Exploring Frequency-based Approaches for Efficient Trajectory Classification

Francisco Vicenzi
Universidade Federal de Santa Catarina (UFSC)
Florianópolis, Brazil
francisco.vicenzi@grad.ufsc.br

Lucas May Petry
Programa de Pós-Graduação em Ciência da Computação, Universidade Federal de Santa Catarina (UFSC)
Florianópolis, Brazil
lucas.petry@posgrad.ufsc.br

Camila Leite da Silva
Programa de Pós-Graduação em Ciência da Computação, Universidade Federal de Santa Catarina (UFSC)
Florianópolis, Brazil
camila.leite.ls@posgrad.ufsc.br

Luis Otavio Alvares
Programa de Pós-Graduação em Ciência da Computação, Universidade Federal de Santa Catarina (UFSC)
Florianópolis, Brazil
luis.alvares@ufsc.br

Vania Bogorny
Programa de Pós-Graduação em Ciência da Computação, Universidade Federal de Santa Catarina (UFSC)
Florianópolis, Brazil
vania.bogorny@ufsc.br

ABSTRACT

In the last few years, several trajectory classification methods have been proposed for mobility data collected from GPS devices. Most of them only use information derived from the physical movement of the object, as speed, acceleration, and direction variation. More recently, trajectory data obtained from location-based social networks, based on user check-ins in Points of Interest (POIs), have been used to analyze user mobility patterns, giving rise to the development of methods for this specific type of data. While GPS trajectories are in general dense, and movement is characterized by spatio-temporal features and sequential patterns, social media trajectories are mostly sparse, and we claim that the moving object can be characterized simply by the frequency of visits. In this paper we propose a simple, effective, and efficient trajectory classification method based on POI frequency. With experiments on three real datasets we show that the proposed method outperforms the state of the art and is suitable for large amounts of data.

CCS CONCEPTS

• **Information systems** → **Social networking sites**; **Geographic information systems**; **Data mining**;

KEYWORDS

Trajectory Data Mining, Trajectory Classification, Frequency-based Trajectory Classification, Multiple Aspect Trajectory, Social Media Trajectory Classification

SAC '20, March 30-April 3, 2020, Brno, Czech Republic

ACM ISBN 978-1-4503-6866-7/20/03…$15.00
https://doi.org/10.1145/3341105.3374045

ACM Reference Format:
Francisco Vicenzi, Lucas May Petry, Camila Leite da Silva, Luis Otavio Alvares, and Vania Bogorny. 2020. Exploring Frequency-based Approaches for Efficient Trajectory Classification. In *The 35th ACM/SIGAPP Symposium on Applied Computing (SAC '20), March 30-April 3, 2020, Brno, Czech Republic.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3341105.3374045

1 INTRODUCTION

Mobility data mining has received significant attention in the last few years, specially trajectory classification. Trajectory classification is the task of finding the class label of the moving object based on its trajectories [9]. Examples of trajectory classification are: (i) finding the type of a vessel (e.g. fishing, cargo, tourism, etc), (ii) determining the transportation mode of the moving object (e.g. car, bike, taxi, pedestrian, etc), (iii) determining the type of an animal, and more recently, with the explosion of social media data, (iv) identifying a user based on his/her trajectories. The last problem can be more challenging, because the number of classes is normally very high (e.g. hundreds or even thousands of classes).

Several works have been proposed to classify trajectory data collected with GPS devices, as for instance the works of [9], [4], [17], [15]. These works are based on trajectory features extracted from

