Spatially Cost-sensitive Active Learning

Alexander Liu* Goo Jun* Joydeep Ghosh*

Abstract

In active learning, one attempts to maximize classifier performance for a given number of labeled training points by allowing the active learning algorithm to choose which points should be labeled. Typically, when the active learner requests labels for the selected points, it assumes that all points require the same amount of effort to label and that the cost of labeling a point is independent of other selected points. In spatially distributed data such as hyperspectral imagery for land-cover classification, the act of labeling a point (i.e., determining the land-type) may involve physically traveling to a location and determining ground truth. In this case, both assumptions about label acquisition costs made by traditional active learning are broken, since costs will depend on physical locations and accessibility of all the visited points. This paper formulates and analyzes the novel problem of performing active learning on spatial data where label acquisition costs are proportional to distance traveled.

1 Introduction

In active learning, typically it is assumed that: 1) the cost of acquiring the label for a particular point is independent of the label acquisition cost for all other points and that 2) label acquisition costs for all points are equal. When applying active learning to spatial data, these assumptions may not be true. For example, for classification of land-cover using hyperspectral data [9], acquiring labels may involve traveling to a particular location and performing some sort of test such as determining the type of land at that point or collecting various samples, such as soil, water, or foliage samples that require physical access. Traveling to this point incurs some type of cost (e.g., gas or time) proportional to the distance traveled. The distance traveled also depends on the order in which one visits the points that need to be labeled, meaning that the label acquisition cost for a particular point is dependent on other, previously visited points.

The problem we address in this paper is as follows: given a set of unlabeled, spatially distributed data, can we create and apply an active learning approach with good performance that incorporates the fact that label acquisition costs are not independent and identically-distributed?

We focus on one particular problem setting for active learning on spatially distributed hyperspectral data. In this setting, there is some "home" location that the labeler must physically return to after labeling some fixed number of points[1]. The amount of effort required to label a point is proportional to the additional distance required to travel to that point. Whether the effort is measured in terms of time spent traveling or gas expended by some vehicle does not affect our analysis. The labeler continues to label a fixed number of points on each iteration or round of active learning, returning home after each round, until all points are expended. Other varations of this problem are possible and will be discussed in section 7.

This paper presents solutions based on combining standard active learning approaches with methods used for approximating solutions to the Traveling Salesman Problem(TSP) and one of its variants, the Traveling Salesman Problem with Profits(TSPP). Details on both active learning and the traveling salesman problem will be discussed in sections 2 and 3. We then discuss our proposed approaches in section 4. Experimental results (section 6) are presented for two hyperspectral datasets. Our approach is general and does not depend on a particular solution to either active learning or the traveling salesman problem (with profits). Our approach is also not restricted to hyperspectral data analysis, and should be applicable to any dataset where label acqusition costs have spatial dependencies.

2 Related work

Most active learning algorithms basically follow an iterative "pick-and-retrain" framework, where, on each iteration, n unlabeled samples are picked, the ground truth is obtained for these n samples, and the classifier is re-trained using the augmented labeled data. One of the main differences in the various proposed active learning methods is how the different approaches determine which n unlabeled points are chosen. While many

*Department of Electrical and Computer Engineering, University of Texas at Austin; {aliu, gjun, ghosh}@ece.utexas.edu

[1] In an actual application, the "home" location may correspond to the labeler's base of operations or where the labeler's vehicle is stored/refueled

popular approaches exist (e.g., [17], [12]), we use the one proposed by Lewis and Gale [11] called uncertainty sampling, which requires a classifier capable of producing uncertainty levels for unlabeled samples. Appropriate uncertainty measures vary from classifier to classifier [18]. In this paper, we use both linear discriminant analysis (LDA) and support vector machine (SVM) classifiers. We discuss the uncertainty scores used with each classifier in section 3.1.

Although many active learning strategies have been proposed during the last 15 years, there exist few algorithms that consider spatial characteristics of unlabeled samples. Rajan *et al* proposed an active learning algorithm [15] for hyperspectral data that adapts a classifier for spatial variation of spectral signatures. However, it does not take into account any form of varying label acquisition costs based on spatial data. An active learning algorithm to efficiently model spatial phenomena with Gaussian processes has been proposed [7], but the algorithm is used to model spatially varying quantities and is not applicable to classification problems. So far we are unaware of any active learning studies which take spatially dependent label acquisition costs into account.

The traveling salesman problem (TSP) is a widely studied problem where, given a set of cities, the goal is to find the shortest path between the cities such that the salesman visits each city. In this paper, we make use of solutions to the traveling salesman problem as well as a variant of the traveling salesman problem called the traveling salesman problem with profits (TSPP) [2], in which a profit is assigned to each city. In TSPP, while the salesman is required to start and end at the same location, the salesman is not required to visit every city. While the exact goal of the salesman varies depending on what variant of TSPP is being solved, the general goal is that the salesman is attempting to maximize profit while minimizing distance traveled under some additional constraint, such as a minimum level of total profit that needs to be achieved or some bound on the total distance the salesman can travel. Both the TSP and TSPP have been widely studied, and many heuristics have been devised (see [2] for a recent survey).

3 Background

In this section, we discuss uncertainty sampling and heuristics for solving the traveling salesman problem and traveling salesman problem with profits in more detail.

3.1 Active learning. In active learning, there is a pool of unlabeled points $\mathcal{U}$ and a pool of labeled points $\mathcal{L}$. The goal is to iteratively pick the most informative points in $\mathcal{U}$ for labeling, obtain the labels from some oracle or teacher, add the labeled points to $\mathcal{L}$, retrain the classifier on $\mathcal{L}$, and then repeat the process until either $\mathcal{U}$ is an empty set or until the oracle is no longer able to provide labels. During each iteration of active learning, n points are selected for labeling. We will refer to this as the batch size. One of the main differences in active learners is how one determines whether a point in $\mathcal{U}$ will be informative if labeled.

In this paper, we use an uncertainty sampling approach [11] to perform active learning. Uncertainty sampling works by assigning an uncertainty score to each point in $\mathcal{U}$ and picking the n points with the highest uncertainty scores. These uncertainty scores are based on the predictions of the classifier currently trained on $\mathcal{L}$.

In our experiments, we use two different classifiers: linear discriminant analysis (LDA) and support vector machines (SVMs). We use a different form of uncertainty score with each classifier to show the flexibility of our approach.

