{if } (i,j) \text{ in the bottom-left part} \\ 0 & \text{otherwise} \end{cases}$$

$$\mathbf{\Delta}_{i,j} = \begin{cases} \mathbf{C}_{i,j-|\mathcal{S}|} & \text{if } (i,j) \text{ in the top-right part} \\ \mathbf{C}_{i-|\mathcal{S}|,j} & \text{if } (i,j) \text{ in the bottom-left part} \\ 0 & \text{otherwise} \end{cases}$$

Note that the top-right part and the bottom-left part of $\mathbf{\Delta}$ are in fact $\mathbf{C}$ and $\mathbf{C}^\top$ respectively.

By solving NMDS with the above-mentioned $\mathbf{\Delta}$ and $\mathbf{W}$, we obtain the target points of $c_i^{(t)}$ and $c_i^{(p)}$, which are denoted as $\mathbf{u}_i^{(t)}$ and $\mathbf{u}_i^{(p)}$ respectively. We further use $\mathcal{U}^{(t)}$ and $\mathcal{U}^{(p)}$ to denote the target point set $\{\mathbf{u}_i^{(t)}\}_{i=1}^K$ and $\{\mathbf{u}_i^{(p)}\}_{i=1}^K$. The cost information is embedded in the distances between the target points in $\mathcal{U}^{(t)}$ and $\mathcal{U}^{(p)}$. Since we view each class c_i as two roles ($c_i^{(t)}$ and $c_i^{(p)}$), now, we have to decide which target point ($\mathbf{u}_i^{(t)}$ or $\mathbf{u}_i^{(p)}$) is the hidden point $\mathbf{z}_i$ of the class c_i. Recall that the goal of the hidden points is to train a multi-target regressor g and obtain $\tilde{\mathbf{z}}$, the "predicted" hidden point. Therefore, we take $\mathbf{u}_i^{(p)}$, which serves the role of the prediction, as the hidden point of c_i. Accordingly, the nearest hidden point $\mathbf{z}_q$ should be the role of the ground truth because the cost information is embedded in the distance between the target points of these two roles. Hence, we find the nearest hidden point $\mathbf{z}_q$ from $\mathcal{U}^{(t)}$, which serves the role of the ground truth.

Now, we can learn a cost-sensitive classifier from the obtained hidden points which preserve the cost information, and make the cost-sensitive prediction from the nearest hidden point. We call this approach cost embedding (CE).

E. Cost-sensitive Uncertainty for CSMCAL

We are going to define the cost-sensitive uncertainty with the help of CE. As mentioned above, for the test instance $\mathbf{x}$ and its predicted hidden point $\tilde{\mathbf{z}} = g(\mathbf{x})$, CE finds the nearest hidden point $\mathbf{z}_q$ of $\tilde{\mathbf{z}}$ from $\mathcal{U}^{(t)}$ such that $d(\mathbf{z}_q, \tilde{\mathbf{z}})$ is the smallest. Recall that in CE, the distance $d(\mathbf{z}_q, \tilde{\mathbf{z}})$ contains the cost information. Specifically, larger (smaller) $d(\mathbf{z}_q, \tilde{\mathbf{z}})$ implies the larger (smaller) cost for the prediction $\tilde{y} = c_q$. Therefore, $d(\mathbf{z}_q, \tilde{\mathbf{z}})$ can be viewed as an estimated cost. Ideally, $d(\mathbf{z}_q, \tilde{\mathbf{z}})$ should be small because we assume there is no cost for the correct prediction (zero-diagonal for $\mathbf{C}$). In case $d(\mathbf{z}_q, \tilde{\mathbf{z}})$ is large, we expect that there is a high cost when taking c_q as the prediction despite the fact that c_q is the best choice. In other words, we are "uncertain" about the prediction for this instance. Based on this idea, we define the cost-sensitive uncertainty of an instance $\mathbf{x}$ as $d(\mathbf{z}_q, \tilde{\mathbf{z}})$.

With the defined cost-sensitive uncertainty, we propose a novel uncertainty sampling algorithm for CSMCAL, called active learning with cost embedding (ALCE). In each iteration, ALCE selects the instance with the highest cost-sensitive uncertainty and queries its label. The cost-sensitive uncertainty makes ALCE be able to find useful instances with respect to the cost.