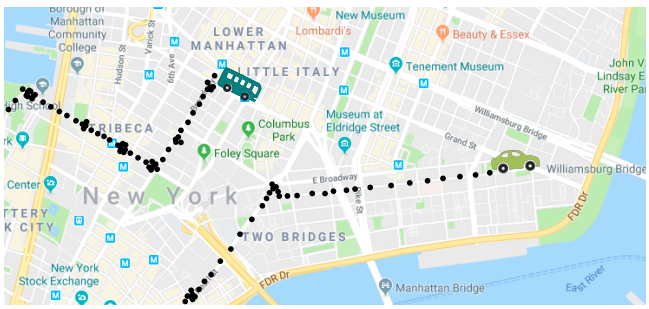

Figure 1: GPS trajectories of a bus and a car

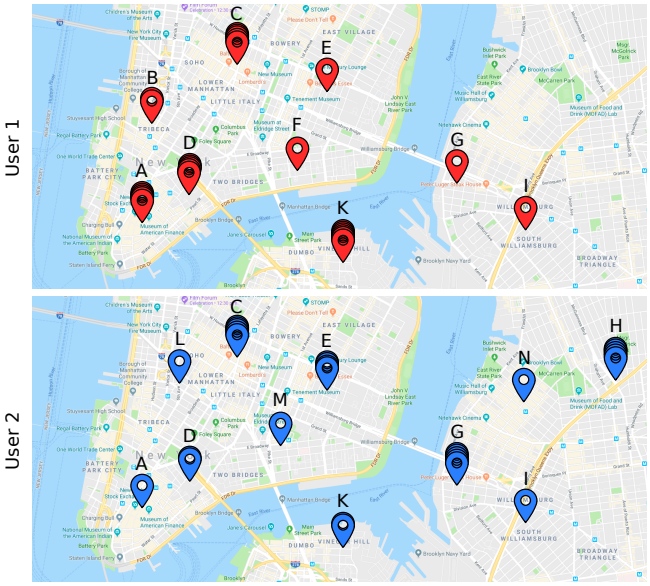

Figure 2: Social media check-ins of user 1 in red and user 2 in blue

the physical spatio-temporal movement behavior as speed, acceleration, direction change, etc. They are important to classify, for instance, the transportation mode of the moving object or the strength of a hurricane, that are highly discriminated by the spatio-temporal features speed and acceleration. Figure 1 shows an example of the trajectory of a bus and a car, where the black points represent their GPS trajectories. We may notice that the points are highly sampled and the movement can be distinguished by the analysis of the individual trajectory of the moving object, as the car moves at a higher speed than the bus (points are more sparse) and a bus has more stopping points and probably exhibits a higher variation in acceleration.

Methods developed for GPS trajectory classification depend on highly sampled data, which is not the case of location-based social network trajectories, that are more sparse in space, as can be seen in the example shown in Figure 2. We may observe in Figure 2 that the detailed movement of the objects is not available, but that the moving objects visit a few places more frequently. For instance, in the figure we see that User 1 frequently visits places A, D, C, and K, while User 2 goes to the places C, E, G, and H more frequently.

We claim that for the problem of social media trajectory classification, i.e., identify the user based on his/her trajectories generated from location based social networks, the frequency of visits is what discriminates the moving object. So far, to the best of our knowledge, there are no methods for trajectory classification based on frequency although the POI frequency is used for other purposes such as activity identification [1], user profile inference [3] and trajectory similarity [14].

The problem of social media trajectory classification, also known as the trajectory-user linking problem, was recently introduced in [7]. Methods for GPS trajectories, that are space-time based methods, are not suitable for this new type of data because the detailed movement between the visited places is not recorded. Only a few works deal with social media trajectory classification, but they do not use frequency. The most well-known works for social media trajectory classification are [7], [18], and [6]. The works of [7] and [18] propose the use of a complex approach based on word embeddings [10] and neural networks for trajectory classification, but they do not achieve a high classification accuracy.

The method proposed in [6], called MOVELETS, deals with both GPS and social media trajectory data, and it has outperformed all previous works, but its main problem is scalability. MOVELETS analyzes all possible subtrajectory sizes, and this process is very time consuming, which can be impracticable on large datasets. The MOVELETS technique finds the subtrajectories that are frequent for one class and that are less frequent for users of other classes. For this purpose it explores all possible subtrajectories in the dataset and their distance to all other trajectories.

In this paper, we propose a frequency-based approach for trajectory classification that is more efficient than MOVELETS, yet we achieve similar or even better accuracy results in comparison with MOVELETS. We take inspiration from the idea of frequency analysis present in MOVELETS, and also from the fact that humans in general follow some routine, i.e., visit a few places but several times, thus having a recurrent pattern [8]. We show with a robust experimental evaluation that the method is as accurate as the MOVELETS approach and is scalable, supporting the analysis of big data, since it is much less time and memory consuming. Indeed, we show that our simple method outperforms the works [7] and [18], that use a much more complex structure, in a variety of datasets.

The remainder of the paper is structured as follows: Section 2 presents some basic concepts and related works, Section 3 describes the proposed method, Section 4 shows the experiments and the discussion of the results, and Section 5 presents our conclusions and future work.

2 BASIC CONCEPTS AND RELATED WORK

In this section we present the basic concepts associated to the method proposed in this paper and describe the main related works in the literature.

2.1 Basic Concepts

The main concept related to this work is the concept of trajectory. Since there are several definitions of trajectory in the literature, in this paper we use the following definition.

Definition 2.1. **Trajectory**. A trajectory T is a sequence of points $\langle p_1, p_2, \ldots, p_n \rangle$, with $p_i = (x, y, t, \mathcal{A})$ being the i-th point of the trajectory at location (x, y) and timestamp t, described by the set $\mathcal{A} = \{a_1, a_2, \ldots, a_r\}$ of r attributes.

The set $\mathcal{A}$ of attributes can contain any information associated to that point such as temperature, weather condition, means of transportation, mood, etc. For a social network trajectory, typically each point is a check-in and contains information like the name of a Point of Interest (POI), and related attributes such as the rating and the price tiers of that POI.

Given that the objective of the method proposed in this work is trajectory classification, we define this type of problem as follows:

Definition 2.2. **Trajectory Classification**. Given a trajectory set defined by a set of pairs $\mathbf{T} = \{(T_1, class_{T_1}), (T_2, class_{T_2}), \ldots, (T_n, class_{T_n})\}$, where each pair contains a trajectory and its class label, *trajectory classification* is the task of learning a prediction function f that maps each trajectory T_i of $\mathbf{T}$ to one of the predefined class labels.

