iSampling: Framework for Developing Sampling Methods Considering User's Interest *

Jinoh Oh and Hwanjo Yu†
Department of Computer Science and Engineering, POSTECH
Pohang, South Korea
{kurin, hwanjoyu}@postech.ac.kr

ABSTRACT

Sampling is one of fundamental techniques for data preprocessing and mining. It helps to reduce computational costs and improve the mining quality. A sampling method is typically developed independently for a specific problem and for a specific user's interest, because it is hard to develop a method that is generalized across various user's interests. An absence of general framework for sampling makes it inefficient to develop or revise a sampling method as user's interest changes. This paper proposes a general framework, iSampling, which facilitates a user developing sampling methods and easily modifying the user's sampling interest in the method. In the framework, a user explicitly describes her sampling interest into a graph model called *interest model*. Then, iSamplingautomatically selects a sample set according to the model, which satisfies the user's interest. In order to demonstrate the effectiveness of our framework, we develop new trajectory sampling methods using our framework; trajectory sampling has been a challenging problem due to its high complexity of data and various user's interests. We demonstrate the flexibility of our framework by showing how easily trajectory samples of different interests can be generated within our framework.

Categories and Subject Descriptors

H.4.0 [**Information Systems Applications**]: General—*Sampling framework*

General Terms

Algorithm, Theory

Keywords

Sampling Framework for various interests, Model-based Sampling, Trajectory Sampling

*This work was partially supported by Mid-career Researcher Program through NRF grant funded by the MEST (No. KRF-2011-0016029). This work was also supported by IT Consilience Creative Program of MKE and NIPA (C1515-1121-0003).
†corresponding author

CIKM'12, October 29–November 2, 2012, Maui, HI, USA.

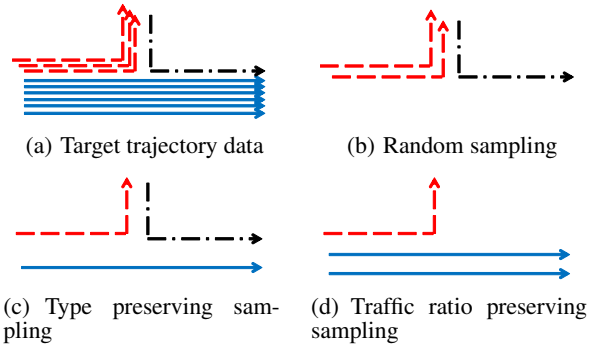
1. INTRODUCTION

Sampling is one of fundamental techniques for data preprocessing and mining. The main purpose of sampling is to reduce the size of target dataset while preserving the characteristics of the dataset. Thus, it helps to reduce computational cost in various applications [1, 2, 3, 12, 13]. Also, an adoption of appropriate sampling techniques provides additional benefits such as enhancing the performance of application [3], reducing the cost (which is not limited to computational time) for data analyzing and gathering [12, 13], and providing solution itself for some problems such as rare-class problem and network traffic inference problem [2].

A sampling method is typically developed independently for a specific problem and for a specific user's interest, because it is hard to develop a method that is generalized across various user's interests. For example, we want to sample 3 trajectories from the trajectory data in Figure 1(a). Figure 1(b), 1(c), and 1(d) show three possible sampling results with different interests – random sampling, type-preserving sampling, and traffic-preserving sampling, respectively. If a user is interested to keep the traffic types, Figure 1(c) might be a desirable sampling result. Similarly, if a user is interested to keep the traffic ratio on each road, Figure 1(d) might be a desirable sampling result.

An absence of general framework for sampling incurs several problems: (1) Researchers have to repeatedly pour the same efforts to develop and verify their sampling methods. (2) The inefficiency of sampling development process results in the shortage of sampling methods. For example, sampling methods for trajectory mining, temporal mining, and pattern mining are not actively proposed though their attentions grow continuously [11]. (3) Sampling interests are sometimes not explicitly specified.

