Geolet: An Interpretable Model for Trajectory Classification

Cristiano Landi[1], Francesco Spinnato[3,4], Riccardo Guidotti[2,4], Anna Monreale[2,4], and Mirco Nanni[4]

[1] University of Pisa, Italy, {name.surname}@phd.unipi.it,
[2] University of Pisa, Italy, {name.surname}@unipi.it,
[3] Scuola Normale Superiore, Pisa, Italy, name.surname@sns.it,
[4] ISTI-CNR, Pisa, Italy, {name.surname}@isti.cnr.it,

Abstract. The large and diverse availability of mobility data enables the development of predictive models capable of recognizing various types of movements. Through a variety of GPS devices, any moving entity, animal, person, or vehicle can generate spatio-temporal trajectories. This data is used to infer migration patterns, manage traffic in large cities, and monitor the spread and impact of diseases, all critical situations that necessitate a thorough understanding of the underlying problem. Researchers, businesses, and governments use mobility data to make decisions that affect people's lives in many ways, employing accurate but opaque deep learning models that are difficult to interpret from a human standpoint. To address these limitations, we propose Geolet, a human-interpretable machine-learning model for trajectory classification. We use discriminative sub-trajectories extracted from mobility data to turn trajectories into a simplified representation that can be used as input by any machine learning classifier. We test our approach against state-of-the-art competitors on real-world datasets. Geolet outperforms black-box models in terms of accuracy while being orders of magnitude faster than its interpretable competitors.

Keywords: Trajectory Classification · Interpretable Machine Learning · Mobility Data Analysis · Explainable AI

1 Introduction

The increasing diffusion of GPS-capable electronic devices, such as mobile phones, vehicles, and trackers, contributes to generating massive amounts of mobility data [9]. In general, any moving entity can generate spatio-temporal trajectories, which companies, governments, and researchers use to address many crucial applications [1]. Thus, mobility data affect the livelihoods of millions of people.

One of the most common tasks in this field is trajectory classification, i.e., predicting the class label of an object based on its movement [5,9,14]. Trajectory classifiers, for example, can differentiate between cars, taxis, buses, pedestrians, and bikes, recognize the movement of various animals, and infer people's jobs based on their routines. In [14] it is presented a survey comparing state-of-the-art trajectory classification approaches. The authors emphasize the main challenges in this field, namely the need for robust experimental evaluations across multiple datasets and the lack of advances in the state-of-the-art. Moreover, the majority of surveyed works are based on complex, black-box models such as Support Vector Machines (SVM), Multilayer Perceptrons (MLP), and deep Convolutional Neural Networks (CNN), which are inherently not interpretable from a human standpoint [7]. This can be a significant problem in high-stakes applications where the explanation aspect of machine learning models is critical for establishing trust in automated decision systems [11].

EXplainable Artificial Intelligence (XAI) for trajectories is an extremely under-explored topic in the literature. For this reason, we take inspiration from studies on XAI from time series [18], and specifically shapelets [22] to present the GEOgraphic ShapeLET classifier Geolet, an interpretable classification approach for trajectory data. First, Geolet uses geographic partitioning to segment the input data into subtrajectories. These subtrajectories are normalized and filtered in order to take only the most discriminative ones. They are then exploited to convert the input trajectories into a simplified, interpretable representation that can be used as input by any machine learning classifier. We evaluate Geolet on five datasets and against state-of-the-art alternatives, considering multiple quantitative metrics. Furthermore, we qualitatively show that the proposed approach produces interpretable and easy-to-read explanations.

2 Related Works

The problem of trajectory classification consists in building a predictive model from labeled historical trajectories to classify new ones [6,9]. Trajectory classifiers can be divided into different families. Classical approaches usually extract *global*, or *local* features from the data, whereas modern approaches tend to directly process the raw trajectories with complex, deep learning-based models.

Global features-based approaches extract features like velocity change, duration, speed, etc. from the whole trajectory [14]; they can be highly effective for simple datasets, where similar properties are maintained throughout the entire path. However, these methods are insufficient for more articulated trajectories in which the target class is linked to an event occurring in a trajectory portion.

Local features-based approaches try to mitigate these problems by segmenting the trajectory into subtrajectories and extracting features from them. In [21], the authors extract statistical features from the segments of the trajectory, first globally and then locally. Finally, they compare Random Forest, Gradient Boosting Decision Tree and XGBoost as classification models. In [20], it is proposed a semi-supervised clustering approach coupled with a majority voting ensemble classifier to learn a metric that brings similar data closer and distances elements with different labels. The first two methods can be viewed as pseudo-interpretable procedures, depending on the classification model used after the dataset transformation. A Random Forest, for example, can be used to determine the average importance of each variable. However, the main issue is that interpretability varies depending on the complexity of the extracted features and the number of weak learners in the ensemble. To the best of our knowledge, the only fully interpretable trajectory classifier is Movelets [5]. Indeed, the idea behind Movelets is to extract discriminative segments from the trajectory and, similarly to the shapelet transform for time series [22], convert the dataset into a new representation that stores the shortest distances between each trajectory and subtrajectory. In line with shapelets, subtrajectories can be used to understand the logic of the classifier [18]. While promising, the proposed method is computationally complex as it generates all possible subtrajectories and is not suitable for large datasets. Furthermore, only the space dimension is used to compute the distance between trajectories and subtrajectories. Thus, the trajectories must be resampled to constant time intervals, and only equal-length subtrajectories can be compared.