Some well-known metrics used to evaluate experiments of classification are ACC@K, Macro-Precision, Macro-Recall and Macro-F1. Their use in classification, as well as in information retrieval, are very common. ACC@K quantifies the accuracy of a technique on correctly predicting the class label of a trajectory, with its probability ranked within the top K most probable class labels. It is computed as

$$\text{ACC@K} = \frac{1}{\mathcal{T}_{test}} \sum_{i=1}^{\mathcal{T}_{test}} \begin{cases} 1, & \text{if } class_{T_i} \in \mathcal{L}_K(T_i) \\ 0, & \text{otherwise,} \end{cases} \quad (1)$$

where $\mathcal{T}_{test}$ is the number of trajectories in the test set, $class_{T_i}$ is the true class label of trajectory T_i, and $\mathcal{L}_K(T_i)$ is the set of K class labels with the highest probabilities predicted by the model for trajectory T_i.

Macro-Precision and Macro-Recall are the average precision and recall among all classes, respectively. Macro-F1 is calculated by the harmonic mean of Macro-Precision and Macro-Recall, as follows:

$$\text{Macro-F1} = \frac{2 \times \text{Macro-Precision} \times \text{Macro-Recall}}{\text{Macro-Precision} + \text{Macro-Recall}} \quad (2)$$

2.2 Related Work

Existing works for trajectory classification extract different trajectory features in a pre-processing step and use these features as input to conventional classifiers. Most works deal only with raw trajectories, such as [9], [4], [17], [15], and [5]. A *raw trajectory* is a sequence of points (x, y, t) where (x, y) correspond to the position of the moving object in space at time t. From these points, in general, either they segment the trajectories in smaller parts or they extract spatio-temporal features as speed, acceleration, direction change, etc. One of the first techniques in the literature for trajectory classification was TraClass, proposed by Lee in [9]. TraClass is a generic method that can be used to classify different types of moving objects, and is strongly based on the spatial dimension. In [11] Patel extended this method to support the temporal dimension.

More recently, Ferrero [6] proposed the method MOVELETS, which outperformed all previous works. MOVELETS can deal with any type of information besides space and time. However, it is very time consuming, as it explores all possible subtrajectories and their distances to all trajectories in the dataset.

There are not many works that deal with multiple aspect or semantic trajectories, i.e., trajectories that have more information than spatial position and time, as for instance, POI name, weather conditions, temperature, etc. Besides of the previously mentioned work of MOVELETS [6] that can be used for classifying any type of trajectory, the main works that support semantic dimensions are developed for a specific application as transportation mode classification [12, 13] and the identification of users of social media data [7, 18].

Tragopoulou in [12] infers features as the trajectory average speed, the altitude change, whether the day of the week is a workday or not, and if the point is near to a metro station or not. Varlamis in [13] extended the previous work by adding the information of whether the point is near of touristic places, and whether it is in a bus or train line. In both techniques the extracted features are passed as input for training classification models for identifying trajectory transportation modes.

Gao [7] and Zhou [18] were the first to propose trajectory classification methods using Recurrent Neural Networks (RNNs), based on a well-known text mining technique for word embeddings, namely *word2vec* [10]. Gao in [7] proposed Bi-TULER, a technique that uses an RNN with the embedding representation of POI identifiers. Mikolov [10] originally proposed a method for embedding a word by identifying patterns in text based on the neighboring words of such word, further representing the embedded word as a numerical feature vector. Analogously, Gao embed POIs based on the context in which they appear in user trajectories. Similar POIs will probably have similar embeddings, i.e., similar feature vectors. Bi-TULER uses trajectories of check-ins extracted from Location-Based Social Networks (LBSNs), and it only considers the POI identifiers assigned to each check-in for trajectory classification. Zhou [18] proposed the TULVAE technique, which is an extension of Bi-TULER that also incorporates a variational autoencoder, thus being able to extract interpretable representations of POI dependencies present in trajectories.

3 THE PROPOSED APPROACH

The approach proposed in this paper was developed having in mind the works of Gao [7] and Zhou [18], which were the first works to address the problem of identifying the user from the sequence of his/her check-ins on location-based social networks.

According to [8], human trajectories typically exhibit some spatial patterns where people visit only a few number of POIs, but several times. With that in mind, in our approach we explore the frequency of the visits to each POI by the users. Thus, we extract features based on the frequency of these visits from the trajectories to, afterward, apply it to a classifier. Although trajectories may have several attributes, we claim that the frequency of visits to the POIs is highly discriminative to classify user behavior.

We propose three alternative approaches to consider the frequency of visits to a POI for trajectory classification: (i) POI frequency, (ii) normalized POI frequency, and (iii) weighted normalized POI frequency.

The POI Frequency (POI-F) is based on considering only the frequency that a POI is visited within a trajectory. POI-F is described in Definition 3.1:

Definition 3.1. **POI Frequency**. Given a trajectory T and a POI P, the frequency of P in T is computed as

$$\text{POI-F}(P, T) = \text{the number of visits of P in T.}$$

We describe in Algorithm 1 the steps for extracting the *POI-F* feature. The input of the algorithm is a dataset with trajectories that have the information of the visited POIs (POI names or POI identifiers), and the output is a matrix with the POI frequency in

each trajectory. Thus, for all trajectories (Line 4), it iterates over all their points (Line 5), adding one to each respective POI (Line 6).