IV. EXPERIMENTS

We conduct experiments on eight public datasets[1] to validate the proposed algorithm. Table I lists the basic information of the datasets. Note that these datasets are in fact for regular MCC. Thus, we follow previous works [10]–[12], [14] and adopt the randomize proportional (RP) cost-generation procedure that was proposed by Beygelzimer et al. [21] to simulate CSMCC. The diagonal elements $\mathbf{C}_{i,i}$ are set to zero and the other elements $\mathbf{C}_{i,j}$ are uniformly sample from $\left[0, 2000\frac{|\{n:y^{(n)}=i\}|}{|\{n:y^{(n)}=j\}|}\right]$. We acknowledge that the RP procedure may not fully reflect realistic application needs. However, we still adopt RP because it is a longstanding benchmark for comparing CSMCC algorithms.

[1]Available from http://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/

TABLE I
BASIC INFORMATION OF DATASETS

dataset	# of class	# of instances	# of features
vehicle	4	846	18
glass	6	264	9
svmguide4	6	612	10
satimage	6	6435	36
segment	7	2310	19
yeast	10	1484	8
usps	10	9298	256
vowel	11	990	10
letter	26	20000	16

TABLE II
t-TEST AT 95% CONFIDENCE LEVEL FOR ALCE-N (# WIN/# TIE/# LOSS)

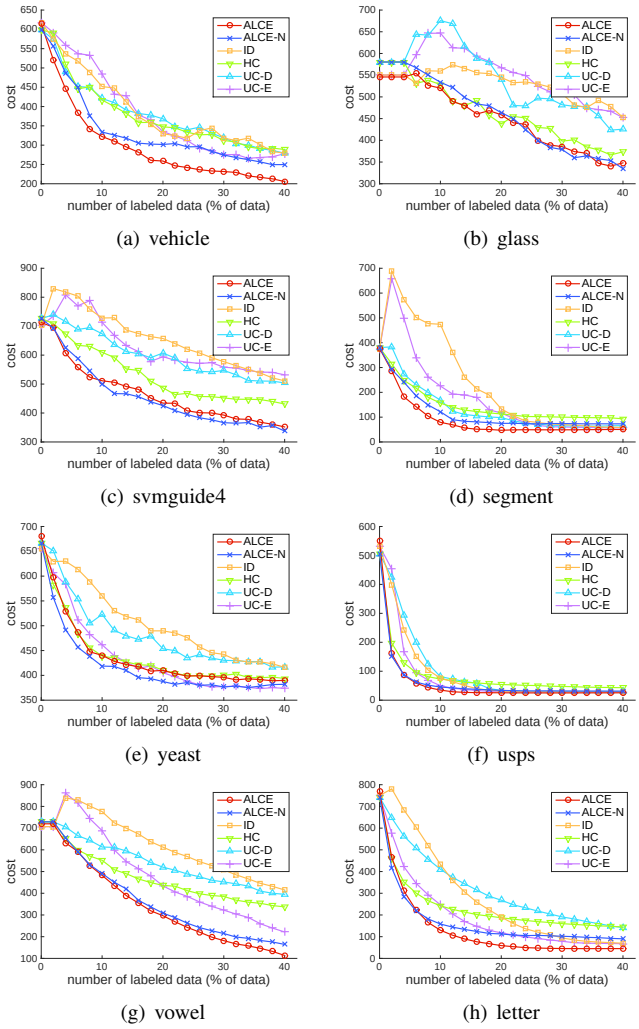
	number of labeled data (% of data)							
algorithm	5%	10%	15%	20%	25%	30%	40%	total
HC	2/6/0	6/2/0	4/4/0	4/4/0	4/4/0	5/3/0	5/3/0	30/26/0
ID	6/2/0	7/1/0	7/1/0	5/3/0	5/2/1	4/2/2	3/3/2	37/14/5
UC-E	6/2/0	6/2/0	5/3/0	3/4/1	3/3/2	3/3/2	3/3/2	29/20/7
UC-D	3/5/0	8/0/0	8/0/0	4/4/0	4/4/0	4/4/0	4/4/0	35/21/0
total	17/15/0	27/5/0	24/8/0	16/15/1	16/13/3	16/12/4	15/13/4	

All the following experimental results are the average results of 20 experiments. In each run of the experiments, we randomly sample 60% of data as the training set and the other 40% of data as the testing set. Then, we randomly select one instance of each class in the training set as the initial labeled pool $\mathcal{D}_l$ and let the other instances be the unlabeled pool $\mathcal{D}_u$.