LDA is capable of returning posterior probabilities for unlabeled data. Thus, on each iteration of active learning, LDA is trained on $\mathcal{L}$, and posterior probabilities are produced for the unlabeled training data in $\mathcal{U}$. The uncertainty scores are then based on these posterior probabilities. When using LDA, we set the uncertainty score to be inversely proportional to the size of the margin between the highest and second highest posterior probabilities. Let $j' = argmax_j P(y_j|\mathbf{x_i})$ and $j'' = argmax_{j \neq j'} P(y_j|\mathbf{x_i})$. Then,

$$u(i) = \frac{1}{P(y_{j'}|\mathbf{x_i}) - P(y_{j''}|\mathbf{x_i})} \tag{3.1}$$

For SVMs, we use an uncertainty score inversely proportional to the distance to the separating hyperplane [18]. That is, the closer a point is to the hyperplane, the higher its uncertainty score.

3.2 Traveling salesman problem. The traveling salesman problem has been widely studied [10], and many solutions have been proposed. We will use the following terminology and notation. Each city that the salesman travels to is denoted as a node $v(i)$, where $v(i)$ is the ith city. In addition, the salesman must start and end at some location. We will refer to this as the "home" location and denote it as $v(0)$. Finally, the distance between two nodes i and j is denoted by $d(i, j)$, where $d(0, j)$ would represent the distance from the "home" location to the jth point.

We use the following heuristic, chosen because the solution is quite fast and also because of readily available code[2]. Intuitively, our heuristic for solving the

[2]Our experiments were run using a modified version of the

traveling salesman problem works as follows. Given n points (one of which is the home location of the salesman), the algorithm begins by randomly initializing a path through the n points. The algorithm then proceeds to try to improve the path by repeatedly using 2-opt and 2.5-opt, where 2-opt refers to exchanging the position of two nodes in the current path, and 2.5-opt refers to removing a single node from the current path and inserting it between two existing nodes in the path [6].

More formally, the algorithm is as follows:

1. Initialization: randomly initialize a path through the n points
2. Iterate until number of max iterations is reached
   (a) Randomly apply 2-opt; if path length is shortened, apply change; otherwise, keep the current path
   (b) Randomly apply 2.5-opt; if path length is shortened, apply change; otherwise, keep the current path

One problem with this heuristic is determining when to terminate the algorithm. We found through preliminary experiments that setting the max number of iterations to 10,000 was quite fast for a moderate number of nodes (i.e., less than 100) and resulted in convergence to a good solution.

3.3 Traveling salesman with profits. We use a variant of the H2 heuristic originally presented in [3] to solve the traveling salesman with profits problem. Our modified algorithm is as follows:

1. Initialization: initialize path to consist of $v(0)$ (i.e., the home node) and the point j that minimizes $\frac{d(0,j)+d(j,0)}{p(j)}$ where $p(j)$ is the profit obtained for visiting node j
2. Iterate until the number of nodes (not counting $v(0)$) in the path is equal to n, the number of points desired in the path: Add point that minimizes $\frac{d(i,j)+d(j,k)-d(i,k)}{p(j)}$ for some point $v(j)$ not in current path and consecutive points $v(i)$ and $v(k)$ in path
3. Optimize path using solution to traveling salesman problem described in section 3.2

In the original H2 algorithm, the algorithm terminates when a certain traveling budget (e.g., distance traveled) is expended. In our modification, the algorithm terminates after a set number of points have been added to the path. In addition, we found that after adding the n chosen points, optimizing the path using the heuristic presented in section 3.2 further reduced the overall path length.

As with our method for solving the traveling salesman problem, the H2 algorithm was chosen since it was simple to implement and because it was fast enough for experimentation. In addition, the H2 algorithm was simple to modify to stop once it had added a set number of nodes to the path.

Our proposed framework makes no restrictions on which TSPP algorithm needs to be used. Thus, better solutions can be easily incorporated.

4 Active learning for spatial data

Active learning on spatial data, as formulated in this paper, proceeds in an iterative fashion, where, on each iteration, the algorithm chooses n points to label, provides the labeler a path to travel between the points, and the labeler travels along the path provided by the algorithm, beginning and ending at the same "home location" on each iteration. Once the labels have been obtained, the algorithm retrains on the set of labeled points $\mathcal{L}$, and the process repeats. The goal is to produce a classifier that reduces empirical errors while minimizing data acquisition costs.

Note that the "distance" between two points may not refer to the geographic distance, but may reflect characteristics such as the terrain between two points. For example, traveling from a point on land to a point on water may be costly regardless of the geographic distance. Thus, we use the term "distance" to denote cost, e.g., time required to travel between points, gas expended for traveling between points, kilometers traveled between points, or some other measure. In addition, the approaches we propose do not require symmetric distance matrices. For example, traveling from a higher to lower elevation may require less effort than traveling from a lower to a higher elevation.

In the remainder of this section, we discuss three approaches for solving this problem, where the third approach is a generalization of the first two.

4.1 Combining active learning with the traveling salesman problem. A simple but somewhat naive approach for incorporating spatially related label acquisition costs with active learning is to do the following:

1. On each iteration, based on active learning criteria, select n points to label
2. Use a solution to the traveling salesman problem

code used in the Matlab demonstration of the traveling salesman problem (travel.m)

to find the shortest path from home through all of the n selected points

That is, if one were using uncertainty as the active learning criteria, the naive method would pick n points via uncertainty sampling and then find an optimal path through those points. We will refer to this approach as "US/TSP".

The reason we call this method naive is that one ignores all spatial information when selecting the initial n points. These n points with highest uncertainty could all be very far from each other, and it may be less costly to select n points with lower uncertainty scores if they are closer to each other. We will see empirically in section 6 that this method is typically outperformed by the more sophisticated (but still simple to implement) approach described next.

4.2 Combining active learning with traveling salesman with profits. As mentioned earlier, in the traveling salesman problem with profits, each city that the salesman can potentially travel to has a profit associated with visiting that city. The goal is to find a short path that maximizes profit without exceeding the salesman's traveling budget (in this case, the budget is the number of visited nodes).