Algorithm 1: Find POI Frequency

```
   input  : D // a trajectory dataset
   output : M // a matrix with POI frequencies
1  m ← number of trajectories in D;
2  n ← number of different POIs in D;
3  M ← a matrix n × m of zeros;
4  for each trajectory T_i in D do
5      for each POI P_j in T_i do
6          M[i][j] ← M[i][j] + 1;
7  return M;
```

The output of Algorithm 1 is matrix M, where each line corresponds to one trajectory and each column corresponds to one POI, having in the cells the frequency of that POI in the respective trajectory. This matrix is used as the input for the classification method, such as C4.5, Neural Network, SVM, etc.

Similarly to the *POI-F*, the Normalized POI Frequency *(NPOI-F)* is based on calculating the frequency of visits of the POIs within a trajectory. The difference is that this frequency is normalized in relation to the length of the trajectory (i.e. the number of points). The intuition is that, if the length of the trajectories have high variation, a normalized value is better than an absolute value, because it is comparable across trajectories of different length. The feature *NPOI-F* is formalized in Definition 3.2:

Definition 3.2. **Normalized POI Frequency**. Given a trajectory T and a POI P, the frequency of P in T is computed as

$$\text{NPOI-F}(P, T) = \frac{\text{POI-F(P,T)}}{\text{\# of points in T}}$$

Calculating the *NPOI-F* feature is quite similar to the *POI-F*, described in Algorithm 1. The only difference consists in a final step that divides each element from the *POI-F* matrix by the respective trajectory size.

The last feature is the Weighted Normalized POI Frequency *(WNPOI-F)*, which differs from the previous features as it adds a weight for each POI. The weight necessity is inspired by the premise that some POIs are more important than others. For example, in classification tasks, a POI that is visited by only a few users may be more important than a POI visited by every user, because it better discriminates a few individuals. WNPOI-F is defined as follows.

Definition 3.3. **Weighted Normalized POI Frequency**. Given a dataset D of trajectories, a trajectory T ∈ D, and a POI P, the weighted normalized POI frequency of P in D is computed as

$$\text{WNPOI-F}(P, T, D) = \text{NPOI-F} * log\Big(\frac{\text{\# of users in D}}{\text{\# of users that have visited P}}\Big)$$

Algorithm 2 describes the method to calculate *WNPOI-F*. The inputs are a trajectory dataset and a matrix with the *NPOI-Fs* extracted from the dataset, while the output is a matrix with the weighted normalized POI frequency in each trajectory.

At first, the algorithm computes how many different users visited each POI. To do this, it iterates over the points of all trajectories from

Algorithm 2: Find Weighted Normalized POI Frequency

```
   input  : D // a trajectory dataset
            F // a matrix with the NPOI-F from D
   output : F // a matrix with the WNPOI-F
1  l ← number of classes in D; // number of users
2  n ← number of different POIs in D;
3  P ← an array with length n of zeros;
4  for each user l_i in D do
5      for each POI P_j in l_i do
6          if P_j not already counted for l_i then
7              P[j] ← P[j] + 1;
8  for each trajectory T_i in D do
9      for each POI P_j in T_i do
10         F[i][j] ← F[i][j] × log(l / P[j]);
11 return F;
```

each user, incrementing a counter for the POIs visited by the user (Lines 4-7). Then, it iterates over the Normalized POI Frequency matrix, multiplying the normalized frequency of a user in a certain POI by the logarithm of the division between the total number of users (classes) and the number of users that visited that POI (Line 10).

The time complexity to compute the proposed features is $O(m \times s)$, where m stands for the number of trajectories in the dataset, while s stands for the average length of the trajectories. With regards to space complexity, it is $O(m \times n)$, where m is the number of trajectories and n is the number of different POIs. In comparison, the MOVELETS [6] method is $O(m^2 \times n^3)$ for time complexity and $O(m \times s^2)$ for space complexity.

4 EXPERIMENTAL EVALUATION

We evaluate the proposed approach for trajectory classification with three real datasets. They were obtained from three location-based social networks: Brightkite [2], Gowalla [2], and Foursquare [16].

We compare our method with two approaches developed specifically for the problem of identifying the user of social network trajectories: Bi-TULER [7] and TULVAE [18]. In addition, we compare our work with MOVELETS [6], that outperformed all other trajectory classification methods for raw trajectories and also supports semantic features such as the visited POI. For reproducibility purposes, we made the source code of the experiments available on GitHub[1].

We describe the experimental setup and present the results on Brightkite, Gowalla, and Foursquare, in Sections 4.1, 4.2, and 4.3, respectively. We also conduct an additional experiment on the Foursquare dataset (Section 4.4), showing the robustness of our method in a more realistic and harder scenario. In Section 4.5, we demonstrate the scalability of our approach, followed by a brief discussion in Section 4.6.

[1] https://github.com/bigdata-ufsc/vicenzi-2020-poifreq

4.1 Experiment with the Brightkite dataset

The first experiment uses the dataset from Brightkite [2], which was a location-based social network where users shared their locations by checking in, and that was used in [7, 18] to classify users based on their check-in identifiers. Each check-in contains the anonymized user id, the timestamp, the spatial location (latitude and longitude), and the check-in venue (POI), identified by a venue-id. From the original dataset containing more than 4.5 million check-ins, collected between 2008 and 2010, we selected places with at least 10 check-ins. We segmented trajectories into weekly trajectories with at least 10 check-ins, limited to 50 check-ins, and users with at least 10 trajectories, resulting in 54,247 weekly trajectories of 2,042 users. We randomly selected 300 users for our experimental evaluation obtaining 7,911 trajectories. The class labels are the 300 user identifiers. Table 1 summarizes this dataset.

