IPSJ SIG Technical Report

Vol.2014-DBS-159 No.4
Vol.2014-IFAT-115 No.4
2014/8/1

Active Learning Based on Geographical Orientation for Automatic Transportation Mode Estimation

Brendan Cowan[1] Yoshihiko Suhara[2] Hiroyuki Toda[2] Yoshimasa Koike[2]

Abstract: We focus on the automatic transportation estimation task, which automatically estimates transportation modes given GPS trajectories of a user. Previous works have used supervised learning frameworks to estimate transportation modes and have reported that it achieves certain performances. However, the main drawback of supervised learning is the requirement of a significant amount of labeled data. Active learning is an effective solution to this problem. Although many studies have developed a wide variety of active learning algorithms, it has previously been unclear as to whether active learning works well for the automatic transportation mode estimation task. In addition, no previous work reveals which aspects are useful for selecting better instances in terms of model improvement for this task. We propose a novel aspect, geographical orientation, to develop a semi-stream-based active learning method. Our method takes into account geographical distance and density separately from the information based solely on feature space.

1. Introduction

Due to the rapid increase of GPS-embedded equipment such as smartphones and tablets, mining a user's context from his/her raw GPS trajectories is an important task in recording a user's lifelog (*e.g.*, Moves[*1]). Moves is a smartphone application which automatically estimates a user's stay area and transportation modes from his/her raw GPS trajectories. Automatic assignment of stay areas and/or transportation modes to a user's raw GPS trajectories can be regarded as assigning semantic labels to raw information. That semantic information is valuable in terms of not only lifelog, but also additional information for personalized applications.

We show the automatic transportation mode assignment task in Figure 1. Automatic transportation mode assignment can be divided into two tasks [1][2]: (1) GPS trajectory segmentation, and (2) automatic transportation mode estimation. GPS trajectory segmentation is the task of splitting raw GPS trajectories into multiple segments. Automatic transportation mode estimation is the task of estimating the transportation modes of given segments.

Supervised learning is used to build a multi-class classifier to estimate the transportation modes for the task. Zheng et al. [1][2] conducted experiments on real users' datasets and reported that supervised learning achieves a constant accuracy on transportation mode inference. We deemed the performance of the GPS trajectory segmentation method presented by Zheng et al. [1][2] to be sufficient for our purposes. Thus, in this paper, we focus on the automatic transportation mode estimation task.

The main drawback of the supervised learning approach is the necessity for a significant amount of labeled data. Preparing enough annotations is difficult due to the high annotation cost.

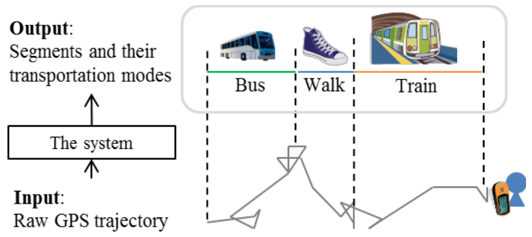

Fig. 1 The automatic transportation mode assignment task we tackle in this paper.

This fact keeps the supervised machine learning approach from achieving a constant performance. If we are able to select appropriate instances to be annotated, the classifier built with sophisticated training data will achieve a higher performance than the classifier built with randomly selected training data.

This situation leads us to active learning. Active learning is an important strategy in the case where supervised learning algorithms perform well, but there are not enough labeled data available. Fundamentally, the core goal of active learning is to iteratively select the instance that will improve the current model the most when the instance is annotated by human assessors (often called *oracles*). Every time a new instance is labeled by the oracle, it will be inserted into the training dataset, which the active learning classifier will then be retrained on.

Although semi-supervised learning is an alternative solution to a lack of labeled data, we focus on the active learning approach in this paper for following two reasons: (1) Adding labeled data is necessary when the size of the initial training data is very small. (2) Active learning can be combined with semi-supervised learning. Thus, we assume that applying semi-supervised learning is out of the scope of this paper.