Our approach for performing active learning on spatial data is to transform the problem into a traveling salesman problem with profits. On each iteration of active learning, the cities that can potentially be visited by the salesman (or labeler) are the points in the unlabeled set $\mathcal{U}$. The profit associated with visiting an unlabeled point is set equal to the uncertainty score of that particular unlabeled point as defined in section 3.1. That is, $p(j) = u(j)$, where $p(j)$ is the profit for visiting the jth unlabeled point and $u(j)$ is the uncertainty score of the jth unlabeled point.

Once profits have been assigned to each city, the n points selected by the algorithm used to solve the traveling salesman with profits problem are the points selected for the oracle to label. We refer to this approach as "US/TSPP".

4.3 A generalization of active learning with traveling salesman with profits. Empirically, we found a variant of the above solution to be useful: instead of supplying all possible unlabeled points to the traveling salesman with profits algorithm, only the top m points with the highest uncertainty scores (where $m \geq n$) are used. The advantages of this variation on our framework is that it is computationally more efficient and does well at trading off between maximizing classification performance and minimizing distance traveled. We refer to this approach as "US/TSPP (filtered)".

More formally, our approach is as follows:

1. Input: batch size n, number of points to consider m, labeled set $\mathcal{L}$, unlabeled set $\mathcal{U}$
2. Iterate
   (a) Assign uncertainty scores $\mathbf{u}$ to all points in $\mathcal{U}$
   (b) Select the $\min(m, |\mathcal{U}|)$ points with the highest uncertainty scores where $|\mathcal{U}|$ is the number of points in $\mathcal{U}$; let this set of unlabeled candidate points be denoted as $\mathcal{U}_C$
   (c) Set the profit for visiting the ith point in $\mathcal{U}_C$ to the uncertainty score of that point
   (d) Select n points from $\mathcal{U}_C$ and create a path through these n points using a solution to TSPP

Note that if $m = n$, then this approach reduces to US/TSP, while if m is equal to the number of points in $\mathcal{U}$, then this approach reduces to US/TSPP. That is, this approach generalizes the approaches described earlier.

5 Experiments

5.1 Classifiers. As mentioned, we use two classifiers in our experiments: LDA and SVMs. These classifiers were chosen for experimentation with spatial label acquisition costs because they performed well on "traditional" active learning with non-spatial label acquisition costs on hyperspectral data. For LDA, we use uncertainty sampling where the uncertainty scores are inversely proportional to the margin between the largest posterior probability and second largest posterior probability as predicted by LDA. For SVMs, the uncertainty score is inversely proportional to the absolute value of the distance from the hyperplane, where points closer to the hyperplane are more uncertain [18].

For SVMs, we found that a linear kernel worked well; moreover, there are fewer parameters to be tuned for a linear kernel. One parameter, however, is the trade-off between margin and empirical error on the training set. We found that setting this parameter equal to one seemed to work well. However, the problem of tuning parameters for SVMs during active learning is, to the best of our knowledge, still an open question. Typically, in classification, one would tune parameters using ten-fold cross validation or by using a single hold-out set for validation. This may not work well in active learning for two reasons: speed and lack of data. Because of the large number of iterations required in active learning, retuning the parameters for each iteration may be unwieldy, particularly if

one were to use some form of cross-validation. In addition, because of the lack of data, particularly in early iterations of active learning, there is typically not enough data to reserve a separate hold-out set. In order to avoid confounding the results of the current study with any affects due to some form of dynamic parameter estimation, we use as few parameters as possible (i.e., we choose a linear kernel) and keep all other parameters constant (i.e., we set trade-off between margin and empirical error on the training set equal to one).

5.2 Datasets. Land cover classification by hyperspectral image (HSI) data analysis has become an important part of remote sensing research in recent years [9][5][13]. Compared to conventional multi-spectral images where each pixel usually contains tens of bands, pixels in hyperspectral images usually consist of more than a hundred spectral bands, providing fine-resolution spectral information. Classification techniques used for this application should be able to handle high-dimensional high-resolution data and a fairly high number of classes. Obtaining ground truth is another challenge, since HSI can cover very large areas but it is not usually possible to obtain highly accurate class labels for all locations in the image.

The proposed method was evaluated on hyperspectral images taken from two geographically different locations: NASA's John F. Kennedy Space Center(KSC) [14] and the Okavango Delta in Botswana [4]. We will call the two datasets the KSC and Botswana datasets, respectively. The goal of these datasets is to correctly classify types of land. Table 1 contains a list of the classes in the KSC and Botswana datasets, while Figure 1 shows images of Botswana and KSC with their corresponding class maps. The grey areas in the figure denote areas which have "no label." As can be seen in this figure, only a small fraction of the entire region actually has class labels.

The KSC dataset was acquired by NASA Airborne Visible/Infrared Imaging Spectrometer (AVIRIS) and originally consisted of 242 bands. After removing noisy bands, only the remaining 176 bands are used for classification. There are 13 different land cover types including water and mixed classes, which causes some difficulties in classification. The hyperspectral image used for experiments has 512×614 pixels with 18m spatial resolution.

The Botswana dataset was obtained from the Okavango Delta by the NASA EO-1 satellite with the Hyperion sensor on May 31, 2001. The acquired data originally consisted of 242 bands, but only 145 bands are used after preprocessing. The area used for experiments has 1476×256 pixels with 30m spatial resolution, with 14 different land cover classes.

In each dataset, each data point has a location as determined by two coordinates. We use the Euclidean distance between coordinates as the distance required to travel between two points.

We study both the full multiclass problem as well as several two-class problems. We used only LDA on the multiclass problem, and used both LDA and SVM on the two-class problems. The two-class problems were selected such that the following question could be answered: for a two-class problem where it is known that "traditional" active learning worked well, how much better does a technique that takes spatial information into account perform?

We also preprocess the data in two ways that are known to be effective for classifying hyperspectral data. First we utilized spatial information in addition to the spectral information, via the max-cut algorithm described in [1], where a pixel's feature vector is augmented with features from neighboring pixels whose spectral features are similar to the pixel of interest. The max-cut algorithm provides a way to identify pixels that are close both in physical and spectral spaces, and produces more accurate and stable classification results for spatial data.

Second, we applied best-basis feature reduction which exploits the high correlation between certain adjacent spectral bands, and is tailored for hyperspectral data analysis [8].

5.3 Experimental setup. We partition the data into training and test sets using five runs of ten-fold cross-validation and average the results. In particular, we use stratified sampling such that each fold has the same proportion of points from each class.