Recently, neural networks have been used in trajectory classification approaches to achieve superior performance in a faster manner. Recurrent Neural Networks (RNN) and Convolutional Neural Networks (CNN), often used with time series data, can be easily extended to trajectories. In [8], the authors propose TraClets, a CNN-based method that represents a trajectory as an image and uses a CNN to solve the trajectory classification task. *MARC* [13] deals with trajectories augmented with semantic textual dimensions, exploiting the GPS data and information in the textual dimensions. Finally, Rocket [4], the state-of-the-art classifier for multivariate time series, can be easily applied to trajectories to achieve fast and extremely accurate performance. Unfortunately, Rocket, RNN, and CNN models lead to a non-interpretable prediction. For this reason, several XAI approaches have been proposed to address the issue. Still, they can only output explanations as saliency maps [3, 15], highlighting the importance of each observation towards the classification.

Given the limitations of the literature, we propose a method for classifying trajectories based on local feature extraction. Geolet attempts to overcome the interpretability limitations of black-box models, and optimize accuracy and runtime, which is often the main problem of feature extraction-based methods.

3 Background and Problem Setting

In this section, we define all the concepts necessary to understand our proposal. We define a trajectory as follows:

Definition 1 (Trajectory)**.** A *trajectory* X is a sequence of spatio-temporal points $X = \{(\vec{\mathbf{x}}_1, t_1), \ldots, (\vec{\mathbf{x}}_m, t_m)\} \in \mathbb{R}^{m\times 3}$ where the spatial vectors $\vec{\mathbf{x}}_j = (\mathrm{lat}_j, \mathrm{long}_j)$ are sorted by increasing time t_j, i.e., $\forall 1 \leq j < m$ we have $t_j < t_{j+1}$.

In a sense, trajectories can be viewed as multivariate time series containing two signals, i.e., the latitude and longitude, recorded at non-constant sampling rates [5, 8, 19]. A trajectory classification dataset is a set of trajectories with a vector of labels attached. Formally:

Definition 2 (Trajectory Classification Dataset)**.** A *trajectory classification dataset* $\mathcal{D} = (\mathcal{X}, \mathbf{y}) \in \mathbb{R}^{n \times m \times 3}$ is a set of n trajectories, $\mathcal{X} = \{X_i \ldots, X_n\}$, with a vector of assigned labels (or classes), $\mathbf{y} = \{y_1, y_2, \ldots, y_n\} \in \mathbb{N}^n$.

For simplicity of notation, we use a single symbol m to denote the lengths of the trajectories, even if a trajectory dataset can contain instances having a different number of observations. We define the trajectory classification problem as follows:

Definition 3 (Trajectory Classification)**.** Given a trajectory classification dataset $\mathcal{D}$, *trajectory classification* is the task of training a function f from the space of possible inputs to a probability distribution over the class values in $\mathbf{y}$.

The resulting trajectory classification function f takes as input a trajectory X and returns y according to what f learned, i.e., $y = f(X)$. In general, y can either be a discrete label or the probability of X belonging to a specific class. Thus, given a trajectory classification dataset $\mathcal{D}$, our objective is to solve a trajectory classification problem by realizing an interpretable trajectory classification function f that allows to understand the reasons for a decision $y = f(X)$.
A fundamental aspect to introduce our proposal is the notion of *subtrajectory*:

Definition 4 (Subtrajectory)**.** Given a trajectory X of length m, a subtrajectory $S = \{(\vec{s}_j, t_j), \ldots, (\vec{s}_{j+l}, t_{j+l})\}$, of length $l \leq m$, is an ordered sequence of consecutive values such that $1 \leq j \leq m - l + 1$.

Subtrajectories can be used for classification purposes, similarly to shapelets, by *selecting the most discriminative ones* w.r.t. the target label, depending on some statistical measure. *Mutual Information* [17] is commonly used for classification purposes, measuring the dependency between continuous and discrete variables. Once the most discriminative subtrajectories are found, the dataset can be transformed in a simpler representation, via the subtrajectory transform. Formally:

Definition 5 (Subtrajectory Transform)**.** Given a trajectory dataset $\mathcal{X}$ and a set $\mathcal{S}$ containing h subtrajectories, the *Subtrajectory Transform* converts $\mathcal{X} \in \mathbb{R}^{n \times m \times 3}$ into a real-valued matrix $T \in \mathbb{R}^{n \times h}$, obtained by taking the *Best Fitting* of each trajectory $X \in \mathcal{X}$, and each subtrajectory $S \in \mathcal{S}$.