The different approaches to active learning vary in terms of instance selection methods. One basic strategy is the uncertainty sampling approach, which selects the target instance based on

[1] University of Alberta
[2] NTT Service Evolution Laboratories, NTT Corporation
[*1] `https://play.google.com/store/apps/details?id=com.protogeo.moves`

the uncertainty score of the instances calculated by the current model. Density-weighted methods [3] are more sophisticated and incorporate density information of the instance with other basic information (*e.g.*, uncertainty score). One representative density-weighted method is the information density algorithm [4]. The information density algorithm calculates the density score of an instance by taking the average similarity between the instance and other instances, as calculated by some similarity function. This algorithm can avoid choosing outlier instances in the feature space because outlier instances have low scores in terms of density. While these algorithms performed nearly adequately in other tasks, even the slightest of improvements in accuracy would have a strong impact on our task. This is especially true during the initial iterations of active learning because the cost of annotation in our task is very high.

We consider the idea that the geographical orientations of instances have essential information that can improve inference models in geo-spatial tasks such as the transportation mode assignment task tackled in this paper. Intuitively, instances in areas that have not yet been explored by the training data will have much information for improving the current model. On the other hand, instances which are near other instances that are already in the training set will potentially contain redundant information. This leads us to propose two novel aspects for active learning: *geographical uniqueness* and *geographical representativeness*. In this paper, we use *geographical orientation* as an inclusive term for the combination of these two concepts.

The geographical orientation algorithm consists of three parts: (1) base information, (2) geographical distance information, and (3) geographical density information. Conventional pool-based or stream based sampling algorithms such as density-weighted methods can be used to obtain the (1) base information. Our method uses distance/density information in geographical space instead of feature space. The novelty of our method is using geographical space separately from feature space to calculate selection scores for instances in the pool.

Through the experiments, we verify the effectiveness of our method, and thus confirm that our method captures the intuitive aspects of the task. We use our method to train a classifier for determining the transportation modes of GPS trajectories. A previously existing baseline method is then trained in the same way. Through comparison of the performances achieved, we conclude that our method has substantial benefits. We also conduct an experiment comparing different options for parameter selection of the method, from which we determine that our method is very flexible in terms of function selection.

Our contributions in this paper include:

- The development of a novel active learning algorithm that incorporates geographical orientation separately from feature space density information.
- The confirmation that geographical orientation aspects are effective indicators of useful instances to label in terms of accuracy for an active learning framework in the transportation mode assignment task.

The rest of the paper is organized as follows: We present the system overview and proposed method in Section 2. We report on the experiments conducted and discuss their results in Section 3, and review related work in Section 4. Finally, we conclude the paper in Section 5.

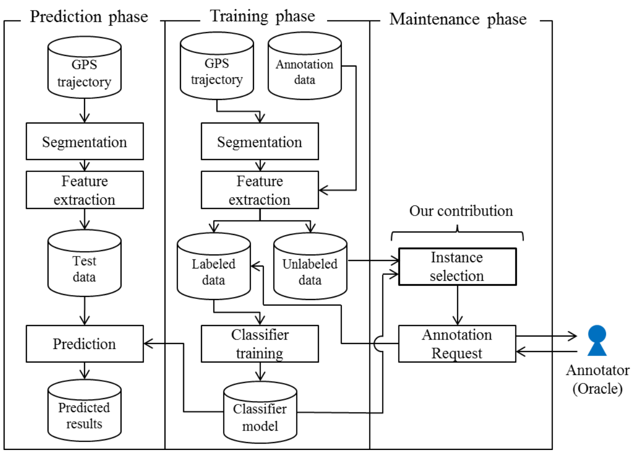

Fig. 2 System overview

2. Automatic Transportation Mode Assignment

2.1 System overview

An overview of the system is drawn in Figure 2. The system consists of three parts: (1) training phase, (2) prediction phase, and (3) model maintenance phase. Each phase is described below.