Once the training and test sets have been created, 10%[3] of the training data is randomly selected and labeled to form the initial labeled set $\mathcal{L}$, and the remaining unlabeled training data is placed in $\mathcal{U}$.

We use the best-basis feature reduction method to create a set of d' features. The value of d' was chosen such that $|\mathcal{L}| > d'$ since this is required by LDA to form a covariance matrix. Experimentally, this was done by setting d' to the smallest value in the set of $(40, 50, 60)$ that would satisfy the constraint $|\mathcal{L}| > d'$ if 10% of the training data were placed in $\mathcal{L}$[4].

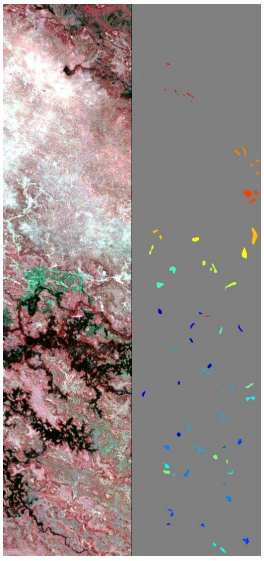
Once training and test sets have been selected, the experiment proceeds as follows. A point in the initially labeled training set $\mathcal{L}$ is randomly selected

[3] For multiclass datasets only, we used 5% of the data as seeds due to the larger size of the dataset

[4] Because of the larger dataset size in the multiclass experiments, it was possible to set d' equal to 120

Table 1: The class names in the KSC and Botswana datasets

KSC Dataset	
Class number	Class Name
1	scrub
2	willow swamp
3	cabbage palm hammock
4	cabbage palm/oak hammock
5	slash pine
6	oak/broadleaf hammock
7	hardwood swamp
8	graminoid marsh
9	spartina marsh
10	cattail marsh
11	salt marsh
12	mud flats
13	water
Botswana Dataset	
Class number	Class Name
1	water
2	hippo grass
3	floodplain grasses 1
4	floodplain grasses 2
5	reeds1
6	riparian
7	firescar2
8	island interior
9	acacia woodlands
10	acacia shrublands
11	acacia grasslands
12	short mopane
13	mixed mopane
14	exposed soils



(a) Botswana

(b) KSC

Figure 1: Botswana and KSC images with class maps

to be the "home" location.[5] Then, n points are selected and labeled in an iterative fashion using either the approaches described in section 4 or the following baseline algorithm: n points are picked randomly on each iteration and a path through the n selected points (and starting and ending at home) is found using the heuristic described in section 3.2. This baseline for spatial active learning is analogous to the baseline commonly used in standard, non-spatial active learning experiments where points are randomly selected; we will refer to this baseline as "random/TSP". We also test our approaches against a "closest next" baseline, which is a method that ignores uncertainty scores and is concerned primarily with picking nearby points.

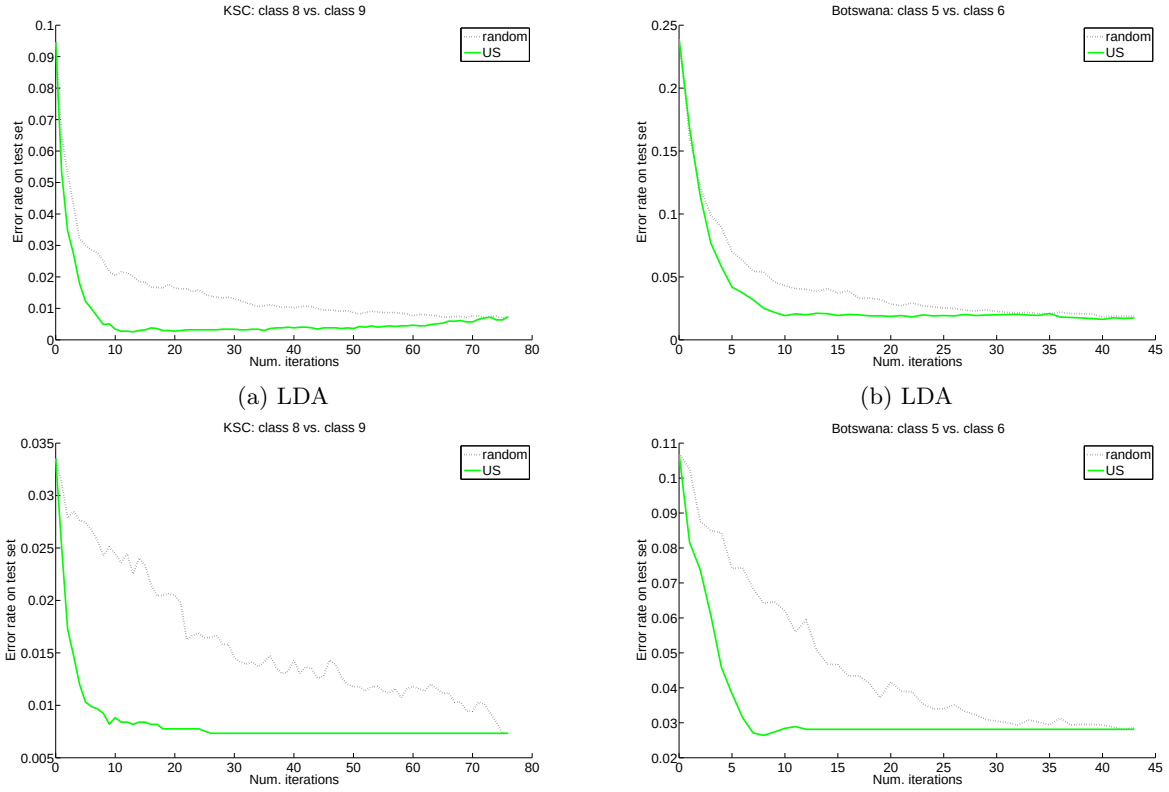
[5] We plan to run controlled experiments to determine whether results vary based on the position of the "home" location (e.g., results for a centrally located home versus a home located north of most of the data) in future work.

The "closest next" baseline works by traveling to and labeling the closest unlabeled point until n points have been labeled. In our experiments, we use $n = 10$ as the batch size. In addition, the approach in section 4.2 requires a parameter m to determine the number of points in $\mathcal{U}_C$, which we simply set to 100 in all experiments. Although we did not investigate this, additional tuning of m may be useful.

