Scooter Trajectories Clustering

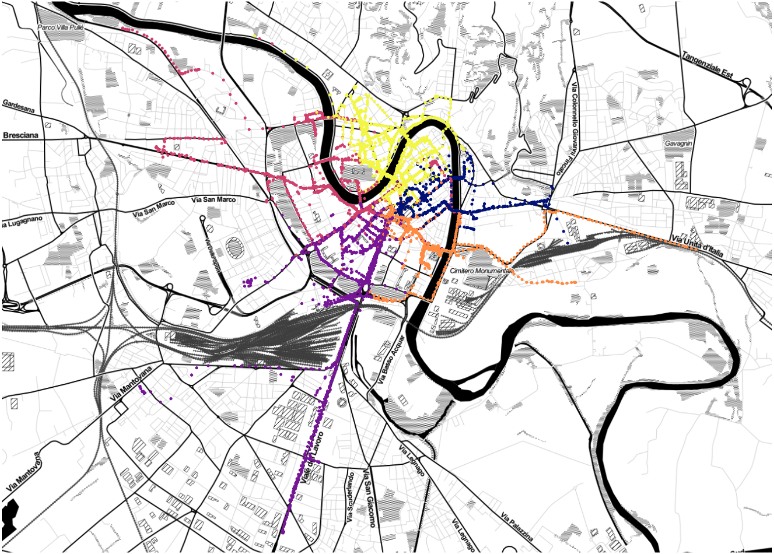
Machine Learning and Deep Learning

Mirco De Marchi VR445319

University of Verona

2020/2021

VERONA!!!



Parco Villa Pullè
Gardesana
Bresciana
Via Lugagnano
Via San Marco
Via San Marco
Cimitero Monumentale
Via Colonnello Giovanni Fincato
Tangenziale Est
Gavagnin
Via Unità d'Italia
Via Mantovana
Via Mantovana
Viale del Lavoro
Via Scuderlando
Via San Giacomo
Via Basso Acquar
Via Legnago
Via Legnago
Via Palazzina

Introduction

Motivation

Trajectory clustering is a problem really difficult to be treated but can be useful for several applications:

- Monitoring
- Forecasting
- Viability
- Smart City
- Security

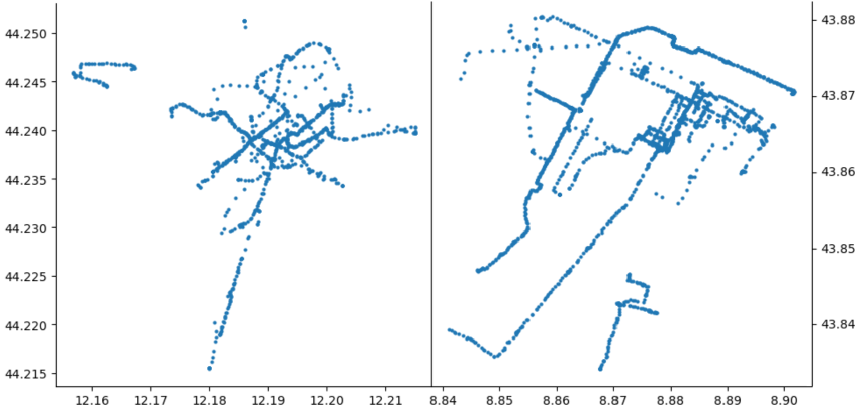
Machine Learning:

- Spatial based clustering: *DBSCAN* algorithm.
- Time depended clustering: *OPTICS* algorithm.
- Partition and group based clustering: *Lee partition & group*.
- Uncertain trajectory clustering: *Fuzzy C-Means* algorithm.
- Semantic trajectory clustering: *Stops and Moves* model.

Deep Learning:

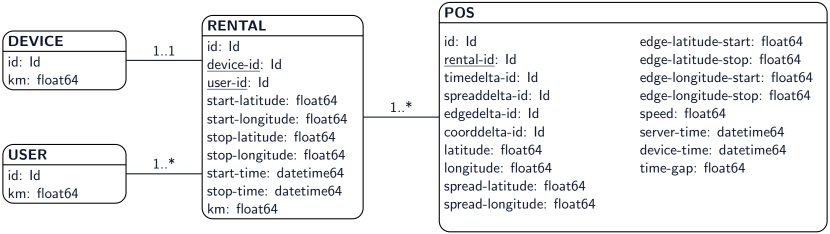
- Autoencoder: DCEC, DETECT...
- Motion Pattern Approach.
- Deep SOM: DPSOM.