Training phase. Given GPS trajectories and (partial) transportation mode annotations of the GPS trajectories as input, the system splits the GPS trajectories into segments. In this step, either splitting by a constant time range or splitting with a change-point based algorithm [1][2] are valid procedures for segmentation. Note that we do not assume any specific algorithm for the segmentation method. Next, the system extracts features for each segment. After feature extraction, we can regard each segment as a separate instance. We then annotate segments with their respective transportation modes in order to create labeled instances. The segments that are not assigned labels are stored as unlabeled data. Those unlabeled data will be used in the model maintenance phase for applying active learning. Finally, the system uses a supervised learning algorithm to build a multi-class classifier.

Prediction phase. In the prediction phase, raw GPS trajectory inputs are split into segments, and then their features are extracted in the same manner as in the training phase. After that, the classifier predicts the transportation mode for each segment.

Model Maintenance phase. In model maintenance phase, the system selects an instance to be annotated. The instance is chosen via active learning from candidates in the unlabeled data created in the training phase. The selected instance is displayed to an oracle to be annotated. Then, the annotated instance is removed from the unlabeled data set and added to labeled data set. The current classifier is then retrained using the updated labeled data. This step is repeated until some stopping condition is met (such as a certain percentage of the unlabeled instances becoming labeled).

The instance selection module is the target of this paper, as well as focus of our contribution. Although, in reality, a user can only annotate his/her own GPS trajectories, we assume that the oracle can annotate any unlabeled data instance in this paper. This problem is out of the scope of the paper.

2.2 Active Learning for Model Maintenance

In a real life scenario for automatic transportation mode estimation, an oracle would label GPS trajectories from their trajectory history. However, a user is likely to only accurately remember the trajectories from their recent history, so the system would only ask for the label of a trajectory if it is contained within a small pool of most recent trajectories. In this paper, we define the semi-streamed sampling scenario, which assumes that a small set of labeled data and small pool (*i.e.*, unlabeled data) are available. The number of points in each subset given is the *pool-size-per-iteration*. In the semi-streamed-based sampling scenario, the general task of active learning algorithms is summarized as: calculating a selection score for each instance in each of the pools. Because the semi-streamed sampling scenario involves selecting points from a pool, all pool-based sampling methods can be applied to it. We deemed this to be a more accurate representation of reality than an entirely pool-based scenario, while still having the properties of, and adhering, to the framework of a pool-based approach. Selecting from only a small subset of points simulates the algorithm asking the user to label a point from their recent history.

Different sampling methods differ in the definition of uncertainty measure. Density-weighted methods incorporate density information to calculate the selection score. The density-weighted methods are usually used together with other sampling methods. We describe a density-weighted algorithm called the information density algorithm and then present our proposed algorithm.

2.2.1 Information Density Algorithm

Density weighted methods [3][4] are pool-based sampling frameworks which take into account the concept of representativeness. *Representativeness* is a measure of how well an unlabeled instance helps describe other instances in the dataset. For example, an instance that is densely surrounded by other data would have high representativeness, as these other instances can be accurately described by it. An instance in a sparsely populated area, on the other hand, would have low representativeness, as labeling this data point would tell us little about the other instances in the pool.

One popular density weighted algorithm is the *information density algorithm*, which can be generally written as:

$$x^* = \underset{x}{\operatorname{argmax}}\, \phi_{Info}(x) \times \phi_{Density}(x)^{\beta}.$$

Here, $\phi_{Info}(x)$ represents the informativeness of x according to some base strategy (*e.g.*, uncertain sampling, query-by-committee etc.). We refer to $\phi_{Info}(x)$ as the *informativeness function* in this paper. $\phi_{Density}(x)$ is a function that returns the representativeness of x according to the density information. We refer to $\phi_{Density}(x)$ as the *representativeness function* in this paper:

$$\phi_{Density}(x) = \left(\frac{1}{U} \sum_{u=1}^{U} \mathrm{sim}(x, x^{(u)}) \right),$$

where U is the number of training samples in x. β is a tuning parameter for balancing $\phi_{Base}(x)$ and $\phi_{Density}(x)$. Settles and Craven [4] use cosine similarity as the representativeness function:

$$\mathrm{sim}_{cos}(u, v) = \frac{\langle u, v \rangle}{\| u \|_2 \| v \|_2}, \qquad (1)$$

where $\langle \cdot, \cdot \rangle$ denotes the inner product.