A. Comparison with Cost-insensitive Algorithms

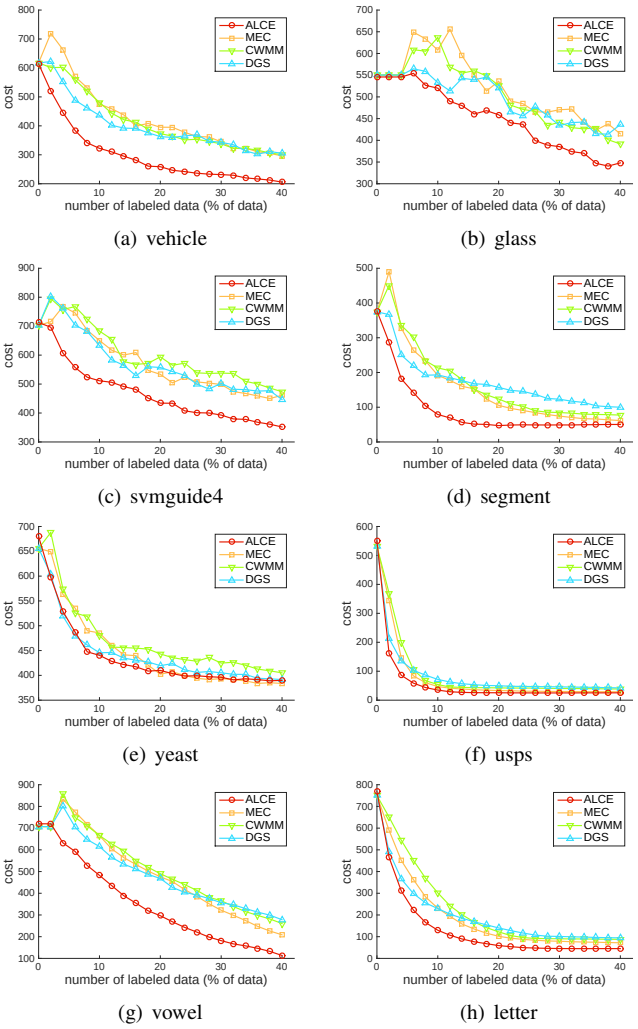
We first demonstrate that ALCE indeed select the instance which contains more information with respect to the cost. We compare ALCE with the following algorithms which do *not* consider the costs: (1) UC-D: uncertainty sampling with *distance* as the uncertainty [6] (2) UC-E: uncertainty sampling with *entropy* as the uncertainty [3] (3) ID: representative sampling with *information density* as the instance weights [7] (4) HC: Representative sampling by *hierarchy clustering* [17]. We use SVM [16] as the classifier for UC-D and use SVM with probability estimation [22] for UC-E, ID, and HC. For ALCE, we set $M = K$ and use M single-target ridge regressors as the multi-target regressor. Note that the classifier of ALCE (obtained by CE) is cost-sensitive. To achieve a fair comparison, we also compare ALCE-N, which uses the same selection strategy as ALCE but obtains the classifier with SVM (cost-insensitive) rather than CE. We consider RBF kernel for all the classifiers and regressors and the parameters are set to the default parameters.

Figure 3 shows the test cost versus different percentages of labeled data. From the figure, we first notice that among the algorithms which use the cost-insensitive classifiers (UC-D, UC-E, ID, HC, and ALCE-N), ALCE-N has the best performance in most of the datasets. We further compare ALCE-N with these algorithms based on the t-test at 95% confidence level in Table II. The results justify that the selection strategy by the proposed cost-sensitive uncertainty is indeed useful.



(a) vehicle (b) glass (c) svmguide4 (d) segment (e) yeast (f) usps (g) vowel (h) letter

Fig. 3. Test cost of ALCE, ALCE-N and other cost-insensitive algorithms



(a) vehicle (b) glass (c) svmguide4 (d) segment (e) yeast (f) usps (g) vowel (h) letter

Fig. 4. Test cost of ALCE and other cost-sensitive algorithms

TABLE III
t-TEST AT 95% CONFIDENCE LEVEL FOR ALCE (# WIN/# TIE/# LOSS)

algorithm	number of labeled data (% of data)							total
	5%	10%	15%	20%	25%	30%	40%	
MEC	6/2/0	7/1/0	6/2/0	6/2/0	6/2/0	6/2/0	6/2/0	43/13/0
CWMM	6/2/0	7/1/0	6/2/0	6/2/0	6/2/0	6/2/0	6/2/0	43/13/0
DGS	6/2/0	6/2/0	5/3/0	6/2/0	6/2/0	6/2/0	7/1/0	42/14/0
total	18/6/0	20/4/0	17/7/0	18/6/0	18/6/0	18/6/0	19/5/0	

From Figure 3, we see that ALCE is generally better and more stable than ALCE-N. This validates that the proposed CE approach is able to catch the cost information to learn a cost-sensitive classifier and make the cost-sensitive predictions.