Table 1: Details of the Brightkite dataset

Minimum trajectory size	10
Maximum trajectory size	50
Number of trajectories	7,911
Number of points	130,494
Number of different POIs	4,913
Attributes	Lat, Lon, Time, POI, UserId
Class	Users (300)

We compare all methods on a stratified holdout evaluation with 67% of the dataset for training and 33% for test. For all methods we use only one attribute, the POI, since Bi-TULER and TULVAE are limited to this attribute. For Bi-TULER and TULVAE, we used the same parameters reported in the respective papers. We build a Bidirectional RNN from word embeddings extracted from the entire dataset. For MOVELETS and the three versions of the proposed approach (POI-F, NPOI-F, and WNPOI-F), we build classification models using a neural network with a single-hidden layer containing 100 units. We train the models using the Adam optimizer, a learning rate of 10^{-3}, batch size of 64, and we also apply a dropout rate of 0.5 in order to reduce the overfitting of the models to the training data.

The results of this experiment are shown in Table 2, where the best results are presented in bold and the second best results are underlined, for each metric (ACC@1, ACC@5 and Macro-F1) described in Section 2.1. As can be seen, the three versions of the proposed method outperformed the state of the art methods. We highlight that for this dataset, the best results over all measures were achieved by the Normalized POI Frequency NPOI-F with an accuracy of 95.34% (4.66% of error). Among the state of the art methods, Bi-TULER achieved the best results, with 90.64% of accuracy (9.36% of error). We may observe that NPOI-F reduces the classification error in comparison to Bi-TULER in 50.21% ($1 - 4.66/9.36$).

4.2 Experiment with the Gowalla dataset

The second experiment uses the dataset from Gowalla [2], which was a location-based social network where users shared their locations by checking-in. This dataset was also used by [7]. Each check-in contains the anonymized user id, the timestamp, the spatial location (latitude and longitude), and the check-in venue (POI). From the original dataset containing more than 6 million check-ins, collected between 2009 and 2010, we selected places with at least 10 check-ins. We segmented trajectories in weekly trajectories with at least 10 check-ins, limited to 50 check-ins, and users with at least 10 trajectories, resulting in 33,816 weekly trajectories of 1,952 users. We randomly selected 300 users for experimental evaluation obtaining 5,329 trajectories. The class labels are the 300 user identifiers. Table 3 summarizes the obtained dataset.

Table 2: Classification results for the Brightkite dataset

Method	ACC@1	ACC@5	Macro-F1
Bi-TULER [7]	90.64	95.55	87.92
MOVELETS [6]	90.55	93.79	89.41
TULVAE [18]	88.41	92.15	83.63
POI-F	<u>95.29</u>	**97.98**	<u>93.80</u>
NPOI-F	**95.34**	**97.98**	**93.91**
WNPOI-F	95.13	<u>97.94</u>	93.68

Table 3: Details of the Gowalla dataset

Minimum trajectory size	10
Maximum trajectory size	50
Number of trajectories	5,329
Number of points	98,158
Number of different POIs	24,374
Attributes	Lat, Lon, Time, POI, UserId
Class	Users (300)

For this dataset we used the same experimental configuration of the previous experiment, and the results are shown in Table 4. As in the previous experiment, the frequency-based proposed approach outperformed all other methods. Independently of the frequency approach, all of them outperform the state-of-the-art, and the second best method is the MOVELETS.

Table 4: Classification results for the Gowalla dataset

Method	ACC@1	ACC@5	Macro-F1
Bi-TULER [7]	66.15	78.36	63.26
MOVELETS [6]	91.44	94.04	90.25
TULVAE [18]	67.94	78.76	64.91
POI-F	<u>93.22</u>	<u>97.62</u>	**92.54**
NPOI-F	**93.46**	**97.68**	<u>91.91</u>
WNPOI-F	93.11	97.45	91.66

4.3 Experiment with the Foursquare dataset

The third experiment uses the Foursquare [16] dataset with check-ins (mostly in New York city) between 2012 and 2013. The original dataset has 227,428 check-ins of 1,083 distinct users. Each check-in

is composed by the anonymized user identification, the timestamp of the check-in, and the corresponding Foursquare venue id (POI).

This dataset was pre-processed by applying the following steps: we removed check-ins belonging to broad categories such as roads, rivers, cities, neighborhoods, etc, and duplicated check-ins (considering a 10-minutes threshold); we segmented the trajectories into weekly trajectories and selected those with at least 10 check-ins and users with at least 10 trajectories, resulting in 3,079 weekly trajectories of 193 users. The class label is the user identifier.

Table 5 gives details about the resulting dataset. The experimental set up was the same of previous experiments. The results of this experiment are presented in Table 6. Once again, the proposed method outperformed the other methods: MOVELETS by a very small margin but Bi-TULER and TULVAE by a wide margin.