6 Results

6.1 Traditional active learning. For the sake of completeness, let us first show some results for a "traditional" active learning experiment similar to that presented in [15], where label acquisition costs for all points are equal and independent of all other points chosen for labeling. The purpose of looking at this case is to demonstrate that uncertainty sampling works well in the "traditional" active learning problem on hyperspectral data.

Sample results are plotted in Figure 2 in the form of a "banana curve," a traditional format for present-



(a) LDA (b) LDA

(c) SVM (d) SVM

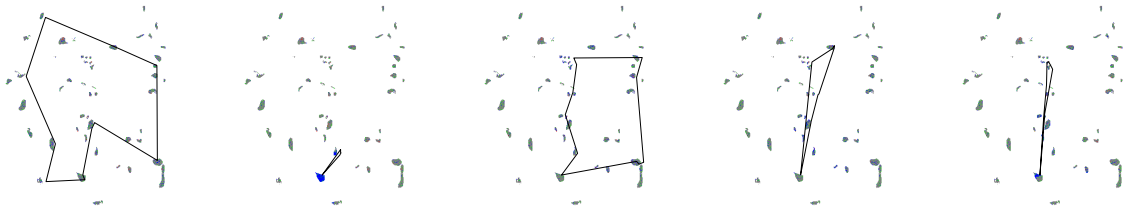
Figure 2: Example results when ignoring label acquisition costs

ing active learning results. Here, the horizontal axis corresponds to number of active learning iterations and the vertical axis corresponds to the error rate on the test set. Uncertainty sampling beats random sampling regardless of the dataset or classifier used.

6.2 Results for spatially cost-sensitive active learning. Now let us examine the case where label acquisition costs are proportional to the distance traveled to reach the set of points to be labeled. Results using SVMs are plotted in figure 4 while results for LDA are plotted in figure 5, 6 and 7. Specific values for certain points in the graphs for two-class problems are listed in table 2.

In these graphs, we have modified the traditional banana curve format in order to take into account different label acquisition costs. In this case, the horizontal axis is now proportional to the total distance traveled to label $\mathcal{L}$. That is, a point on the curve represents the distance the labeler needed to travel in order for the classifier to achieve a certain level of performance as measured by error rate on the test set. A number of interesting observations can be drawn from this type of figure. Unlike a traditional curve in active learning, the right-most point of each curve on each graph will be at a different place. This point corresponds to where all the points originally in $\mathcal{U}$ have been labeled (we have denoted this point with an "X" in the graphs for ease of visibility, and have also listed example values for this point in table 2). The fact that different curves end at different places indicates that each method requires different amounts of effort to label all of $\mathcal{U}$. Just comparing the right-most point is rather informative. Not surprisingly, the "closest next" baseline is best in minimizing total distance traveled in order to label $\mathcal{U}$, but US/TSPP is very competitive with the "closest next" baseline. Both "closest next" and US/TSPP outperform US/TSPP (filtered) which outperforms US/TSP which outperforms random sampling. Numerically, from the last column of table 2, the distance that one needs to travel to label all of $\mathcal{U}$ using US/TSPP is, on average, less than half the distance that one travels if using US/TSP. In addition, the additional distance traveled by labeling points using US/TSPP versus the "closest next" baseline when labeling all of $\mathcal{U}$ is relatively small.

For the sake of illustration, we have plotted five



(a) random/TSP (b) closest next (c) US/TSP (d) US/TSPP (filtered) (e) US/TSPP

Figure 3: Example paths in our experiments

example paths on the multiclass KSC dataset using LDA[6], with one path for each method, in figure 3. Here, the distance saved by US/TSPP and US/TSPP(filtered) over random/TSP and US/TSP is also apparent. While the "closest next" baseline has an extremely short path length in the figure, the resulting error rate is not competitive with the other methods, as we will discuss below.

Another question in active learning is how quickly a method reduces classification error. Looking at the graphs, one can qualitatively make the judgement that, for LDA, "closest next" and US/TSPP tends to reduce the error rate fastest, while for SVMs, there seems to be no clear winner in terms of which uncertainty sampling approach reduces error rate the fastest. Figure 7, which "zooms in" on the initial rounds of active learning when the labeler has yet to travel very far, gives a clearer picture of the relative speeds at which each method reduces error. Quantitatively, one can compare the error rate when the labeler has not yet traveled very far. In table 2, we have listed error rate over some example two-class datasets when the labeler has traveled 25, 50, 100, and 150 kilometers. The results in table 2 indicate that US/TSPP and US/TSPP(filtered) reduce error rate more quickly than US/TSP or random/TSP.

Finally, one can also compare algorithms based on the lowest error rate that approach is able to achieve, regardless of how much effort it requires to reach that point or how "quickly" an algorithm reduces error rate. When looking at lowest error rate, the "closest next" baseline fares very poorly. This is also apparent in Figure 7, where the error rate of the "closest next" baseline essentially plateaus much earlier than any other method. Of the methods that take uncertainty scores into account, for both SVMs and LDA, while US/TSP can sometimes achieve a lower error rate, the lowest error rate, regardless of labeling effort required to achieve it, is comparable for US/TSPP(filtered) and US/TSPP.

One may note that the lowest error rate achieved by a method may be much lower than the final error rate achieved by labeling all points in $\mathcal{U}$. That is, one may achieve a lower error rate by labeling only a subset of $\mathcal{U}$. The error rate may increase when additional points are added, for example, if noisy or outlying points are added to $\mathcal{L}$. A similar phenomenon can be seen in many of the curves for SVMs, where the curves seem to reach a low error rate and where there seems to be no improvement even after many more labeled points are placed in $\mathcal{L}$. This is most likely due to the fact that only support vectors, which are close to the hyperplane, are used to define the separating hyperplane. Adding points which are not support vectors will not affect the learned hyperplane. When to stop active learning is an open research question outside of the current scope of this paper [16].

In summary, the proposed methods that combine active learning with a solution to the traveling salesman with profits problem trade off between achieving a low error rate and reducing label acquisition costs better than the random baseline, the naive solution which combines active learning with the traveling salesman problem, and the "closest next" baseline which is only concerned with reducing label acquisition costs and not with choosing points useful for reducing error rate.