2.3 Geographical Orientation Algorithm

We propose the *geographical orientation algorithm* in this paper. We consider the idea that the geographical information of an instance contains some information useful for improving the model. Intuitively, GPS trajectory data in areas that have not been explored by the training data have less redundant information than data points close to instances currently in the training set. This is the concept of *geographical uniqueness*, which can be expanded upon when we consider that there is another aspect for choosing instances: especially in geo-spatial tasks. For example, when the training data contains no labeled data in certain areas (we refer to these as *sparse areas*), the instances in the sparse area should improve the current model more than instances which are not in the sparse area. We combine these intuitions into one method.

Our method is based on the assumption that a unique instance in the geographical space has essential information in terms of model improvement. We note that the application of information density algorithms with geographical features (*e.g.*, longitude, latitude) cannot capture these characteristics because information density algorithms assign high scores on agglomerated instances in the feature space. Geographical features contain information which is significantly different from the rest of the feature set, and so we want to treat them differently, and train them separately. Thus, we separately calculate the geographical uniqueness score with the geographical distance function, which assigns scores on unlabeled instances based on the geographical distance from labeled data.

We draw this intuition in Figure 3. In this figure, we separately prepare the feature space (below) and the geographical space (above). When instances represent some geographical positions (*e.g.*, longitude and latitude values of the starting point of a segment), we can deploy the instances in the geographical space. There are three unlabeled instances around the top left area in the feature space. Those instances have very high selection scores if we use the information density algorithm because they are near the separating hyperplane (*i.e.*, a high informativeness score) and they are aggregated (*i.e.*, a high representativeness score). When switching the viewpoint to the geographical space, we find that those instances are surrounded by the labeled instances. This means that those instances have low geographical uniqueness in this situation. On the other hand, the instances around the bottom right in the feature space are located in separate areas in the geographical space. The instance has high informativeness and representativeness scores, and high geographical uniqueness score simultaneously. In this example, our method selects this instance as a query for the oracle in the model maintenance phase.

Geographical uniqueness differs from information density. The information density algorithm assumes that instances in high-density areas have more information. Instead, geographical uniqueness captures the intuition that data in rare areas have more information.

We call this approach the *geographical orientation algorithm* in this paper. The geographical orientation algorithm consists of three factors: (1) base factor, (2) geographical distance factor, and

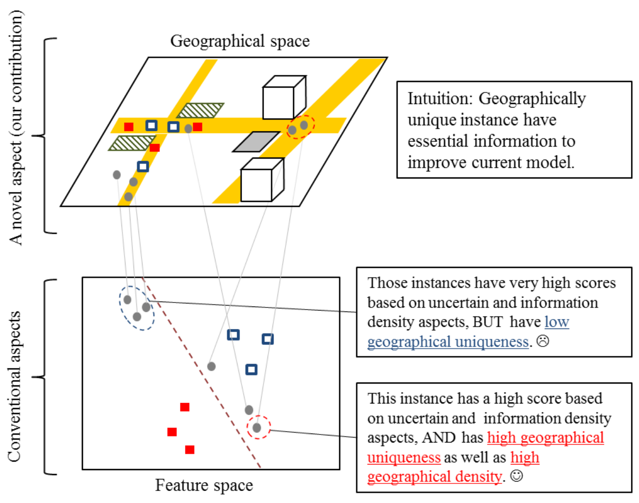
A novel aspect (our contribution)
Geographical space
Intuition: Geographically unique instance have essential information to improve current model.
Conventional aspects
Those instances have very high scores based on uncertain and information density aspects, BUT have low geographical uniqueness. ☹
This instance has a high score based on uncertain and information density aspects, AND has high geographical uniqueness as well as high geographical density. ☺
Feature space

Fig. 3 Intuitive description of our method.

(3) geographical density factor.