Table 5: Details of the Foursquare dataset

Minimum trajectory size	10
Maximum trajectory size	144
Number of trajectories	3,079
Number of points	66,962
Number of different POIs	13,848
Attributes	Lat, Lon, Time, POI, UserId
Class	Users (193)

Table 6: Classification results for the Foursquare dataset

Method	ACC@1	ACC@5	Macro-F1
Bi-TULER [7]	48.20	67.38	40.56
MOVELETS [6]	97.66	98.93	96.45
TULVAE [18]	54.33	73.81	46.54
POI-F	98.05	**99.41**	97.29
NPOI-F	**98.24**	**99.41**	97.37
WNPOI-F	**98.24**	**99.41**	**97.54**

4.4 Experiment on Foursquare using the POI category

In this experiment we use the same Foursquare dataset of Section 4.3, but we replaced the POI identifier by the most general POI category[2], in order to make the problem more difficult. Of course, we do not use the spatial location, as it is a highly discriminative factor of the user movement [8]. This experiment represents a more realistic scenario in which the specific POI locations visited by the users might not be available due to privacy concerns. For comparison purposes, we use the same experimental setup of previous experiments.

The results are presented in Table 7. Indeed, the results show that this classification problem is much more difficult compared to the previous experiment on Foursquare, with the POI category instead of the exact POI, since the best accuracy ACC@1 decreased from 98.24 with the exact POI to only 36.22 with the POI Category. Besides, there were no large variations in the results between the methods, different from the previous experiment with the Foursquare dataset.

Table 7: Classification results for the Foursquare dataset considering only the POI category

Method	ACC@1	ACC@5	Macro-F1
Bi-TULER [7]	33.50	60.76	28.29
MOVELETS [6]	35.34	65.43	29.39
TULVAE [18]	32.81	61.15	28.14
POI-F	**36.22**	**67.96**	**31.12**
NPOI-F	35.24	67.47	28.30
WNPOI-F	34.07	67.47	30.26

Given the restricted capability of the methods to correctly identify a user based solely on the POI Category instead of the exact POI identifier, we perform another experiment in order to understand how our method can take advantage of additional semantic information. In order to do this analysis, we included three other attributes in this dataset: the POI rating, the POI price tier, and the weather condition. We obtained the POI rating and price tier for each POI from the Foursquare API[3], and the weather condition was collected via the Weather Wunderground API[4]. These attributes considered in this new scenario are described in Table 8.

Table 8: Foursquare attributes description.

Attribute	Type	Range or examples
Time	Temporal	[00:00,23:59]
POI Category	Nominal	Foursquare categories
Price	Numeric	$\{1, 2, 3, 4\}$
Rating	Numeric	$[4.0, 10.0]$
Weather condition	Nominal	{Clear, Cloudy, Fog, Haze, Rainy, Snow}

To use the proposed methods with more than one attribute, we must concatenate the attribute values of the respective trajectory points. For instance, to consider POI Category and Price, the frequencies are counted for each combination of POI Category and Price, such as "Food+1" (POI Category = *Food* and Price = *1*), "Food+2" (POI Category = *Food* and Price = *2*), etc. In order to implement this approach, all attributes must be discretized. Therefore, for the time attribute we discretized by hour (having a total of 24 possible values), and for the Rating attribute we discretized considering one decimal position (having a total of 62 possible values). For the MOVELETS method we considered the following distance measures: for the Time attribute, the difference in minutes; for the POI Category, a binary distance (0 for the same category and 1 for different categories); for Rating and Price the Manhattan distance;

[2]Foursquare categories are Shop & Service, Professional & Other Places, Food, Travel & Transport, Outdoors & Recreation, Arts & Entertainment, Residence, Nightlife Spot, Event, College & University.

[3]https://developer.foursquare.com/

[4]https://www.wunderground.com/weather/api/

and for the Weather condition a binary distance (0 or 1). The final computed distance is the average of the normalized distance measures of all considered attributes.

For this experiment we ran MOVELETS and the proposed methods with several combinations of the attributes, and present in Table 9 the best result achieved by each method with the corresponding attribute combination used. We do not present results for Bi-TULER and TULVAE, because these methods do not support other attributes.

Table 9: Classification results for the Foursquare dataset considering additional attributes

Method	Attributes	ACC@1
MOVELETS [6]	POI Category + Time	44.59
POI-F	POI Category + Rating	71.56
NPOI-F	POI Category + Price + Rating	**74.09**
WNPOI-F	POI Category + Price + Rating	**74.09**

As we can observe from the results, considering more attributes together with the POI Category the accuracy has significantly increased. This is because these additional attributes increase the semantics of the user check-ins and, therefore, it is easier for the methods to discriminate between different users. Although we achieve a higher discriminative power with these new attributes, the user privacy is not affected as much as in previous experiments, because now we rely on user preference information instead of the specific locations that he/she visits.

With multiple attributes, the best results were achieved with NPOI-F and WNPOI-F using the attributes POI Category, Price, and Rating. It is interesting to note that for MOVELETS, which is a distance-based method, the best result was obtained considering the attributes POI Category and Time, in contrast to the proposed methods that performed best using the attributes POI Category, Price and Rating.