7 Future Work

There are several areas where the work presented in this paper could be extended. One area of future work is to try other active learning methods or other solutions to the traveling salesman problem in our framework. Our framework is quite general, so the main goal of this future work would be to benchmark possible approaches. In particular, the approaches we have used in this paper are quite basic, and more sophisticated approaches for solving the traveling salesman problem and traveling salesman problem with profits should

[6] we arbitrarily chose to plot the 25th iteration of active learning for each method

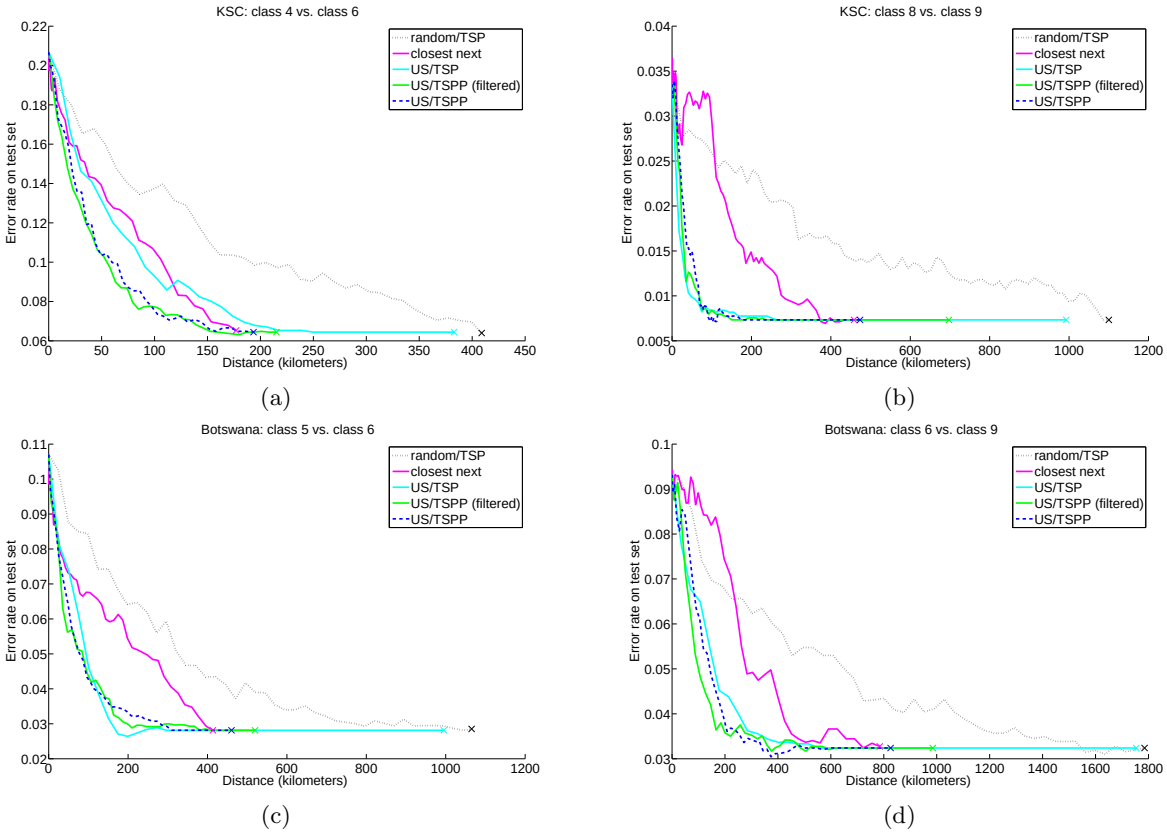


Figure 4: Results for SVMs while accounting for label acquisition costs

further improve our results. Another area of future work is to apply our approach on other spatial data, such as LIDAR data or data obtained for other GIS purposes.

A more extensive area of future work is to extend our framework to other scenarios. For example, one possible variation is that there are no restrictions on how many points can be labeled each iteration, but there is a restriction on the total distance traveled. Another variation is that the labeler is not required to return to some "home" location after each iteration. In this case, a good solution might take into account that it may be beneficial if the ending location for the current iteration is nearby promising points that the labeler could visit on the next iteration.

8 Conclusion

When performing active learning on spatially distributed data such as hyperspectral data or other GIS applications, there are variable label acquisition costs involved in labeling each point. These label acquisition costs may depend on the distance the labeler needs to travel in order to label each point selected by active learning. Standard active learning techniques are unable to handle such variable costs, and assume that the cost of labeling each point is uniform and that the cost of labeling each point is independent of all other points being labeled on a particular iteration of active learning. If the cost of labeling a point is proportional to the distance needed to travel to that point, and if the labeler travels to several points on each iteration of active learning, then both of these assumptions made by traditional active learning are broken.

In this paper, we not only introduce the problem of active learning on spatial data where label acquisition costs are proportional to the distance required to travel to that point, but also present a framework for improving classifier performance while minimizing the distance traveled by the labeler.

We hope that this paper will encourage other researchers in this area since improved solutions to this problem have the potential to save significant time, money, and effort in applications involving spatial data.