Usually, the best fitting of S in each X is computed by taking the minimum distance via a sliding window of length l. The most used distance functions to compare sequential data are the Euclidean distance and Dynamic Time Warping [2]. However, both have drawbacks when applied to trajectories. First, the Euclidean distance requires trajectories to have the same number of points, which is uncommon in real data. Secondly, both DTW and Euclidean distance implicitly need a constant sampling rate, which is not always guaranteed. For this reason, in our proposal, we adopt a distance specifically designed for trajectories, i.e., the *Interpolated Route Distance* (IRD) [19], which allows the comparison of trajectories having different lengths and sampling rates. IRD uses the *temporal* dimension to align two trajectories and, if two observations do not occur at the same time-step, values are projected by interpolating the information. In other

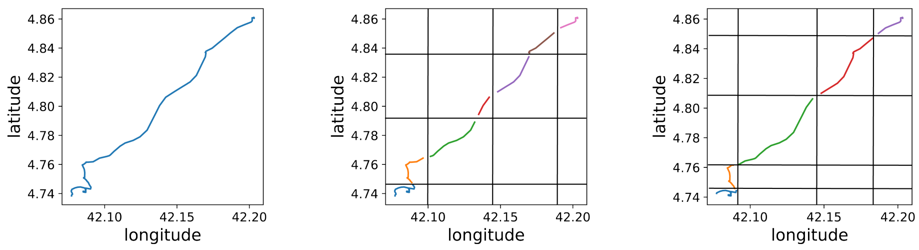

Fig. 1. Examples of partitioning. From left to right: original trajectory, Geohash, SAX.

words, given two trajectories, IRD calculates the distance between them for each timestamp. If a timestamp is not present in the other time series, IRD uses the neighboring timestamps to interpolate the values and estimate a position.

4 Geolet

This section presents the GEOgraphic ShapeLET classifier (Geolet), an interpretable classification approach for trajectory data. Geolet is our answer to the problem of designing an interpretable trajectory classification function f for a trajectory classification task. Geolet first *partitions* trajectories into multiple segments, yielding candidate subtrajectories. Secondly, it *normalizes* and *filters* them to produce a set of prototypical subtrajectories. Then, Geolet *transforms* the dataset using the Subtrajectory Transform. Finally, any interpretable classification model can be used to classify the transformed data.

Partitioning. Several approaches can be used to partition a trajectory: binning approaches like *Symbolic Aggregate Approximation (SAX)* [10], or geographical ones like *Geohash* [16] (Figure 1). SAX [10] is a discretization technique to convert time series into a sequence of symbols. It is usually applied by sliding window [12], creating a collection of SAX words that can be interpreted as time series subsequences. *We extend SAX to trajectories* by applying the approach independently to latitude and longitude signals, converting both into symbol sequences. In layman's terms, multiple coordinates in a trajectory are binned into a single symbol that represents an area. The converted signals are then used to generate a new symbol for each pair of observed symbols. The specific hyperparameters' configurations are detailed in Section 5. Another partitioning approach is *Geohash* [16], an indexing system encoding a rectangular geographic area into strings of letters and digits. Geohash divides the Earth into 32 regions via a bit array, associating each area with one of the symbols in [0-1a-z]. Then the process is repeated recursively until the algorithm reaches the desired accuracy.

Normalization. Following the partitioning phase, the segments must be normalized so that the domains of the various partitions overlap. This can be accomplished with a *Geohash normalization* or a *FirstPoint* normalization. In the Geohash normalization, the bottom-left vertex latitude and longitude of the Geohash cell are subtracted from the coordinates of each point of the subtrajectories. Intuitively, this is equivalent to overlapping each rectangle of the Geohash partitioning. On the other hand, in the FirstPoint normalization, the latitude and longitude of the first point of the subtrajectory are subtracted from the coordinates of each point of the subtrajectory. Intuitively, this is equivalent to overlapping the first point of each subtrajectory.

Filtering. After the partitioning and normalization phases, depending on the dimensionality of the data, we might end up with an enormous amount of subtrajectories. As a result, a filtering phase is carried out to reduce computational complexity and produce a smaller set of relevant subtrajectories. In this phase, subtrajectories are filtered by selecting a subset following some specific criterion. In the shapelet literature, these criteria can be unsupervised, such as random sampling and clustering, or supervised using statistical approaches, such as the Mutual Information or the Chi-squared test [12], that are used to find the subsequences that better discriminate between different classes. We experiment with both unsupervised and supervised approaches in Section 5.