4.5 Scalability Analysis

In this section we present a scalability analysis, comparing the processing time of the proposed methods POI-F, NPOI-F and WNPOI-F with MOVELETS, that outperformed the methods Bi-TULER and TULVAE. Although the processing time is not so important for classification as it is for clustering or association, since the classification model is generated only once, if the processing time is too high it can prevent the use of a method for very big datasets.

We performed two experiments: (i) fixing the number of trajectories in 200 and varying the length of each trajectory from 10 to 400 points, and (ii) fixing the size of the trajectories in 20 points and varying the number of trajectories from 100 to 4,000. For each data configuration and every method we executed 10 runs and reported the respective average computation times. The scalability analysis was performed on an Intel Core i7-4790S CPU @ 3.20GHz, with 8 cores, 8 GB of memory, running Ubuntu 18.04.1 LTS. We used only one thread in this experiment.

Figure 3 shows the experiment varying the length of the trajectories. It is important to note that in this plot, the time is represented in logarithm scale. As can be observed in the figure, the methods proposed in this paper (POI-F, NPOI-F and WNPOI-F) scale much better than the method MOVELETS. Considering the right side of the graph (trajectories with 400 points) the proposed methods are more than 10,000 times faster than MOVELETS.

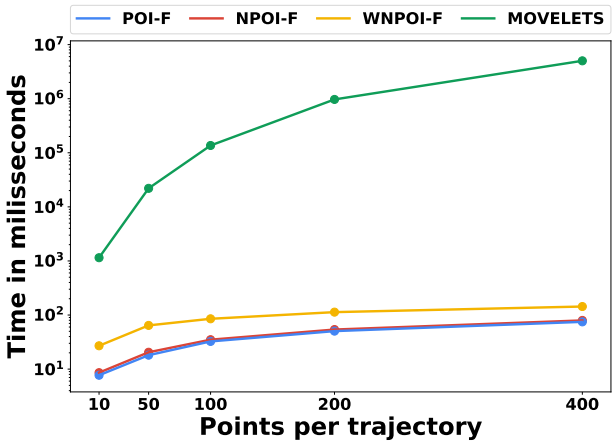

Figure 3: Computational time spent by POI-F, NPOI-F, WNPOI-F, and MOVELETS, varying the length of the trajectories

Figure 4 presents the comparison varying the number of trajectories. In this case, for 4,000 trajectories, the proposed methods are more than 1,000 times faster than MOVELETS.

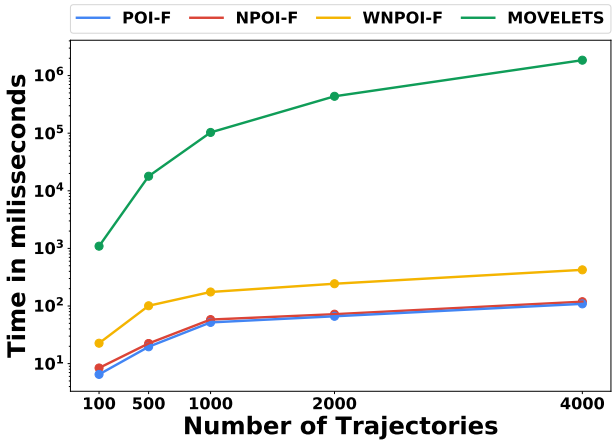

Figure 4: Computational time spent by POI-F, NPOI-F, WNPOI-F, and MOVELETS, varying the number of trajectories

4.6 Discussion

The results of all experiments showed that the proposed methods outperformed the accuracy of the methods Bi-TULER and TULVAE for a large margin, and the accuracy of the method MOVELETS by a small margin. In fact, the achieved results corroborate with our claim in Section 1: simply by analyzing the frequency in which

users visit different places we are able to discriminate between distinct users and, therefore, obtain high classification accuracy results. In addition, our approach scales easily to very large datasets, due to its linear complexity of both space and time as seen in our experiments in Section 4.5.

Even though the frequency is a highly discriminative factor of human behavior, a possible limitation of the proposed methods is that they do not consider the sequence of check-ins. In our experiments, however, the results showed that the sequence of check-ins is less important than the frequency of visits, since we outperformed MOVELETS, which takes into account the sequential aspect.

5 CONCLUSIONS AND FUTURE WORKS

In this paper we proposed a simple, effective and efficient method for trajectory classification, where the features are generated based on the frequency of the visits to the POIs by the users. We extensively evaluated the proposed method with data from three real location-based social networks: Gowalla, Brightkite and Foursquare.

We compared the proposed methods with two complex methods, Bi-TULER [7] and TULVAE [18], that use state-of-the-art techniques as word embeddings and neural networks, and with MOVELETS [6], that outperformed all previous works for raw trajectories. The experiments have shown that the accuracy of our work is much higher than Bi-TULER and TULVAE, and is equal or higher than the MOVELETS approach, which is so far the best state-of-the-art method for trajectory classification. Equally important is the fact that our approach is much faster than the others, what makes it feasible to be used for big trajectory data.

Future works include the extension of the proposed method to consider the sequence of POIs.

ACKNOWLEDGEMENTS

This work has been partially supported by the Brazilian agencies CAPES (Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Finance Code 001), CNPQ (Conselho Nacional de Desenvolvimento Científico e Tecnológico), FAPESC (Fundação de Amparo a Pesquisa e Inovação do Estado de Santa Catarina - Project Match - Co-financing of H2020 Projects - Grant 2018TR 1266).