The geographical orientation algorithm can be written as follows:

$$x^* = \underset{x}{\operatorname{argmax}}\, \phi_{Base}(x) \times \phi_{GeoDist}(x)^{\gamma} \times \phi_{GeoDens}(x)^{\delta}.$$

Here, $\phi_{Base}(x)$ is the function for calculating the base factor. The base factor should return a measure of the points estimated "usefulness", where a high value corresponds to a more useful instance. It should be calculated in a given feature set, and conventional methods including the information density algorithm can be applied. For example, a function that returns a high value for points which have a low classification certainty can be used in order to combine geographical orientation with least confident sampling. As for new factors, $\phi_{GeoDist}(x)$ and $\phi_{GeoDens}(x)$ capture the geographical orientation in unlabeled data.

Geographical distance factor. The geographical distance factor represents the geographical uniqueness directly. The factor takes a higher score if the instance is not surrounded by other instances in the training data. We note that our method does not define the calculation method of $\phi_{GeoDist}(x)$. Several functions can be used as $\phi_{GeoDist}(x)$. In this paper, we use the minimum distance from the training data as the geographical distance factor:

$$\phi_{GeoDist}(x) = \min_{x \in D} \| x^{(i)} - x \|_2,$$

where D denotes the set of instances in the training data, and $x^{(i)}$ denotes the i-th candidate instance in the instance selection step, with only features *latitude* and *longitude*.

Geographical density factor. The geographical density factor works for avoiding selecting outliers in the geographical space as annotations. This factor can be regarded as the density calculation part of the information density algorithm. We note that the geographical density factor differs from the information density algorithm because the geographical density factor uses the density information in the geographical space separately from the feature space. The calculation is, however, done in a similar manner to that of the geographical density factor:

$$\phi_{GeoDens}(x) = \left(\frac{1}{U} \sum_{u=1}^{U} \mathrm{geosim}(x, x^{(u)}) \right),$$

where U is the number of training samples in x. Note here that the geographical orientation algorithm does not define a specific geosim function; one of many possible functions can be used, such as cosine similarity Eq. (1), Euclidean distance $\mathrm{geosim}_{Euc}(u, v) = \| u - v \|_2$, or Gaussian similarity $\mathrm{geosim}_G(u, v) = \exp\left(\| u - v \|_2^2 / \sigma^2\right)$, where σ represents the variance in the Gaussian distribution.

3. Evaluation

We conducted the experiments with a real-world dataset for the automatic transportation mode assignment task to verify the effectiveness of out method.

3.1 Dataset

In this paper, we used the GeoLife dataset[*2], which contains 182 users' GPS trajectories with partial transportation annotations. Each user has his/her track point recorded every 1-3 second(s), which consists of longitude, latitude, and a timestamp. Some users annotate their GPS trajectories with their transportation modes (*e.g.*, walk, car, bus etc.). In this experiment, we only consider seven transportation modes out of the eleven given transportation modes (walk, bike, bus, car, taxi, subway, train) because the number of labeled instances of the other transportation modes is too small to be accurately evaluated. The majority of the points also lie between the latitudes [5.49, 54.68], and the longitudes [89.44, 140.29]. For the simplicity of our calculations, we opted to ignore any data points that lied outside of this range. We also consider that the annotation set of a user who makes few annotations has annotation bias. That is, if a person who usually drives a car to work assigns the car label only a few times, the annotation result does not reflect the real user activity. We assume that if a user assigns many labels to their dataset, this is indicative of their dataset being accurately annotated. Thus, we remove the data of users who have a small number of annotations: specifically less than ten annotations. From this reduced dataset, we parsed the GPS points into trajectories using the methods described in Zheng et al. [1]. As a result, we have the dataset of 54 users, and 45,668 data instances. We calculate the features described in [1] (segment distance, maximum/average velocity, maximum acceleration, heading change rate, stop rate, etc.) for each of the segments[*3].