References

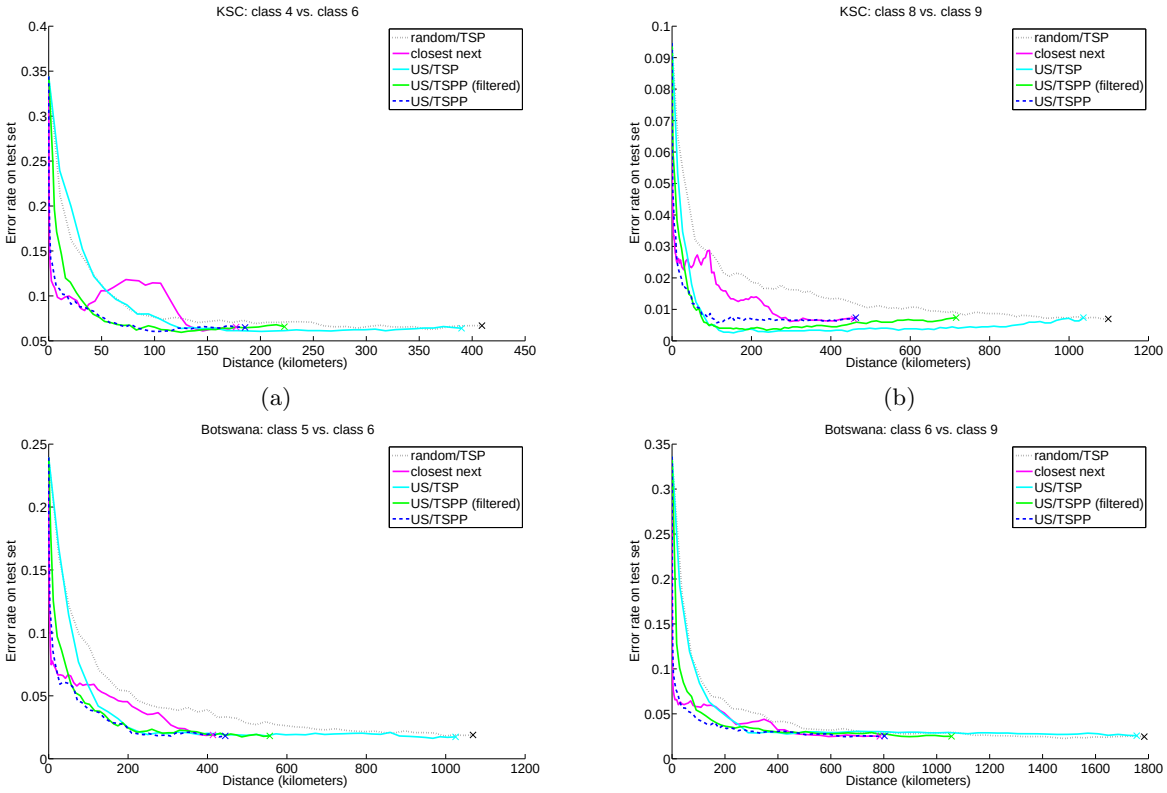


Figure 5: Results for LDA while accounting for label acquisition costs

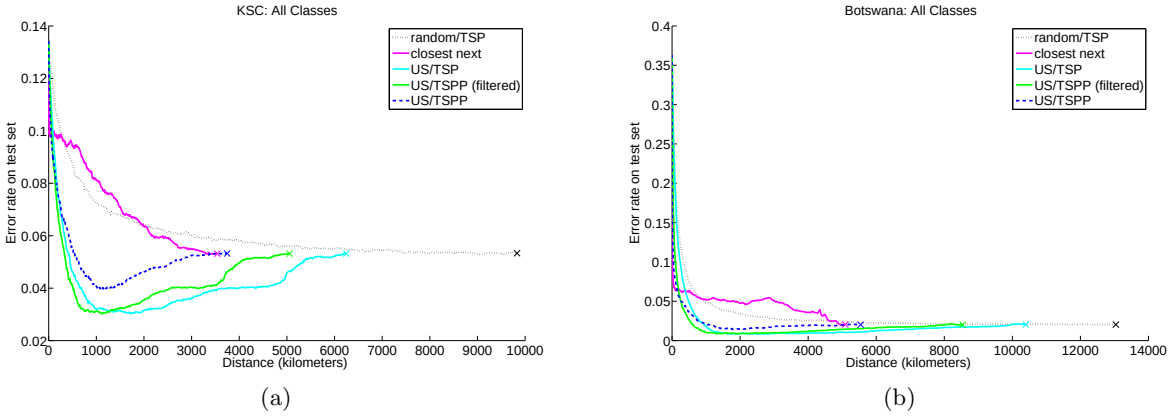
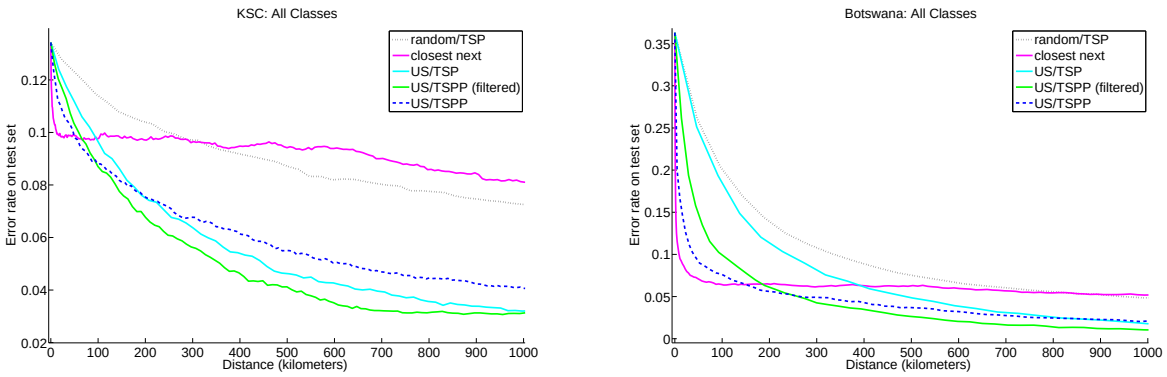


Figure 6: Results for LDA on multiclass problems while accounting for label acquisition costs



(a) (b)

Figure 7: Results for LDA on multiclass problems while accounting for label acquisition costs while distance traveled is less than 1000 km

[1] Y. Chen, M. Crawford, and J. Ghosh. Knowledge based stacking of hyperspectral data for land cover classification. *Computational Intelligence and Data Mining, 2007. CIDM 2007. IEEE Symposium on*, pages 316–322, 1 2007-April 5 2007.

[2] D. Feillet, P. Dejax, and M. Gendreau. Traveling salesman problems with profits. transportation science 39(2. *Transportation Science*, 39:188–205, 2005.

[3] M. Gendreau, G. Laporte, and F. Semet. A branch-and-cut algorithm for the undirected selective traveling salesman problem. *Networks*, 32:263–273, 1998.

[4] J. Ham, Y. Chen, M. M. Crawford, and J. Ghosh. Investigation of the random forest framework for classification of hyperspectral data. *IEEE Trans. Geosci. and Remote Sens.*, 43(3):492–501, 2005.

[5] L. Jimenez and D. Landgrebe. Supervised classification in high-dimensional space: geometrical, statistical, and asymptotical properties of multivariate data. *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, 28(1):39–54, Feb 1998.