REFERENCES

[1] Marco Aurelio Beber, Carlos Andres Ferrero, Renato Fileto, and Vania Bogorny. 2017. Individual and Group Activity Recognition in Moving Object Trajectories. *JIDM* 8, 1 (2017), 50–66. https://seer.lcc.ufmg.br/index.php/jidm/article/view/4601

[2] Eunjoon Cho, Seth A Myers, and Jure Leskovec. 2011. Friendship and mobility: user movement in location-based social networks. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 1082–1090.

[3] Lucas Andre de Alencar, Luis Otávio Alvares, Chiara Renso, Alessandra Raffaetà, and Vania Bogorny. 2015. A Rule-based Method for Discovering Trajectory Profiles. In *The 27th International Conference on Software Engineering and Knowledge Engineering, SEKE 2015, Wyndham Pittsburgh University Center, Pittsburgh, PA, USA, July 6-8, 2015*, Haiping Xu (Ed.). KSI Research Inc. and Knowledge Systems Institute Graduate School, 244–249. https://doi.org/10.18293/SEKE2015-143

[4] Somayeh Dodge, Robert Weibel, and Ehsan Forootan. 2009. Revealing the physics of movement: Comparing the similarity of movement characteristics of different types of moving objects. *Computers, Environment and Urban Systems* 33, 6 (2009), 419–434.

[5] Mohammad Etemad, Amílcar Soares Júnior, and Stan Matwin. 2018. Predicting Transportation Modes of GPS Trajectories using Feature Engineering and Noise Removal. In *Advances in Artificial Intelligence: 31st Canadian Conference on Artificial Intelligence, Canadian AI 2018, Toronto, ON, Canada, May 8–11, 2018, Proceedings 31*. Springer, 259–264.

[6] Carlos Andres Ferrero, Luis Otavio Alvares, Willian Zalewski, and Vania Bogorny. 2018. MOVELETS: Exploring Relevant Subtrajectories for Robust Trajectory Classification. In *Proceedings of the 33rd ACM/SIGAPP Symposium on Applied Computing, Pau, France*. 849–856.

[7] Qiang Gao, Fan Zhou, Kunpeng Zhang, Goce Trajcevski, Xucheng Luo, and Fengli Zhang. 2017. Identifying Human Mobility via Trajectory Embeddings. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence (IJCAI'17)*. AAAI Press, 1689–1695. http://dl.acm.org/citation.cfm?id=3172077.3172122

[8] Marta C. González, César A. Hidalgo, and Albert-László Barabási. 2008. Understanding individual human mobility patterns. *Nature* 453, 7196 (June 2008), 779–782.

[9] Jae-Gil Lee, Jiawei Han, Xiaolei Li, and Hector Gonzalez. 2008. TraClass: trajectory classification using hierarchical region-based and trajectory-based clustering. *Proceedings of the VLDB Endowment* 1, 1 (2008), 1081–1094.

[10] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781* (2013).

[11] Dhaval Patel, Chang Sheng, Wynne Hsu, and Mong Li Lee. 2012. Incorporating duration information for trajectory classification. In *Data Engineering (ICDE), 2012 IEEE 28th International Conference on*. IEEE, 1132–1143.

[12] Spiridoula Tragopoulou, Iraklis Varlamis, and Magdalini Eirinaki. 2014. Classification of movement data concerning user's activity recognition via mobile phones. In *Proceedings of the 4th International Conference on Web Intelligence, Mining and Semantics (WIMS14)*. ACM, 42: 1–6.

[13] Iraklis Varlamis. 2015. Evolutionary data sampling for user movement classification. In *Congress on Evolutionary Computation (CEC), 2015*. IEEE.

[14] Xiangye Xiao, Yu Zheng, Qiong Luo, and Xing Xie. 2014. Inferring social ties between users with human location history. *Journal of Ambient Intelligence and Humanized Computing* 5, 1 (01 Feb 2014), 3–19. https://doi.org/10.1007/s12652-012-0117-z

[15] Zhibin Xiao, Yang Wang, Kun Fu, and Fan Wu. 2017. Identifying Different Transportation Modes from Trajectory Data Using Tree-Based Ensemble Classifiers. *ISPRS Int. J. Geo-Inf.* 6, 2 (2017), 57.

[16] Dingqi Yang, Daqing Zhang, Vincent W Zheng, and Zhiyong Yu. 2015. Modeling user activity preference by leveraging user spatial temporal characteristics in LBSNs. *IEEE Transactions on Systems, Man, and Cybernetics: Systems* 45, 1 (2015), 129–142.

[17] Yu Zheng, Yukun Chen, Quannan Li, Xing Xie, and Wei-Ying Ma. 2010. Understanding transportation modes based on GPS data for web applications. *ACM Transactions on the Web (TWEB)* 4, 1 (2010), 1–36.

[18] Fan Zhou, Qiang Gao, Goce Trajcevski, Kunpeng Zhang, Ting Zhong, and Fengli Zhang. 2018. Trajectory-user Linking via Variational Autoencoder. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence (IJCAI'18)*. AAAI Press, 3212–3218. http://dl.acm.org/citation.cfm?id=3304889.3305107