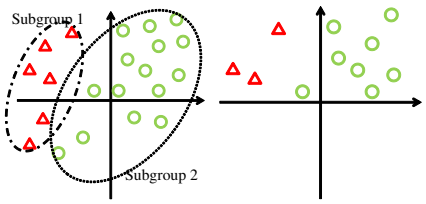
This paper proposes a general framework, iSampling, for developing sampling methods considering the user's interest. Our key idea is that there are *implicit sampling interests* on existing



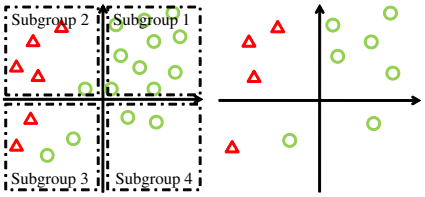
(a) Target trajectory data (b) Random sampling

(c) Type preserving sampling (d) Traffic ratio preserving sampling

Figure 1: Three different trajectory sampling results. A line is a trajectory and there are three types of trajectories.



(a) Class-based subgrouping and its result



(b) Quadrant-based subgrouping and its result

Figure 2: Two different subgrouping policies reflecting different interests for stratified sampling. The shape of points indicate the class.

sampling techniques, and we can extract general *interest model* by formalizing such *implicit sampling interests*. For example, stratified sampling is defined to be sampling based on disjoint and exhaustive subgroups. However, when it is applied to a classification problem, the subgroups implicitly denote the classes. This, focusing on class, is the user's *implicit sampling interest* because it is not the only way to stratify. For example, consider the data of two classes in Figure 2(a). The stratified sampling would generate a different result when the subgroups are quadrants (Figure 2(b)). The data sampled according to the quadrants can be more representative than those sampled from the typical stratified sampling in terms of data distribution.

iSampling facilitates a user developing sampling methods and easily modifying the user's sampling interest in the method. In the framework, a user explicitly describes her sampling interest into a graph model called *interest model*. Then, iSampling automatically selects a sample according to the model, which satisfies the user's interest. In particular, the framework consists of two steps:

- Model construction: a user constructs an interest model, i.e., a graph, which explicitly specifies the user's interest in sampling.
- Sample selection: a sample is selected according to the model.

Sample selection is done automatically, thus a user only needs to change the interest model in order to select a sample of different interest.

In order to demonstrate the effectiveness of our framework, we develop new trajectory sampling methods using our framework; trajectory sampling has been a challenging problem due to its high complexity of data and various user's interests. We demonstrate the flexibility of our framework by showing how easily trajectory samples of different interests can be generated within our framework.

2. RELATED WORKS

2.1 Sampling Method Overview

As we discussed in Section 1, stratified sampling works as follows: (1) At first, a process called *stratification* divides the population into *mutually exclusive and exhaustive homogeneous subgroups*, (2) After that, a simple random sampling is processed within each subgroup proportional to the size of subgroup. Stratified sampling requires the dataset to be well divided into a finite number of *disjoint* groups. However, for some unsupervised problems, it is hard to find strictly mutually exclusive and exhaustive subgroups without loss of generality. For this reason, stratified sampling is not used for unsupervised mining problems such as frequent pattern mining, clustering, trajectory mining.

Under or over-sampling is also popularly used to relieve the problem of imbalanced dataset in classification [8]. For example, Chawla et al. proposed a sampling technique for imbalanced dataset [1]. The key idea of the paper is that combining both under-sampling and over-sampling is better than performing one of them. The paper proposes and evaluates the performance of SMOTE on varying dataset and empirically proves that SMOTE generates higher ROC accuracy for imbalanced dataset.

Yu et al. proposed selective sampling technique for ranking SVM [12]. Proposed sampling technique is a sampling method for picking most informative samples during active learning process. Donmez et al. addressed the problem of how to jointly learn the accuracy of labeling sources and obtain the most informative labels for the active learning task to minimize total labeling effort [3].

Sampling has been also developed for measuring network traffic [2, 4]. Those works aim to observe network traffic by sampling on packet data; they sample packets from networks, and based on the sampled packets, they infer the entire traffic of network.