Transform. Once a set of representative subtrajectories is found, the subtrajectory transform can be applied, transforming trajectories in a simpler representation, containing the Best Fitting (BF) between each trajectory in the original dataset and each extracted subtrajectory. For time series, the distance of choice is usually the Normalized Euclidean Distance (ED), however, as detailed in Section 3, this is not always the best choice for trajectories. Therefore, given a trajectory X of length m and a subtrajectory S of length l, the BF can be computed in different ways. For the Normalized Euclidean distance, a sliding window of size l is used to compare S with each subtrajectory of X. Formally,

$$BF_{ED}(X,S) = \min_{j=1}^{m-l+1} (ED(X_{j:j+l}, S))$$

where $X_{j:j+l}$ denotes a subtrajectory of X from j to $j+l$. On the other hand, defining the notion of best-fitting with DTW is not trivial. Indeed, using the same approach adopted for ED would limit the purpose for which DTW exists. Hence, we propose a similar approach, but where we use an expanded sliding window of length $l' > l$:

$$BF_{DTW}(X,S) = \min_{j=0}^{m-l'+1} (DTW(X_{j:j+l'}, S)).$$

Finally, since IRD exploits the time dimension to interpolate points when two time series do not have the same sampling rate, we calculate the sliding window size not w.r.t. the number of observations, but w.r.t. the time interval between the first observation of the subtrajectory S and its last timestamp t_l. Formally,

$$BF_{IRD}(X,S) = \min_{j=1}^{m-l+1} (IRD(X_{j:j+t_l}, S)).$$

This formula describes the calculation of the best fitting with IRD, using a sliding window where the length is defined not as the number of features but as the time interval. The sliding window length must correspond to the minimum number of

Table 1. Datasets description.

	`animals`	`vehicles`	`seabirds`	`geoLife`	`taxi`
# trajectories	102	381	108	5,977	121,312
avg length (std)	173 (56)	601 (230)	2,904 (1162)	8,100 (15178)	55 (22)
avg Δtime (std)	3.75 (6)	83 (596)	100 (0)	3.88 (12)	15 (0)
Δ time min-75%-max	0-4-258	0-30-53,857	100-100-100	0-2-665	15-15-15
target class (#classes)	species (3)	category (2)	species (3)	transport (2)	call type (3)

points necessary so that the trajectory time interval from j to $j + t_l$ is as close as possible to the subtrajectory's length.

Independently of the distance function adopted, the original dataset $\mathcal{X}$ is transformed in a simplified matrix representation T. We experiment both with continuous and discretized subtrajectories in Section 5. The transformed dataset T can be paired with any classification algorithm, having the advantage of a more interpretable data representation.

5 Experiments

We experiment with Geolet quantitatively on five datasets and we report visual examples to show the benefits of an interpretable-by-design trajectory classifier.

Datasets. The trajectory classification datasets are described in Table 1. For `animals` the task consists in recognizing different species. For `vehicles` we want to distinguish between buses and trucks. For `seabirds` the task is recognizing flying trajectories of three species of seabirds. For `geolife`, due to the high number of classes and to the unbalancing, we simplify the problem to recognizing trajectories of public vs private means of transport. Finally, for `taxi`, the objective is to distinguish among different types of taxi calls within one month of observations. We highlight that, state-of-the-art interpretable classification methods are experimented only on very small datasets like `animal` and `vehicles`. Each dataset is divided in train/test with a ratio 70/30%.

Competitors. We compare Geolet against two state-of-the-art methods, i.e., Movelets [5] and Rocket [4]. Movelets, similarly to Geolet, is an interpretable trajectory classifier that extracts discriminative subtrajectories and uses them to transform the dataset. The Movelets algorithm requires setting the minimum and maximum length of the generated subtrajectories. We use the default implementation values for `animals` and `vehicles`. Furthermore, we limit the maximum length to the logarithm of the number of maximum observations per trajectory for `seabirds`, `geolife`, and `taxi`. Rocket is a not interpretable time series classifier that transforms the dataset by applying random convolutional kernels to generate multiple feature maps that capture different data trends. The only hyperparameter to choose for Rocket is the number of convolutional kernels, which is set to $10,000$ as recommended by the authors [4].

Geolet Parameters Setting.[1]. For Geolet, we use Geohash as a partitioning algorithm, FirstPoint normalization, Mutual Information (as implemented by

[1] Code available at: github.com/cri98li/Geolet

`scikit-learn`) for filtering subtrajectories, and IRD as a distance measure. With this configuration, Geolet requires two hyperparameters: the Geohash precision (*prec*), and the number of subtrajectories to extract (*ns*). We set the optimal parameters via grid-search on the training set[2].