Evaluation Measure. We used accuracy learning curves as the performance evaluation metric. To globally compare different methods, we also used a *deficiency metric* [5] that has been widely used in active learning studies [6][7]. A deficiency metric shows the overall performance of an active learning method by returning a numerical value corresponding to the performance over all active learning iterations. The value given is equal to the ratio between the areas above the two learning curves being compared. Thus, this measure is always non-negative and a lower value implies that the learner preforms better compared to the algorithm it is being compared to, and vice versa. A value of 1 implies that the two methods preform equally as well. In these experiments, we show the deficiency scores for multiple steps of the algorithm,

[*2] `http://research.microsoft.com/en-us/downloads/b16d359d-d164-469e-9fd4-daa38f2b2e13/`

[*3] We used the same threshold values used by [2] to calculate heading change rate, stop rate, and velocity change rate

corresponding to different training set sizes.

Because each segment consists of multiple labeled GPS points, deciding the true label of each segment, as well as deciding exactly how to measure accuracy, can be difficult. In our experiments, the label that each segment consists of the most is what we assign to the ground truth for each segment. For example, if 90% of the points in a segment are labeled as "walk", and 10% are labeled as "bike", we would assign "walk" to be the ground truth of the segment. Any segments whose most common mode takes up less than 75% of the segment are ignored. Different methods for measuring accuracy are also available. In our experiments, we weigh each segment based on its distance feature [2].

$$Acc = \frac{\sum_{j=1}^{m} \text{CorrectSegment}[j].\text{Distance}}{\sum_{i=1}^{N} \text{Segment}[i].\text{Distance}},$$

where N is the total number of segments, and m is the number of correctly predicted segments.

3.2 Experiment 1: Comparison of Distance Functions

We compared three different functions for geosim(): (1) cosine similarity, (2) Euclidean distance, and (3) Gaussian similarity.

Settings. As a classifier for this experiment, we used a Decision Tree classifier, which was determined to be the best performing classifier in Zheng et al. [1][2] . We used the scikit-learn library[*4] for the implementation of the Decision Tree classifier. As input, each parsed GPS segment is treated as an instance, using the features described in section 3.1. 80% of the data is used as training data, and the remaining 20% is used as test data. The trajectories are sorted such that segments of a given user will only appear in either the training data or test data in order to avoid bias towards any of the users. Cross validation is performed on the data in 5 iterations: a different 20% of the data is used every time, and the final result is taken as the average of the passes. We repeat this 5-fold cross-validation process 5 times with different random seeds, resulting in total of 25 iterations.

We used a pool-size-per-iteration of 50. We used a Decision Tree as the base classifier. The function for calculating the base factor of the geographical orientation algorithm was an information density algorithm with least confidence sampling as a base. For β, γ, δ, a value of 1 was selected for each. For Gaussian similarity, σ was set to 1.

Results. We show the results in Figure 4. The x-axis represents the number of training data that have been selected and labeled by active learning. The curves shown represent the average of the 5-fold cross-validation iterations, and the error bars represent on standard deviation on either side. The graph shows that the results of using different similarity functions are more or less the same. This implies flexibility of the algorithm: the results will not change much for the system if any of the above methods are used.

3.3 Experiment 2: Comparison of Active Learning Methods

We conducted an experiment to compare the different active learning methods. We prepared the information density algorithm (ID) as a baseline method. We also prepared the geographical distance algorithm (GDI), geographical density algorithm (GDE),

*4 v.0.14 `http://scikit-learn.org/`

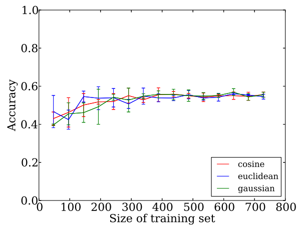

Fig. 4 Results of the distance functions from Experiment 1.

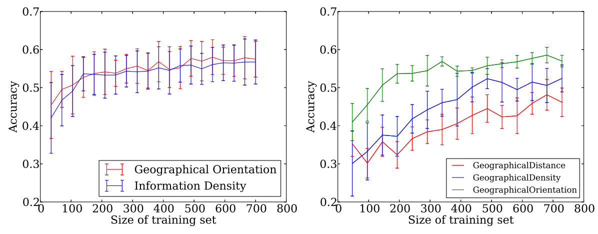

Fig. 5 Training accuracies from Experiment 2.

and geographical orientation algorithm (GO) as proposed methods.