2.2 Trajectory Sampling

So far, trajectory sampling methods have focused on sampling network packets and inferring overall network traffic. Duffield et al. aims to observe traffic by using sampling techniques [5]. A simple random sampling is used in the paper and it turns out to be useful to observe the traffic data. El Mahrsi et al. proposed a new sampling technique which is capable of constructing high quality summaries of incoming data on the fly [6]. In this paper, sampling is done on a single trajectory, and the sampling target is the intermediate points of a trajectory.

Recently, Pelekis et al. proposed a method which samples trajectories from a trajectory dataset [11]. The method divided the whole spatial domain into a fine-grained grid and, simplified a trajectory to have p line segments. Based on these preprocessed segments and cells, the method calculates the weight of each pair by using voting scheme and uses the score while sampling. However, the method lacks flexibility, and major interest is already fixed as coverage of the sample. The method also requires several parameters to tune and manual preprocessing which could produce information loss.

3. ISAMPLING

3.1 Model Construction

3.1.1 *Model Construction Overview*

The interest model consists of the following primitives, which are actually components for constructing a graph

1. **Interest elements** – nodes of the graph
2. **Criteria function** – computing weights of nodes (*input: a dataset; output: node weights*).
3. **Assigning function** – linking between nodes and data instances (*input: an instance; output: nodes*)
4. **Element distance function** – computing distances (or edges) among nodes (*input: nodes; output: distances among nodes*)

Constructing an interest model corresponds to building the four primitives. First, the user specifies the sampling interest using *interest elements* and *criteria function*. For example, in the quadrant-based stratified sampling (Figure 2(b)), since the user wants to keep

the *quadrant ratio* in the sample, interest elements are the *quadrants*, and criteria function computes the *ratios of each quadrants* from the dataset. Thus, the weights of nodes will be the ratios of the quadrants.

Second, to select data instances or evaluate the quality of data selection, data instances must be related to some interest elements so as to check whether the selected instances satisfy the specified interests. Thus, *assigning function* returns corresponding interest elements (or nodes) for each data instance. For example, in the quadrant-bassed stratified sampling, assigning function returns the quadrant that each data instance belongs to.

Finally, we need distances among nodes in order to measure and minimize the cost of mis-sampling. For example, in the quadrant-based stratified sampling, when we sample instances missing the target interest elements, (e.g., instances that fail to satisfy the quadrant ratio), it will incur sampling costs (e.g., degrade in sampling accuracy). In that case, we should be able to measure the degree of cost and try to minimize the cost in sampling. Thus, *element distance function*, returning distances among nodes, is used in the sample selection step in order to sample according to the model while minimizing the cost. In the example of the quadrant-based stratified sampling, the distances among quadrants may be set equally, or the distance between adjacent quadrants may be set shorter than that between diagonal quadrants.

3.1.2 Model Primitives

In this section, to explain each primitive of interest model in detail, we exemplify the following four cases.

- Case 1: Class-based stratified sampling
- Case 2: Quadrant(or grid)-based stratified sampling
- Case 3: Under- or over-sampling to balance two classes
- Case 4: Traffic-preserving trajectory sampling

Interest elements: Interest elements are the targets on which the user wants to specify sampling criteria. It corresponds to the nodes in the graph. Examples of interest elements follows.

- Case 1 - Class-based stratified sampling: Since the user has an interest on the class ratio, interest elements (or the nodes) are the class labels.
- Case 2 - Quadrant-based stratified sampling: Since the user has an interest on the quadrant ratio, interest elements (or the nodes) are the quadrant labels.
- Case 3 - Under- or over-sampling to balance two classes: Since the user has an interest on the class ratio, interest elements (or the nodes) are the labels of two classes.
- Case 4 - Traffic-preserving trajectory sampling: Since the user has an interest on traffic ratio at a certain spatio-temporal spot, interest elements (or the nodes) are spatio-temporal points, each labeled by a triple of longitude, latitude and time.