Geolet Alternative Implementations. For a fair benchmarking of Geolet, we devise some alternative versions that are still interpretable but extract explanations using different processes.

First, we compare the geographic-based segmentation of Geolet against a purely SAX-based approach. For this purpose, we apply the SAX approximation, as detailed in Section 4. We name this baseline MrSQM-T because, as part of the filtering phase, we adopt MrSQM [12], a time series approach that extracts the top symbolic subsequences using the Chi-squared test. MrSQM-T randomly generates k configurations of the triples l, w, α where l is the size of the sliding window, w is the SAX word length, and α is the alphabet size. MrSQM-T generates these sets using the same seed to guarantee that the previous configurations remain the same as k increases. The optimal value of k is set to 25 for `animals` and 11 for `vehicles`. In the experiments, we observe that MrSQM-T achieves good accuracy but requires a great computational effort, resulting in high runtimes, even for these relatively simple datasets.

Secondly, we aim at comparing the supervised subtrajectory selection of Geolet and MrSQM-T against an unsupervised one. For this purpose, we use a clustering approach to filter the extracted subtrajectories. Specifically, after Geohash segmentation and partitioning, prototypical subtrajectories are extracted through K-Medoid, using the Normalized Euclidean distance. Once the cluster centroids are extracted, they are compared using a sliding window to the original subsequences. Each trajectory is encoded with the identifier of the cluster centroids it contains. We name this baseline TrAC, Trajectory Approximation-based Classifier. TrAC requires four hyperparameters, i.e., the Geohash precision *prec*, the number of cluster k to use with K-Medoids (it also identifies the number of symbols in the alphabet), the sliding window length w, and the number of symbols subsequences top_{ss} to select based on the Mutual Information score. For each parameter, we performed a grid-search[3].

We highlight that, besides Geolet, MrSQM-T and TrAC are original contributions and do not exist in the literature as interpretable trajectory classifiers.

5.1 Classification Performance

Since Geolet, Movelets, Rocket, MrSQM-T and TrAC perform a transformation of the original data, any classification model can be applied to the transformed dataset. To compare the transformations fairly, we adopted the

[2] `animals`: $prec = 2$ $ns = 21$; `vehicles`: $prec = 6$ $ns = 20$; `seabirds`: $prec = 5$ $ns = 50$; `geolife`: $prec = 6$ $ns = 50$; `taxi`: $prec = 5$ $ns = 50$.

[3] $prec \in [4, 5, 6, 7]$; $k \in [2, 5, 20, 100]$; $w \in [2, 3, 5]$; $top_{ss} \in [1, 2, 10, 50]$ on the training set. Hyperparameter choice does not significantly affect the method's performance. We found constant accuracy values for most of the hyperparameters tested. There were, however, peaks in the accuracy score for some values. Thus, for `animals` we set $prec = 4, w = 3$ and $top_{ss} = 2$. For the `vehicles` $prec = 6, w = 3$ and $top_{ss} = 10$.

Table 2. Performance scores, best values in bold.

		`animals`	`vehicles`	`seabirds`	`geolife`	`taxi`
accuracy	Geolet	**0.935**	**0.965**	**0.967**	**0.861**	**0.578**
	Rocket	0.871	0.928	0.667	0.733	0.566
	Movelets	0.563	0.921	0.718	-	-
	MrSQM-T	0.677	0.887	-	-	-
	TrAC	0.742	0.791	-	-	-
runtime	Geolet	27.6s	50.1s	48m	2.42h	44m
	Rocket	**2.4s**	**31.5s**	**15.7s**	**29.1m**	**13.3m**
	Movelets	25.7s	141m	126.9s	-	-
	MrSQM-T	22.5m	1.16h	-	-	-
	TrAC	25.4s	1.18h	-	-	-

same effective model for all five approaches, i.e., a Random Forest classifier as implemented by the `scikit-learn` library. The best hyperparameters are found via grid-search with 10-fold cross-validation[4] on the training set.

Results in terms of accuracy and runtime are reported in Table 2[5]. We measure the execution time of each algorithm from the data preparation phase to the end of the dataset transformation. Hence, we exclude the time for training the final model. From a first glance, we can see that Rocket is the method that takes the least time to execute. As for Movelets, we performed several attempts with the `geolife` and `taxi`, but all the tests ended with an "insufficient memory error". In addition, we recorded anomalous results with the `animals`, which we suspect was due to a bug in the original code. As for Geolet, we can see that it manages to get the best results between these two methods, but it takes a longer execution time. The weakness of Geolet compared to Rocket and Movelets lies in the number of hyperparameters and configurations from which one can choose, which is discussed in Section 5.3. TrAC and MrSQM-T perform competitively w.r.t. Movelets in small datasets, but are both outperformed by Geolet and Rocket. Moreover, due to their high computational cost, they are hardly usable when dealing with real-world datasets.