Settings. We used a Euclidean similarity function as geosim() for GDI and GO. The values of β, γ, δ were decided using a grid search cross validation on the values $\{0.2, 0.4, \ldots, 2.0\}$ for each of the parameters. In GOA, the selected values of β, γ, δ were 1.6, 2, and 1 respectively. In information density, the selected value of β was 1.6 to match GOA. All other relevant settings were set to be the same as those in Experiment 1.

Results. We show the results in Figure 5. The x-axis represents the number of training data in the training set, as selected by active learning. We show the accuracy of the GO in comparison with the results from using only GDI or GDE. It is clear that GO outperforms the performance of either GDI or GDE individually. As is shown in the graph, GO is also competitive with ID. The algorithm appears to have similar performances in some areas, though there is a significant difference during the first 100 iterations, and after about 600 iterations. In addition to this, the deficiency scores of GO with ID as a base method show a significant improvement. The deficiency score represents the overall performance of the method taking into account all accuracies from below one point. To evaluate the significance of the values obtained by the deficiency score, we performed a paired t-test on the deficiency scores attained across all 5 iterations. We used the null hypothesis that the deficiency scores average to 1, implying that the methods perform equally as well, and a significance level of 0.1. As can be seen by the deficiency scores in Table 1, when the size of the training set is less than 100, GO preforms much better than the ID. This initial section is very valuable to active learning, as the main goal of active learning is to label very few points, and still achieve a high accuracy. While the deficiency score briefly rises, it consistently stays below one. It also tends to decrease during the latter iterations, implying improved performance over ID during this time, which is further supported by the learning curves and the results from the paired t-test. We can reject our null hypothesis when the size of the training set is 100 or 700, showing that GO is indeed significantly better than ID for datasets of these sizes, but no worse than ID for datasets of sizes in between.

Table 1 Deficiency scores from Experiment 2. A bold number indicates a value where geographical orientation outperforms information density, and an asterisk denotes a statistically significant rate (paired *t*-test $p < 0.1$).

# data	100	200	300	400	500	600	700
score	**.790***	**.859**	**.860**	**.848**	**.838**	**.822**	**.812***

4. Related Work

Density-weighted methods [3] in pool-based sampling scenarios are related to the geographical orientation algorithm. Settles and Craven [4] propose the information density framework. The main idea of their method is that informative instances should not only be uncertain, but also be representative of the underlying distribution in feature space. Their method uses similarity scores among instances to calculate the representativeness of instances. Other density-weighted methods are generalized to the information density framework. The variations of those density-weighted methods is in how they calculate density weights for each instance. Fujii et al. [8] propose the instance selection method that maximizes two aspects of the instances: (a) difference from the labeled instances, and (b) similarity to the unlabeled instances. Although the aspect of (a) is similar to our method, our method differs from their method in two ways. First, their method does not consider splitting features into two feature sets to calculate the selection score. Second, their method calculates the score of (a) in the same manner as the expected error reduction framework [3][9]; that is, they calculate how much the model is likely to change based on current labeled and unlabeled instances. Instead, our method calculates the similarity between the labeled instances directly. McCallum and Nigam [10] propose a density-weighted algorithm for text classification. The algorithm can be regarded as a special case of the information density algorithm by Settles and Craven [4]. Li et al. [11][12] develop an extension of SVMs, called HintSVM, which takes into account the uncertainty and representativeness for the density-weighted method.

There are several studies on active learning for user activity recognition. Alemdar et al. [13] apply active learning to activity recognition in a home setting. They use three uncertainty sampling methods: least confident sampling, margin sampling, and entropy based. They confirmed the effectiveness of active learning methods for this task. Longstaff et al. [14] apply semi-supervised learning and active learning for an activity recognition task. They have compared least confident sampling and three semi-supervised learning algorithms. The experimental results show that a semi-supervised learning algorithm called democratic co-learning achieves competitive performance with least confident sampling. This supports the idea that active learning robustly performs better than semi-supervised learning because it uses additional labeled data. Stikic et al. [15] apply least confident sampling and the co-testing algorithm for activity recognition tasks. They verify that least confident sampling performs better than co-testing for the tasks.