Criteria function: Criteria function returns sampling criteria for interest elements (i.e., weights of nodes), which describe the characteristics to preserve in the sample. Criteria function computes normalized weights of nodes ($\sum weights = 1$) from an input of a dataset. (The weights will be used in the sample selection step in order to evaluate the quality of selected sample.) The user defines a criteria function to specify the sampling criteria on interest elements. Depending on the data distribution or concern of users, criteria function can be defined in various ways. Examples of criteria function follow.

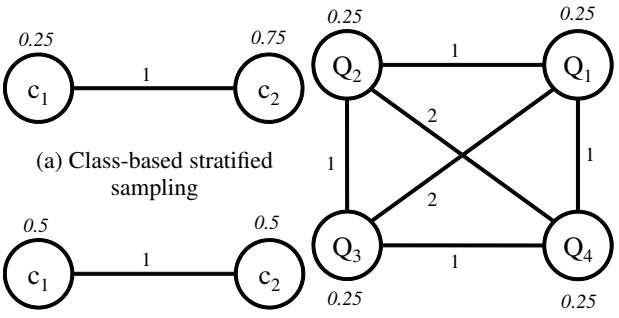
- Case 1 - Class-based stratified sampling: Since the characteristics to preserve is the class ratio, criteria function computes and returns the ratios of classes given the input of dataset.
- Case 2 - Quadrant-based stratified sampling: Since the characteristics to preserve is the quadrant ratio, criteria function computes and returns the ratios of quadrants given the input of dataset.
- Case 3 - Under- or over-sampling to balance two classes: Since the characteristics to preserve is to make the class ratio equal, criteria function returns the same values for both classes (e.g., $1/2$ and $1/2$) regardless of the input of dataset.
- Case 4 - Traffic-preserving trajectory sampling: Since the characteristics to preserve is the traffic ratio, criteria function computes and returns the normalized traffic ratios of spatio-temporal points given the input of trajectory dataset.

Assigning function: In order to sample or select instances matching the specified interest, we need to link each data instance to interest elements. Assigning function returns corresponding nodes of each data instance. This assigning function can be used in criteria function (to compute node weights) and also used in the sample selection step. Examples of assigning function follow.

- Case 1 - Class-based stratified sampling: Since the node is the class, assigning function returns the class of an instance.
- Case 2 - Quadrant-based stratified sampling: Since the node is the quadrant, assigning function returns the quadrant that an instance belongs to.
- Case 3 - Under- or over-sampling to balance two classes: Since the node is the class, assigning function returns the class of an instance.
- Case 4 - Traffic-preserving trajectory sampling: A trajectory is usually related to multiple locations at different times. Since the node is a spatio-temporal spot, assigning function returns a set of spatio-temporal points that the trajectory passes by.

Element distance function: Element distance function computes and returns the distance given a pair of nodes. Element distances are the semantic distances among interest elements (i.e., edges among nodes). Why do we need element distances in the framework? For some cases, it is nearly impossible to sample instances exactly satisfying the specified interest. For instance, in trajectory dataset, it is hard to sample while perfectly preserving the traffic ratio, thus sometimes we have to sample instances missing the interest (e.g., instances that fail to satisfy the traffic ratio), which will incur sampling costs (e.g., inaccurate traffic ratio in the sample set). In that case, we must be able to measure the degree of the cost and try to minimize the cost while sampling. Thus, element distances are used in the sample selection step in order to sample according to the model while minimizing the cost. Examples of element distance function follow.

- Case 1 - Class-based stratified sampling: Since there is no notion of distance among classes in classification, element distances are not meaningful thus it returns the same value (e.g., 1) regardless of node pairs. However, if the distances among classes are different (e.g., class 1 and 3 is farther apart than class 1 and 2), element distance function must return different values according to the semantic distances between classes.
- Case 2 - Quadrant-based stratified sampling: Since the node is the quadrant, element distance function returns the distance between a pair of quadrants.
- Case 3 - Under- or over-sampling to balance two classes: Since the node is the class, element distance is not meaningful just as Case 1.



(a) Class-based stratified sampling

(b) Quadrants-based stratified sampling

(c) Under- or over-sampling

Figure 3: Examples of interest models for existing sampling methods. Italic numbers denote weights of nodes.