5.2 Geolet Interpretability

This section provides an example of the kind of interpretable classification that Geolet can provide. We apply Geolet on `vehicles` with $prec = 4$ and Geohash as partitioning method, FirstPoint normalization, Normalized Euclidean distance and Mutual Information for the transform. Finally, we use a Decision Tree as a classification model as implemented by `scikit-learn`, which allows us to visualize the resulting model graphically and extract rules summarizing its decision boundaries. In particular, for `vehicles`, we identified the following rules:

$r_1 = \{dist(X, S_4)\ is\ low\ \wedge\ dist(X, S_0)\ is\ low\} \rightarrow Bus$

$r_2 = \{dist(X, S_4)\ is\ low\ \wedge\ dist(X, S_0)\ is\ high\} \rightarrow Truck$

[4] n_estimators=range(300, 1500, 300), criterion=[gini, entropy], max_depth=range(2, 20, 3)

[5] Tests are performed on a machine with CPU: AMD Ryzen 9 3900X; RAM: 32GB; OS: EndeavourOS Linux. Due to resource limitations, we used 20% of `geolife` and 70% of `taxi`.

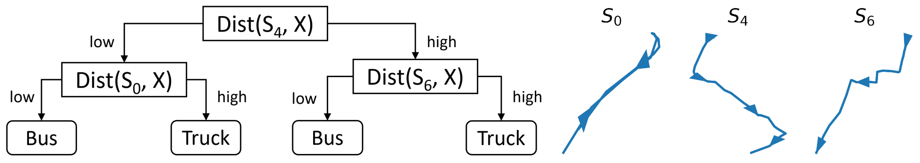

Fig. 2. GEOLET's Decision Tree for `vehicles` (left) and subtrajectories used (right).

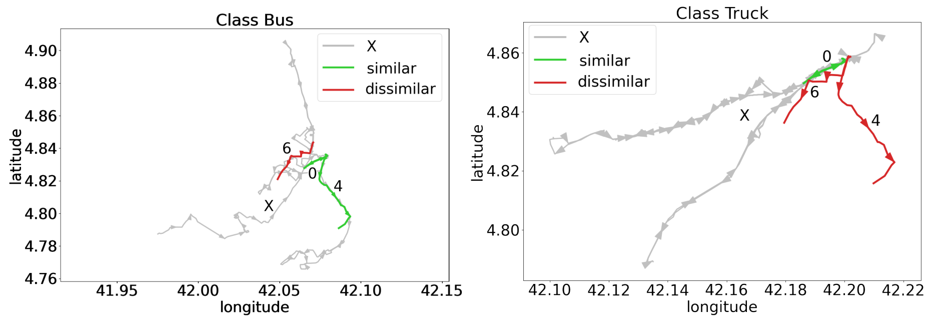

Fig. 3. Examples of the GEOLET explanation on two instances from `vehicles`. Left: instance of the class *Bus*. Right: instance of the class *Truck*.

$$r_3 = \{dist(X, S_4)\ is\ high\ \wedge\ dist(X, S_6)\ is\ low\} \rightarrow Bus$$
$$r_4 = \{dist(X, S_4)\ is\ high\ \wedge\ dist(X, S_6)\ is\ high\} \rightarrow Truck$$

We highlight that, to ease the understanding, we report "is high" / "is low" instead of the real distance because it is sufficient to understand the meaning of the rule without accounting for the specific threshold numbers. Specifically, "low" indicates that the distance measurement is below the split threshold value, and "high" indicates that the value exceeds it For instance, $dist(X, S_4) \leq 0.3$ is translated into $dist(X, S_4)$ *is low*. The decision tree and the subtrajectories are illustrated in Figure 2. These rules show that the most representative subtrajectories are those with indices 0, 4, and 6. We can now understand the decisions of the classifier by visualizing where the subtrajectories fit within the trajectory. Figure 3 presents the classification of GEOLET for two instances. In particular, the instance belonging to the class Bus has segments very similar to subtrajectories 0 and 4, and are instead quite different from subtrajectory 6. On the other hand, the Truck instance contains almost perfectly the subtrajectory 0, but it is quite different from 4 and 6.

5.3 Geolet Parameters Sensitivity

In general, it is extremely difficult to define a global heuristic for this approach. For this reason, we describe here our implementation choices and analyze how hyperparameters selection affects the GEOLET's results on `animals` and `vehicles`, providing some practical insights and guidelines.