[6] D. S. Johnson and L. A. McGeoch. The traveling salesman problem: A case study. In E. H. Aarts and J. K. Lenstra, editors, *Local search in combinatorial optimization*, pages 215–310. Wiley, 1997.

[7] A. Krause and C. Guestrin. Nonmyopic active learning of gaussian processes: an exploration-exploitation approach. In *ICML '07: Proceedings of the 24th international conference on Machine learning*, pages 449–456, New York, NY, USA, 2007. ACM.

[8] S. Kumar, J. Ghosh, and M. M. Crawford. Best-bases feature extraction algorithms for classification of hyperspectral data. *IEEE Trans. on Geosci. and Remote Sens.*, 39(7):1368–1379, 2001.

[9] D. Landgrebe. Hyperspectral image data analysis. *Signal Processing Magazine, IEEE*, 19(1):17–28, Jan 2002.

[10] E. L. Laweler, J. K. Lenstra, A. H. G. R. Kan, and D. B. Shmoys. *The Traveling Salesman Problem: A Guided Tour of Combinatorial Optimization*. Wiley, 1985.

[11] D. D. Lewis and W. A. Gale. A sequential algorithm for training text classifiers. In *Proceedings of the 17th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 3–12. Springer-Verlag New York, Inc., 1994.

[12] D. MacKay. Information-based objective functions for active data selection. *Neural Computation*, 4(4):590–604, 1992.

[13] F. Melgani and L. Bruzzone. Classification of hyperspectral remote sensing images with support vector machines. *Geoscience and Remote Sensing, IEEE Transactions on*, 42(8):1778–1790, Aug. 2004.

[14] J. T. Morgan. *Adaptive hierarchical classifier with limited training data*. PhD thesis, Univ. of Texas at Austin, 2002.

[15] S. Rajan, J. Ghosh, and M. M. Crawford. An active learning approach to hyperspectral data classification. *IEEE Trans. on Geosci. and Remote Sens.*, 46(4):1231–1242, 2008.

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

[17] H. S. Seung, M. Opper, and H. Sompolinsky. Query by committee. In *Proceedings of the Fifth Annual Workshop on Computational Learning Theory*, pages 287–294, Pittsburgh, PA, USA, 1992. ACM Press.

[18] S. Tong and D. Koller. Support vector machine active learning with applications to text classification. *J. Mach. Learn. Res.*, 2:45–66, 2002.

Table 2: Values for specific points on our graphs (2-class problems)

LDA results						
dataset	activeLearner	errorRateAt25KM	errorRateAt50KM	errorRateAt100KM	errorRateAt150KM	totalDist
ksc(4vs6)	random/TSP	0.156	0.113	0.076	0.071	409.226
ksc(4vs6)	closest next	0.107	0.119	0.080	0.063	176.994
ksc(4vs6)	US/TSP	0.184	0.111	0.079	0.063	389.946
ksc(4vs6)	US/TSPP(filt)	0.105	0.077	0.060	0.063	222.650
ksc(4vs6)	US/TSPP	0.093	0.075	0.062	0.066	185.487
ksc(8vs9)	random/TSP	0.056	0.038	0.027	0.021	1098.788
ksc(8vs9)	closest next	0.023	0.025	0.020	0.017	458.560
ksc(8vs9)	US/TSP	0.038	0.019	0.005	0.003	1036.086
ksc(8vs9)	US/TSPP(filt)	0.025	0.013	0.005	0.004	715.482
ksc(8vs9)	US/TSPP	0.020	0.012	0.008	0.006	462.972
bots(5vs6)	random/TSP	0.159	0.124	0.090	0.063	1068.804
bots(5vs6)	closest next	0.069	0.065	0.060	0.047	413.718
bots(5vs6)	US/TSP	0.165	0.116	0.057	0.038	1024.816
bots(5vs6)	US/TSPP(filt)	0.095	0.070	0.043	0.032	556.944
bots(5vs6)	US/TSPP	0.072	0.057	0.043	0.027	444.768
bots(6vs9)	random/TSP	0.183	0.160	0.096	0.070	1784.187
bots(6vs9)	closest next	0.067	0.068	0.056	0.047	785.045
bots(6vs9)	US/TSP	0.186	0.150	0.091	0.061	1755.511
bots(6vs9)	US/TSPP(filt)	0.115	0.076	0.055	0.044	1055.999
bots(6vs9)	US/TSPP	0.075	0.059	0.043	0.038	802.682
SVM results						
dataset	activeLearner	errorRateAt25KM	errorRateAt50KM	errorRateAt100KM	errorRateAt150KM	totalDist
ksc(4vs6)	random/TSP	0.177	0.161	0.139	0.109	408.981
ksc(4vs6)	closest next	0.167	0.138	0.089	0.064	176.994
ksc(4vs6)	US/TSP	0.158	0.128	0.094	0.080	382.925
ksc(4vs6)	US/TSPP(filt)	0.133	0.102	0.074	0.062	215.088
ksc(4vs6)	US/TSPP	0.144	0.105	0.075	0.065	193.434
ksc(8vs9)	random/TSP	0.029	0.028	0.026	0.024	1100.130
ksc(8vs9)	closest next	0.025	0.028	0.027	0.020	458.560
ksc(8vs9)	US/TSP	0.015	0.008	0.009	0.008	992.885
ksc(8vs9)	US/TSPP(filt)	0.017	0.011	0.009	0.007	697.326
ksc(8vs9)	US/TSPP	0.021	0.015	0.010	0.009	473.571
bots(5vs6)	random/TSP	0.101	0.087	0.085	0.073	1065.287
bots(5vs6)	closest next	0.078	0.074	0.072	0.064	413.718
bots(5vs6)	US/TSP	0.081	0.074	0.049	0.032	995.193
bots(5vs6)	US/TSPP(filt)	0.080	0.056	0.044	0.037	519.772
bots(5vs6)	US/TSPP	0.076	0.061	0.045	0.035	460.466
bots(6vs9)	random/TSP	0.091	0.089	0.077	0.070	1785.266
bots(6vs9)	closest next	0.095	0.092	0.085	0.070	785.045
bots(6vs9)	US/TSP	0.078	0.072	0.066	0.052	1753.798
bots(6vs9)	US/TSPP(filt)	0.083	0.069	0.052	0.047	985.121
bots(6vs9)	US/TSPP	0.083	0.077	0.056	0.046	825.727