- Case 4 - Traffic-preserving trajectory sampling: Since the node is a spatio-temporal spot, element distance must reflect the distance between spatio-temporal points. For example, using Euclidean distance, element distance function can be defined as follows.

$$D(p,q) = \sqrt{(p_x - q_x)^2 + (p_y - q_y)^2 + (p_t - q_t)^2} \quad (1)$$

where p and q are spatio-temporal locations and p_x, p_y and p_t indicate latitude, longitude and time, respectively, of a certain spatio-temporal point p.

Figure 3 shows some examples of interest models. Examples of interest models and how to draw them using the primitives are detailed in our technical report [10].

3.2 Sample Selection

3.2.1 Sample Selection Overview

The aim of an interest model is to describe the status of dataset with respect to the user's sampling interest, thus a selected sample set can be evaluated according to the model whether the sample set satisfies the user's interest. Note that the node weights are generated by criteria function that is defined by the user, and criteria function computes the weights from the dataset. For example, class ratio or quadrant ratio in the stratified sampling is computed from the dataset. If the weights are computed from a selected sample set instead of the original dataset, then this interest model represent the status of the sample set with respect to users' interest.

Our key idea is that, in the optimal result, the sampling set should construct a similar interest model compared to that from the original dataset. Consequently, the quality of a selected sample can be evaluated based on the similarity of two graphs – the graph whose weights are computed from the sample and that computed from the original data set. The remaining challenge is 1) how to compare two interest models, and 2) how to select a sample set which construct a interest model which having minimum distance from that of the original dataset.

3.2.2 Computing Distance Between Models

We transform the problem of measuring the distance between interest models into the transportation problem since the distance is closely related to the concept of flow. For example, suppose an city x_i produces x_i^+ apples and consumes x_i^- apples. Then, the source of the city is x_i^+ and sink is x_i^-. Depending on the amount of source and sink, the city may need to import or export apples to balance the source and sink, i.e., $x_i^+ - x_i^- = 0$. Given a set of cities with source and sink, the goal of transportation problem is to find optimal flows that minimize transportation costs to balance the source and sink of all cities, i.e., $x_i^+ - x_i^- = 0$ for all x_i.

In our application, we treat the normalized weights of model M_U, which is constructed from the original dataset, as source (i.e., the amount of weights to be moved), and the normalized weights of model M_S, which is constructed from the selected sample set, as sink (i.e., the amount of weights to be filled). Then, the problem of measuring distance between two interest models M_U and M_S is transformed to finding minimum cost to balance the weights of source and sink. To solve the transportation problem, we adopt the Earth Mover's Distance (EMD) measure which is frequently used and actively researched in Computer Vision, Statistic, and Privacy and Security societies [9].

Earth Mover's Distance: Let S^+ and S^- are the source and sink defined on a graph G consisting of nodes $\{n_1, \ldots, n_k\}$. Then, $S^+ = \{(n_1, w_1^+), \ldots, (n_k, w_k^+)\}$, and $S^- = \{(n_1, w_1^-), \ldots, (n_k, w_k^-)\}$ where w_k^+ and w_k^- denotes the weight of source and sink at a node n_k, respectively. Earth Mover's Distance (EMD) captures the minimum costs of transporting particles to balance source and sink. The workload, which is needed to balance source and sink, is defined as follows.

$$\text{WORK}(S^+, S^-, F) = \sum_{i=1}^{m} \sum_{j=1}^{n} f_{ij} d_{ij} \quad (2)$$

with constraints

$$f_{ij} \geq 0, \quad 1 \leq i, j \leq k \quad (3)$$

$$\sum_{j=1}^{k} f_{ij} \leq w_i^+ \quad , \quad 1 \leq i \leq k \quad (4)$$

$$\sum_{i=1}^{k} f_{ij} \leq w_j^- \quad , \quad 1 \leq j \leq k \quad (5)$$

$$\sum_{i=1}^{k} \sum_{j=1}^{k} f_{ij} = \min\left(\sum_{i=1}^{k} w_i^+, \sum_{j=1}^{k} w_j^-\right) \quad (6)$$