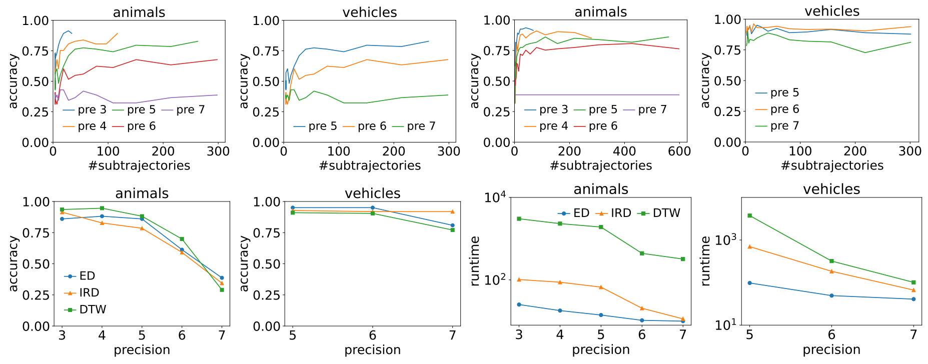

Fig. 4. Top: accuracy of GEOLET varying the number of subtrajectories. Bottom: comparison of the distance measures in terms of accuracy (left) and runtime (right). The first two columns are the results obtained using GEOHASH-RND, and the last two columns are the results obtained using GEOHASH-MIG.

Partitioning. In Figure 4 we study how Geohash precision and number of subtrajectories affect accuracy. Also, we determine the importance of selecting trajectory using a well-founded criterion such as the Mutual Information Gain (GEOHASH-MIG), instead of simply selecting them randomly (GEOHASH-RND). IRD is used as distance, and FirstPoint is used as the normalization strategy. From the results, we can observe that, although random selection (GEOHASH-RND) leads to a worse result, it could be a great way to quickly determine the best precision for Geohash partitioning. The average runtime of GEOHASH-RND compared to GEOHASH-MIG turns out to be 13 times faster for `animals` and two times faster for `vehicles`. On the other hand, by selecting subtrajectories using Mutual Information (GEOHASH-MIG), we can achieve better results faster and with fewer subtrajectories. Regarding `animals`, we note that increasing the length of the subtrajectories improves the results.

Normalization. We study here the impact of using different normalization techniques, i.e., Geohash (GEOLET-GH) and FirstPoint (GEOLET-FP). Our experiments show that the accuracy of GEOLET-GH is 0.677 for `animals` and 0.791 for `vehicles`, while for GEOLET-FP is 0.935 for `animals` and 0.965 for `vehicles`. Therefore, we select FirstPoint as normalization for GEOLET.

Distance. Finally, we analyze the impact of different distance metrics on performance. Figure 4 (bottom) shows that the best distance for `animals` is DTW, while the best distance for `vehicles` is ED. However, when the computation time for the dataset transformation is considered, it is clear that larger datasets cannot use DTW. Thus, excluding DTW, IRD has the best accuracy score for `animals`, while it performs negligibly worse than ED for `vehicles`. As a result, our intuition is that: *(i)* for small datasets, the DTW is the best distance, *(ii)* for large datasets with consistent sample rates, ED is the best choice, while *(iii)* for large datasets with variegated sample rates, IRD is the best compromise between accuracy and runtime.

In summary, the most sensible hyperparameter of Geolet is the precision.

6 Conclusion

We have presented Geolet, an interpretable classifier for trajectory data. Geolet is able to transform trajectory data into a simplified representation that any classifier can use as an interpretable input source. We have shown that Geolet outperforms state-of-the-art competitors in terms of accuracy while remaining competitive in terms of runtime. Besides, Geolet is interpretable, returning subtrajectory-based explanations that are easily interpretable from a human standpoint. As future research directions, we intend to improve Geolet's performance in terms of accuracy, runtime, and explainability. In this sense, many extensions are possible. Subtrajectories, can be improved by embedding properties such as scale and rotation invariance, resulting in a smaller set of prototypical and interpretable subsequences. Also, Geolet can be extended to work with data that includes additional features like height and semantic textual dimensions, as well as data that uses different coordinate systems. To accomplish this, the modularity of the implementation can be used to introduce new distance measures, filtering approaches, normalization techniques, and partitioning methods. Finally, we want to investigate the regression and forecasting tasks, which are fundamental in this field but remain unexplored from an XAI standpoint.

Acknowledgment. This work is partially supported by the European Union NextGenerationEU programme under the fuding schemes PNRR-PE-AI scheme (M4C2, investment 1.3, line on Artificial Intelligence) FAIR (Future Artificial Intelligence Research), and "SoBigData.it - Strengthening the Italian RI for Social Mining and Big Data Analytics" - Prot. IR0000013. This work is partially supported by the European Community H2020 programme under the funding schemes: H2020-INFRAIA-2019-1: Res. Infr. G.A. 871042 *SoBigData++*, G.A. 761758 *Humane AI*, G.A. 952215 *TAILOR*, ERC-2018-ADG G.A. 834756 *XAI*, and CHIST-ERA-19-XAI-010 SAI, and by the Green.Dat.AI Horizon Europe research and innovation programme under the G.A. 101070416.