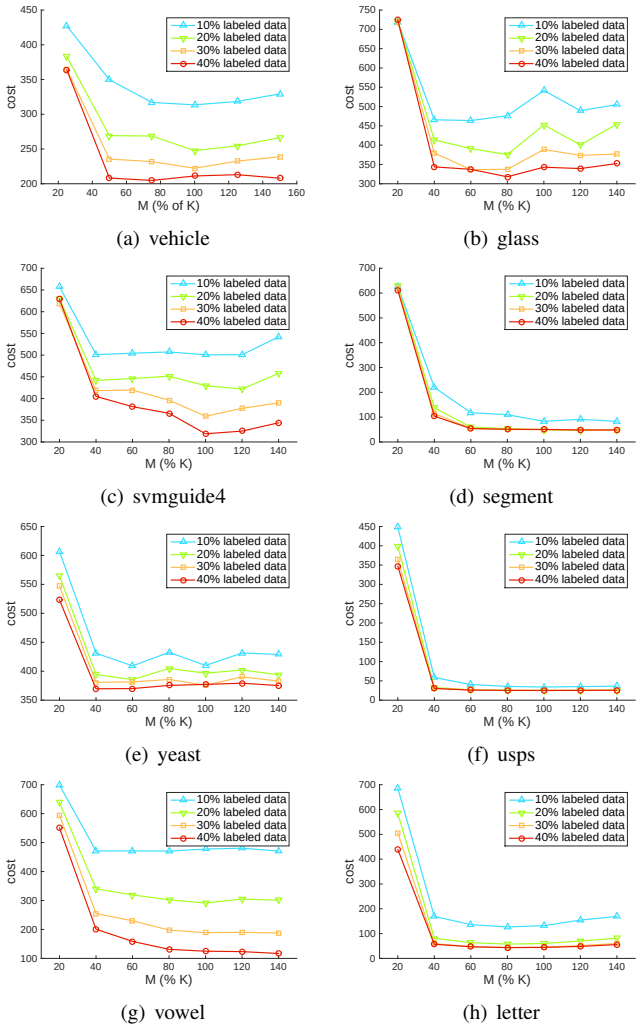
B. Comparison with Cost-sensitive Algorithms

Next, we compare ALCE with the existing cost-sensitive algorithms: (1) MEC: *Maximum expected cost* proposed by Chen and Lin [14] (2) CWMM: *Cost-weighted minimum margin* proposed by Chen and Lin [14] (3) DGS: *DGS selection rule* proposed by Agarwal [15]. These three algorithms are all probabilistic uncertainty sampling algorithms, hence we use SVM with probability estimation [22] for them. We consider RBF kernel and the parameters are set to the default parameters. Note that DGS is for stream-based active learning. Thus, in each iteration, we keep uniformly and randomly sampling an instance from $\mathcal{D}_u$ to present to DGS until one instance is decided to query. This simulation is also used by Li et al. [23] to compare pool-based active learning algorithms and stream-based active learning algorithms.

Figure 4 shows the test cost versus different percentages of labeled data. ALCE outperforms the cost-sensitive algorithms (MEC, CWMM, DGS) in most of the datasets. Table III lists the t-test results of ALCE versus the cost-sensitive algorithms based on 95% confidence level. The results again demonstrate the superiority of ALCE. ALCE does not rely on the probability estimation and hence could perform better for kernel classifiers like SVM.

C. Dimension of Hidden Space

Finally, we discuss the influence of the dimension of the hidden space. Figure 5 shows the results of ALCE with different dimension M. From the figure, we notice that the larger dimension leads to the better performance in general. Nevertheless, when M is greater than 60% of K, the improvement is insignificant. This implies that setting M as 60% of K is generally sufficient to embed the cost information.



(a) vehicle (b) glass

(c) svmguide4 (d) segment

(e) yeast (f) usps

(g) vowel (h) letter

Fig. 5. Test cost of ALCE with different dimensions of the hidden space

V. Conclusion

We proposed a novel uncertainty sampling algorithm for cost-sensitive multiclass active learning called active learning with cost embedding (ALCE). ALCE is based on our newly-proposed cost embedding approach (CE) for the cost-sensitive multiclass classification (CSMCC). CE transforms each possible label to a hidden point in a special hidden space and embeds the cost information in the distance measure of the hidden space with non-metric multidimensional scaling. By our carefully designed mirroring trick, CE deals with both symmetric and asymmetric cost information. The embedding allows ALCE to define the cost-sensitive uncertainty directly from the distance in the hidden space and select more important instances to achieve the cost-sensitivity. Extensive experimental results not only demonstrate that ALCE indeed selects more useful instances by taking the cost information into account through the embedding, but also show the superiority of ALCE to the existing cost-sensitive active learning algorithms.