where f_{ij} is a flow of mass (the amount of moving particles) from n_i to n_j, d_{ij} is a ground distance from n_i to n_j, and F is a full matrix of flow f_{ij} (i.e., $F = [f_{ij}]$). Note that, in our application, equality holds for Equation 4, 5 since we used normalized weights for source and sink. From this setting, EMD measures the minimum workload, defined as

$$D_{EMD}(S^+, S^-) = \min_F \text{WORK}(S^+, S^-, F) \quad (7)$$

More details are in our technical report [10].

3.2.3 Sampling Algorithm

Now, the remaining challenge for sampling process is how to select a sample set that minimizes the EMD distance. We formulate the objective of our optimization problem as follows.

$$\min_{M_s} D_{EMD}(M_u, M_s) = \min_{M_s} \min_F WORK(M_u, M_s, F)$$

Unfortunately, this problem cannot be solved analytically, and calculating the EMD between two interest models is an another optimization problem itself. A naive method is trying all possible combinations of samples and choosing the optimal sample among them. However, such a naive method is not applicable because the complexity of the naive method is ${}_nC_k$, where n is the size of original dataset and k is the size of the sample set, and n is a very large number in most applications. We propose a method based on greedy approach. Our algorithm iteratively picks an instance which produces the minimum EMD. More specifically, our sampling algorithm is processed in the following steps

1. Start with an empty set S.
2. For each instance x_i in D, evaluate EMD(M_U, $M_{S \cup x_i}$).
3. Pick the instance having the minimum EMD and accumulate it to S.
4. Repeat step 2 and 3 until S is filled.

4. TRAJECTORY SAMPLING USING ISAMPLING

This section develops new trajectory sampling methods using our framework. Trajectory data mining has been actively researched, but sampling on trajectory data has not been actively proposed, because users may have various interests on trajectory sampling, and sampling reflecting the user's interest is nontrivial because of the high complexity of trajectory data.

In this section, we are interested to preserve traffic-ratio of each time window on each spatio-temporal spot. In other words, we want to preserve the traffic ratio of each spatio-temporal location. However, it is not easy to preserve the traffic ratio in sampling, because the traffic ratio of each spatio-temporal point is connected to others, and sampling a trajectory affects on the traffic of multiple spatio-temporal locations. We used interest model which is already provided for explanation on primitives.

4.1 Evaluation results

We use a real-world trajectory dataset, Trucks [7]. This dataset contains the trajectory of a fleet of trucks, and consists of 276 individual trajectories. Each trajectory has various length of GPS signal, and the total number of GPS signal is 112,203. To make dataset realistic, we divided a trajectories into multiple instances if the trajectory has a time gap longer than 10 minutes. Our final preprocessed dataset contains 1,938 trajectories with 107,616 GPS signals. More evaluation results and discussion are provided in our technical report [10].

Sampling quality: Our method shows better results with respect to time. Figure 4 shows the difference of traffic ratio from the original dataset at each time window. From this figure, we can easily notify that random sampling technique samples far more trajectories at time window 9 than as it is. And, this over sampling results in under sample for trajectories at time window 10. On the contrary, our method shows smaller difference than random sampling.

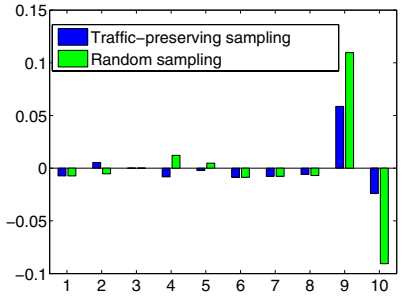


Figure 4: Difference of traffic ratio from the original dataset at each time window. X-axis: time window; Y-axis: sum of each node's traffic ratio differences

Result with varying sizes of sample: We report the change of sampling quality, as the size of sample changes. We run random sampling 100 times and report the average distance. Figure 5 shows the distances using EMD, which considers the semantic distance among interest elements. In this figure, our traffic-preserving sampling shows substantially lower distance than random sampling, which implies that sample set