Dataset size: **2GB**.



Dataset diagram

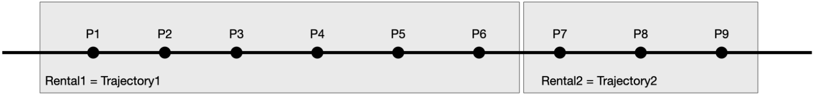
Dataset	Samples	Features
rental	14826	10
pos	817076	18
merge	817076	18
dataset	14826	13
partition city 1	608251	18
partition city 2	202795	18



Methodology

A trajectory is a set of positions that belong to the same rental sorted by the timestamp.

$$trajectory(rental\_id) = \{p \mid p.rental\_id == rental\_id\} \qquad (1)$$



Each position p is a tuple $(t_p, lat_p lon_p, s_p)$.

Rentals trajectories

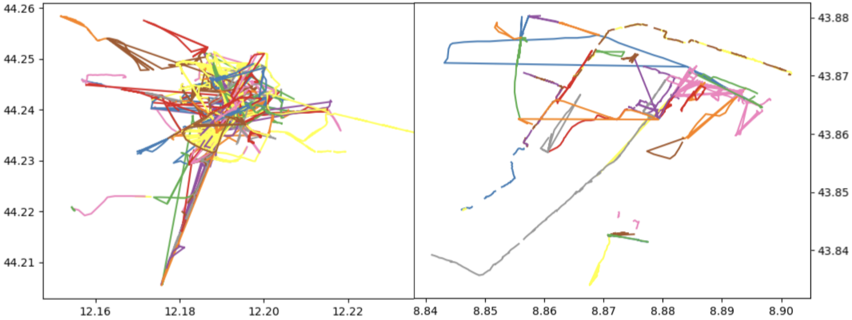


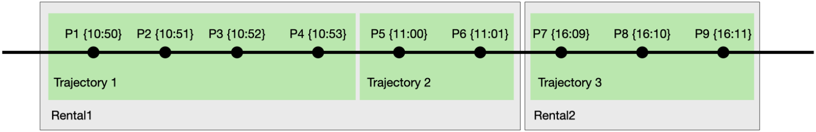
Figure: Rentals showed in the 2 cities: 200 (left), 50 (right).

Heuristics: *timedelta*

The following heuristics methodologies use a `delta` value that is valued with the statistic's empirical rule.

- **timedelta heuristic**: a rental trajectory can be divided in a sequence of trajectories if the time gap between a position and previous one exceeds a *timedelta* value.

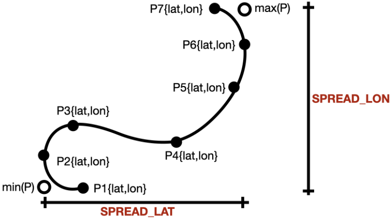
$$TIMEGAPS = \{t.time - shift(t, -1).time \mid \forall t \in TRAJ\} \quad (2)$$

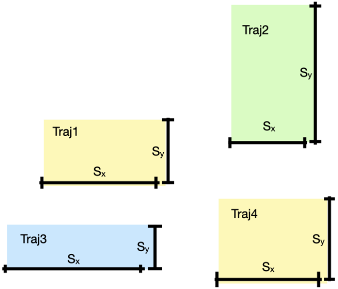


HEURISTICS: *spreaddelta*

- **spreaddelta heuristic**: a rental trajectory is similar to another one if they spread a similar amount of area in relation with *spreaddelta* value.

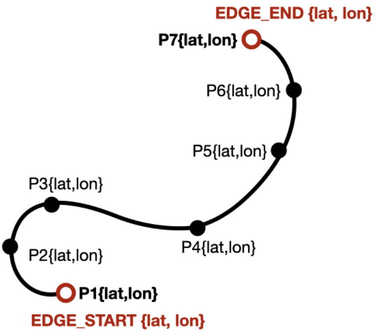
$$SPREADS = \{max(t) - min(t) \mid \forall t \in TRAJ\} \qquad (3)$$