Acknowledgment

We thank the anonymous reviewers for valuable suggestions. This material is based upon work supported by the Air Force Office of Scientific Research, Asian Office of Aerospace Research and Development (AOARD) under award number FA2386-15-1-4012, and by the Ministry of Science and Technology of Taiwan under number MOST 103-2221-E-002-148-MY3.

References

[1] B. Settles, "Active learning literature survey," *University of Wisconsin, Madison*, vol. 52, no. 55-66, p. 11, 2010.

[2] D. D. Lewis and W. A. Gale, "A sequential algorithm for training text classifiers," in *SIGIR*, 1994, pp. 3–12.

[3] F. Jing, M. Li, H. Zhang, and B. Zhang, "Entropy-based active learning with support vector machines for content-based image retrieval," in *ICME*, 2004, pp. 85–88.

[4] A. Culotta and A. McCallum, "Reducing labeling effort for structured prediction tasks," in *AAAI*, 2005, pp. 746–751.

[5] T. Scheffer, C. Decomain, and S. Wrobel, "Active hidden markov models for information extraction," in *IDA*, 2001, pp. 309–318.

[6] S. Tong and D. Koller, "Support vector machine active learning with applications to text classification," *Journal of Machine Learning Research*, vol. 2, pp. 45–66, 2001.

[7] B. Settles and M. Craven, "An analysis of active learning strategies for sequence labeling tasks," in *EMNLP*, 2008, pp. 1070–1079.

[8] S.-J. Huang, R. Jin, and Z.-H. Zhou, "Active learning by querying informative and representative examples," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 36, no. 10, pp. 1936–1949, 2014.

[9] Z. Xu, K. Yu, V. Tresp, X. Xu, and J. Wang, "Representative sampling for text classification using support vector machines," in *ECIR*, 2003, pp. 393–407.

[10] P. M. Domingos, "Metacost: A general method for making classifiers cost-sensitive," in *KDD*, 1999, pp. 155–164.

[11] N. Abe, B. Zadrozny, and J. Langford, "An iterative method for multi-class cost-sensitive learning," in *KDD*, 2004, pp. 3–11.

[12] H.-H. Tu and H.-T. Lin, "One-sided support vector regression for multiclass cost-sensitive classification," in *ICML*, 2010, pp. 1095–1102.

[13] A. Beygelzimer, J. Langford, and P. Ravikumar, "Error-correcting tournaments," in *ALT*, 2009, pp. 247–262.

[14] P.-L. Chen and H.-T. Lin, "Active learning for multiclass cost-sensitive classification using probabilistic models," in *TAAI*, 2013, pp. 13–18.

[15] A. Agarwal, "Selective sampling algorithms for cost-sensitive multiclass prediction," in *ICML*, 2013, pp. 1220–1228.

[16] V. Vapnik, *Statistical learning theory*. Wiley, 1998.

[17] S. Dasgupta and D. J. Hsu, "Hierarchical sampling for active learning," in *ICML*, 2008, pp. 208–215.

[18] J. B. Kruskal, "Nonmetric multidimensional scaling: a numerical method," *Psychometrika*, vol. 29, no. 2, pp. 115–129, 1964.

[19] W. S. Torgerson, "Multidimensional scaling: I. theory and method," *Psychometrika*, vol. 17, no. 4, pp. 401–419, 1952.

[20] J. De Leeuw, "Applications of convex analysis to multidimensional scaling," *Recent Developments in Statistics*, pp. 133–145, 1977.

[21] A. Beygelzimer, V. Dani, T. P. Hayes, J. Langford, and B. Zadrozny, "Error limiting reductions between classification tasks," in *ICML*, 2005, pp. 49–56.

[22] T.-F. Wu, C.-J. Lin, and R. C. Weng, "Probability estimates for multi-class classification by pairwise coupling," *Journal of Machine Learning Research*, vol. 5, pp. 975–1005, 2004.

[23] C.-L. Li, C.-S. Ferng, and H.-T. Lin, "Active learning using hint information," *Neural Computation*, vol. 27, no. 8, pp. 1738–1765, 2015.