5. CONCLUSION AND FUTURE WORK

This paper addresses the problem of an absence of general framework for developing sampling technique supporting various user's interests. We propose a general framework, iSampling, in which a user explicitly describes her sampling interest into a graph model called *interest model*, and a sample is selected according to the model. The problem of selecting a sample set is transformed to the problem of choosing the most similar graph based on the concept of flow. We adopt Earth Mover's Distance (EMD) to measure the sampling cost (i.e., distance between interest models), and provide an efficient method for selecting a sample set based on greedy approach. To verify the effectiveness of our framework, we develop new trajectory sampling methods using our framework. Our traffic-preserving sampling methods, developed based on our framework, produce samples reflecting the user's interest and also are flexible so that the user can easily change her interest in sampling.

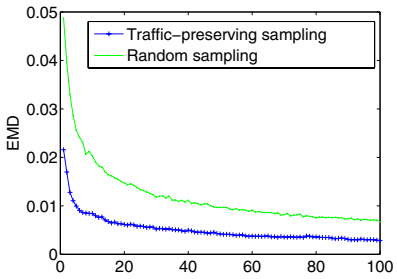


Figure 5: Change of sampling quality w.r.t. the sample size. X-axis: sample size; Y-axis: distance from the original model using two different distance measures

References

[1] N. Chawla, K. Bowyer, L. Hall, and W. Kegelmeyer. SMOTE: synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research*, 16(1):321–357, 2002.

[2] B.-y. Choi and J. Park. Adaptive random sampling for traffic load measurement. *IEEE International Conference on Communications, 2003. ICC '03.*, pages 1552–1556, 2003.

[3] P. Donmez, J. G. Carbonell, and J. Schneider. Efficiently learning the accuracy of labeling sources for selective sampling. *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining - KDD '09*, page 259, 2009.

[4] X. Du, R. Jin, L. Ding, V. Lee, and J. Thornton Jr. Migration motif: a spatial-temporal pattern mining approach for financial markets. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1135–1144, 2009.

[5] N. Duffield and M. Grossglauser. Trajectory sampling for direct traffic observation. *ACM SIGCOMM Computer Communication Review*, 30(4): 271–282, 2000.

[6] M. El Mahrsi, C. Potier, G. Hébrail, F. Rossi, and M. K. E. Mahrsi. Spatiotemporal sampling for trajectory streams. In *Proceedings of the 2010 ACM Symposium on Applied Computing*, pages 1627–1628.

[7] E. Frentzos, K. Gratsias, and N. Pelekis. Nearest neighbor search on moving object trajectories. *Advances in Spatial and Temporal Databases*, 3633: 923–923, 2005.

[8] M. Kubat and S. Matwin. Addressing the curse of imbalanced training sets: One-sided selection. In *Proceedings of the Fourteenth International Conference on Machine Learning*, pages 179–186. Morgan Kaufmann, 1997.

[9] J. Lee, J. Han, and K. Whang. Trajectory clustering: a partition-and-group framework. In *Proceedings of the 2007 ACM SIGMOD international conference on Management of data*, pages 593–604. ACM, 2007.

[10] J. Oh, T. Kim, S. Park, and H. Yu. PubMed Search and Exploration with Real-Time Semantic Network Construction. Technical report, 2012. URL `http://dm.postech.ac.kr/techreport/TechReport-POSTECH-CSE-2012-03-SemanticNetwork.pdf`.

[11] N. Pelekis, I. Kopanakis, C. Panagiotakis, and Y. Theodoridis. Unsupervised trajectory sampling. *Machine Learning and Knowledge Discovery in Databases*, 6323:17–33, 2010.

[12] H. Yu. SVM selective sampling for ranking with application to data retrieval. *Proceeding of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining - KDD '05*, page 354, 2005.

[13] H. Yu and S. Kim. Passive Sampling for Regression. In *Proceedings of the 2010 IEEE International Conference on Data Mining*, pages 1151–1156, 2010.