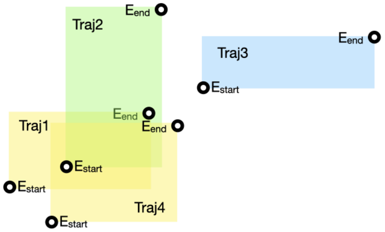




- **edgedelta heuristic**: acts as the *spreaddelta heuristic*, but it considers the edges of a trajectory, or rather the first position and the last position of a trajectory in relation with *edgedelta* value.

$$EDGES = \{concat(p[0], p[-1]) \mid \forall t \in TRAJ\} \tag{4}$$





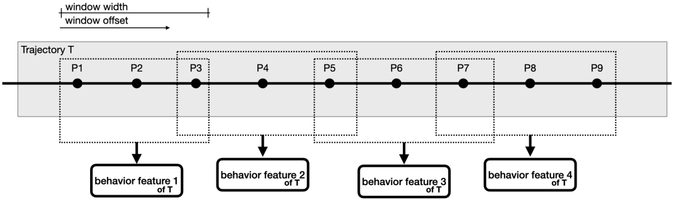
Pipeline: integration of heuristic data as features, *Standardization*, *Normalization* and than *Principal Component Analysis (PCA)*.

The component extracted by *PCA* can be decided in 3 different ways:

- By a number a priori;
- By the cumulative variance with 80% cover;
- Concatenation of columns produced by *PCA* for different subset of features;

$$\{\{latitude\}, \{longitude\}, \{spread\,latitude, spread\,longitude\},$$
$$\{edge\,latitude\,start, edge\,latitude\,stop,$$
$$edge\,longitude\,start, edge\,longitude\,stop\}\}$$

Obtain space- and time- invariant features to describe the moving behaviors of the object with a sliding window.



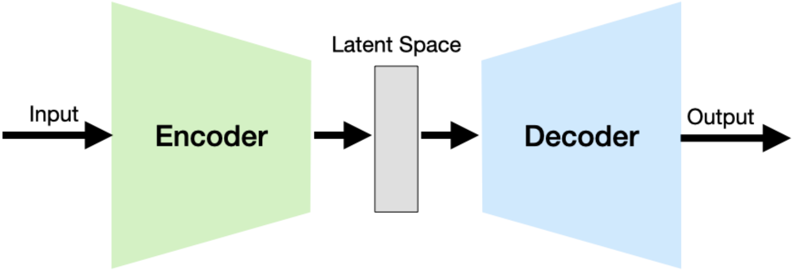
$$f_{\Delta lat_i} = \Delta lat_i / \Delta t_i \qquad f_{\Delta lon_i} = \Delta lon_i / \Delta t_i$$

$$f_{\Delta s_i} = \Delta s_i \qquad f_{\Delta r_i} = \Delta r_i$$

with statistics: $mean, max, 75\%quantile, 50\%quantile, 25\%quantile, min$

- **K-Means**: simple technique with distance based metric, fast and cheap in memory terms. $O(n * k * l)$
- **Mean Shift**: density based, automatically sets the number of clusters, but it needs a bandwidth parameter. $O(n^2)$
- **Gaussian Mixture**: estimation of linear combination of a finite number of Gaussian distributions with unknown parameters and *expectation-maximization (EM)* algorithm. $O(l * n^3)$
- **Full Hierarchy Agglomerative**: hierarchical clustering with bottom up approach and minimization metric on the maximum distance between observations in pairs of clusters. $O(n^3)$
- **Ward Hierarchy Agglomerative**: hierarchical clustering with bottom up approach and minimization metric on the sum of squared differences between all clusters. $O(n^3)$

Autoencoder



The autoencoder is trained to reproduce the input in the output.

- *Simple Autoencoder*: the model is composed by two LSTM that which acts as encoder and decoder.
- *Autoregressive Autoencoder*: the LSTM decoder takes the output of the current step as input for the following step.
- *Addons Autoencoder*: already implemented decoder contained in *TensorFlow Addons* library.

The training can be performed creating a dataset of sliding windows over the input timeseries trajectory.