References

1. Andrienko, G.L., Andrienko, N.V., Boldrini, C., Caldarelli, G., Cintia, P., Cresci, S., Facchini, A., Giannotti, F., Gionis, A., Guidotti, R., Mathioudakis, M., Muntean, C.I., Pappalardo, L., Pedreschi, D., Pournaras, E., Pratesi, F., Tesconi, M., Trasarti, R.: (so) big data and the transformation of the city. Int. J. Data Sci. Anal. **11**(4), 311–340 (2021)
2. Bellman, R., Kalaba, R.: On adaptive control processes. IRE Transactions on Automatic Control **4**(2), 1–9 (1959)
3. Bodria, F., Giannotti, F., Guidotti, R., Naretto, F., Pedreschi, D., Rinzivillo, S.: Benchmarking and survey of explanation methods for black box models. CoRR **abs/2102.13076** (2021)

4. Dempster, A., Petitjean, F., Webb, G.I.: ROCKET: exceptionally fast and accurate time series classification using random convolutional kernels. Data Min. Knowl. Discov. **34**(5), 1454–1495 (2020)
5. Ferrero, C.A., Alvares, L.O., Zalewski, W., Bogorny, V.: MOVELETS: exploring relevant subtrajectories for robust trajectory classification. In: SAC. pp. 849–856. ACM (2018)
6. de Freitas, N.C.A., da Silva, T.L.C., de Macêdo, J.A.F., Junior, L.M.: Using deep learning for trajectory classification in imbalanced dataset. In: FLAIRS Conference (2021)
7. Guidotti, R., Monreale, A., Ruggieri, S., Turini, F., Giannotti, F., Pedreschi, D.: A survey of methods for explaining black box models. ACM computing surveys (CSUR) **51**(5), 1–42 (2018)
8. Kontopoulos, I., Makris, A., Tserpes, K., Bogorny, V.: Traclets: Harnessing the power of computer vision for trajectory classification (2022)
9. Lee, J., Han, J., Li, X., Gonzalez, H.: *TraClass*: trajectory classification using hierarchical region-based and trajectory-based clustering. Proc. VLDB Endow. **1**(1), 1081–1094 (2008)
10. Lin, J., Keogh, E.J., Lonardi, S., Chiu, B.Y.: A symbolic representation of time series, with implications for streaming algorithms. In: DMKD. pp. 2–11. ACM (2003)
11. Miller, T.: Explanation in artificial intelligence: Insights from the social sciences. Artif. Intell. **267**, 1–38 (2019)
12. Nguyen, T.L., Ifrim, G.: Mrsqm: Fast time series classification with symbolic representations. CoRR **abs/2109.01036** (2021), `https://arxiv.org/abs/2109.01036`
13. Petry, L.M., da Silva, C.L., Esuli, A., Renso, C., Bogorny, V.: MARC: a robust method for multiple-aspect trajectory classification via space, time, and semantic embeddings. Int. J. Geogr. Inf. Sci. **34**(7), 1428–1450 (2020)
14. da Silva, C.L., Petry, L.M., Bogorny, V.: A survey and comparison of trajectory classification methods. In: BRACIS. pp. 788–793. IEEE (2019)
15. Simonyan, K., Vedaldi, A., Zisserman, A.: Deep inside convolutional networks: Visualising image classification models and saliency maps (2014)
16. Suwardi, I.S., Dharma, D., Satya, D.P., Lestari, D.P.: Geohash index based spatial data model for corporate. In: 2015 International Conference on Electrical Engineering and Informatics (ICEEI). pp. 478–483. IEEE (2015)
17. Tan, P.N., Steinbach, M.S., Kumar, V.: Introduction to data mining. Pearson Education India (2016)
18. Theissler, A., Spinnato, F., Schlegel, U., Guidotti, R.: Explainable AI for time series classification: A review, taxonomy and research directions. IEEE Access **10**, 100700–100724 (2022)
19. Trasarti, R., Guidotti, R., Monreale, A., Giannotti, F.: Myway: Location prediction via mobility profiling. Inf. Syst. **64**, 350–367 (2017)
20. Vouros, A., Gehring, T.V., Szydlowska, K., Janusz, A., Tu, Z., Croucher, M., Lukasiuk, K., Konopka, W., Sandi, C., Vasilaki, E.: A generalised framework for detailed classification of swimming paths inside the morris water maze. Scientific reports **8**(1), 1–15 (2018)
21. Xiao, Z., Wang, Y., Fu, K., Wu, F.: Identifying different transportation modes from trajectory data using tree-based ensemble classifiers. ISPRS Int. J. Geo Inf. **6**(2), 57 (2017)
22. Ye, L., Keogh, E.J.: Time series shapelets: a new primitive for data mining. In: KDD. pp. 947–956. ACM (2009)