To the best of our knowledge, no previous work has tackled the problem of active learning for automatic transportation mode assignment. In addition, this is the first work to develop an active learning algorithm that explicitly incorporates geo-spatial knowledge.

5. Conclusion

We proposed a novel active learning method that uses geographical orientation in the geographical space separately from the distribution information in the feature space. The geographical orientation algorithm contains two novel aspects: (1) the geographical distance factor assigns high scores to instances that are not surrounded by the instances in the geographical space of the training data; (2) the geographical density factor assigns high scores to instance that are surrounded by the instances in the geographical space of the unlabeled data. Since geographical distance/density factors capture the information that complements the information obtained by conventional methods such as the information density algorithm, we combined those two factors with the conventional information density algorithm to develop the geographical orientation algorithm. Experiments have shown that the geographical orientation algorithm outperforms conventional active learning methods in the automatic transportation mode assignment task. We have concluded that the orientation information in the geographical space has complementary information to that in feature space.

References

[1] Zheng, Y., Liu, L., Wang, L. and Xie, X.: Learning transportation mode from raw gps data for geographic applications on the web, *Proc. WWW '08*, pp. 247–256 (2008).

[2] Zheng, Y., Chen, Y., Li, Q., Xie, X. and Ma, W.-Y.: Understanding transportation modes based on GPS data for web applications, *ACM Trans. Web*, Vol. 4, No. 1, pp. 1:1–1:36 (2010).

[3] Settles, B.: Active learning literature survey, *University of Wisconsin, Madison*, Vol. 52, pp. 55–66 (2010).

[4] Settles, B. and Craven, M.: An analysis of active learning strategies for sequence labeling tasks, *Proc. EMNLP '08*, Association for Computational Linguistics, pp. 1070–1079 (2008).

[5] Baram, Y., El-Yaniv, R. and Luz, K.: Online Choice of Active Learning Algorithms, *J. Mach. Learn. Res.*, Vol. 5, pp. 255–291 (2004).

[6] Schein, A. I. and Ungar, L. H.: Active Learning for Logistic Regression: An Evaluation, *Mach. Learn.*, Vol. 68, No. 3, pp. 235–265 (2007).

[7] Zhu, J. and Ma, M.: Uncertainty-based Active Learning with Instability Estimation for Text Classification, *ACM Trans. Speech Lang. Process.*, Vol. 8, No. 4, pp. 5:1–5:21 (2012).

[8] Fujii, A., Tokunaga, T., Inui, K. and Tanaka, H.: Selective Sampling for Example-based Word Sense Disambiguation, *Comput. Linguist.*, Vol. 24, No. 4, pp. 573–597 (1998).

[9] Roy, N. and McCallum, A.: Toward Optimal Active Learning Through Sampling Estimation of Error Reduction, *Proc. ICML '01*, pp. 441–448 (2001).

[10] McCallum, A. and Nigam, K.: Employing EM and Pool-Based Active Learning for Text Classification, *Proc. ICML '98*, pp. 350–358 (1998).

[11] Li, C.-L., Ferng, C.-S. and Lin, H.-T.: Active Learning with Hinted Support Vector Machine, *Proc. ACML '12*, pp. 221–235 (2012).

[12] Li, C.-L., Ferng, C.-S. and Lin, H.-T.: Active Learning Using Hint Information, Technical report, National Taiwan University (2014).

[13] Alemdar, H., van Kasteren, T. L. M. and Ersoy, C.: Using Active Learning to Allow Activity Recognition on a Large Scale, *Proc. AmI '11*, pp. 105–114 (2011).

[14] Longstaff, B., Reddy, S. and Estrin, D.: Improving activity classification for health applications on mobile devices using active and semi-supervised learning, *Proc. PervasiveHealth '10*, IEEE, pp. 1–7 (2010).

[15] Stikic, M., Van Laerhoven, K. and Schiele, B.: Exploring semi-supervised and active learning for activity recognition, *Proc. ISWC '08*, IEEE, pp. 81–88 (2008).