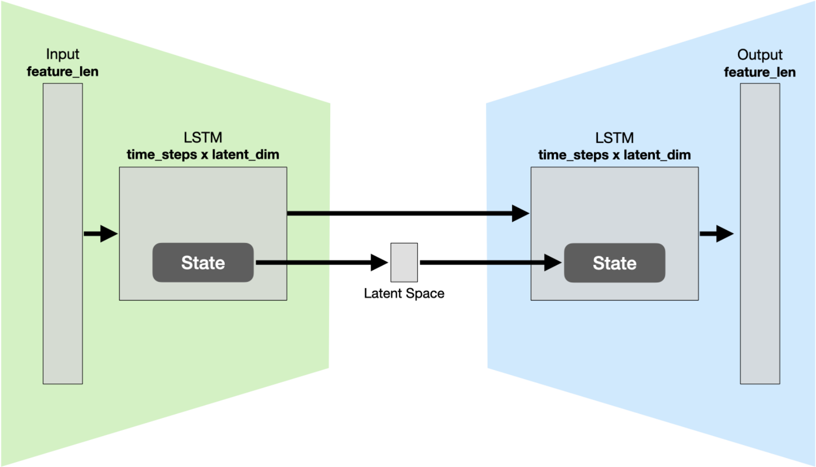


Figure: Simple Autoencoder schema

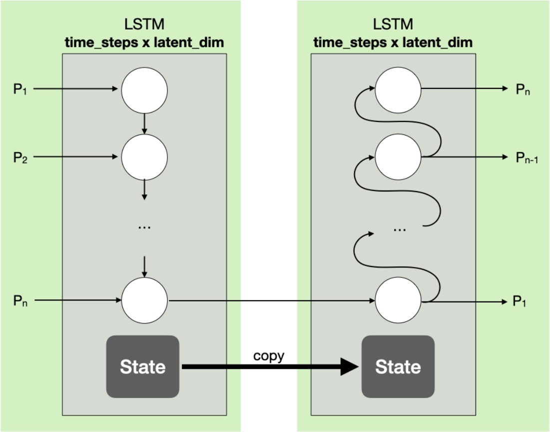
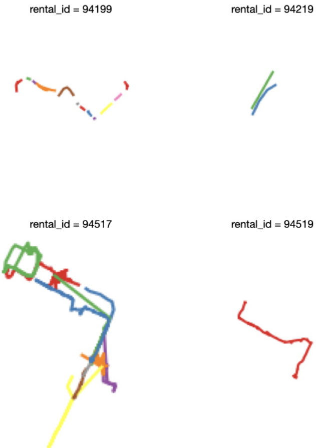


Figure: Autoregressive Autoencoder schema

Results

Timedelta heuristic results



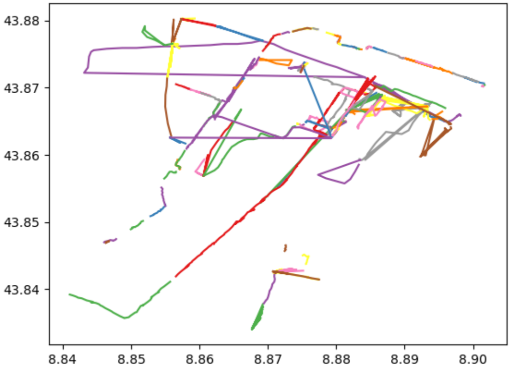


Figure: Rentals showed: 50.

Spreaddelta heuristic results

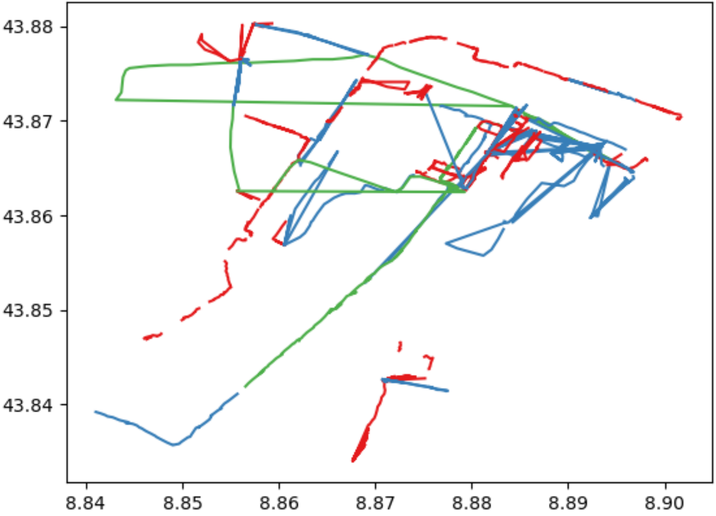


Figure: Rentals showed: 50.

Edgedelta heuristic results

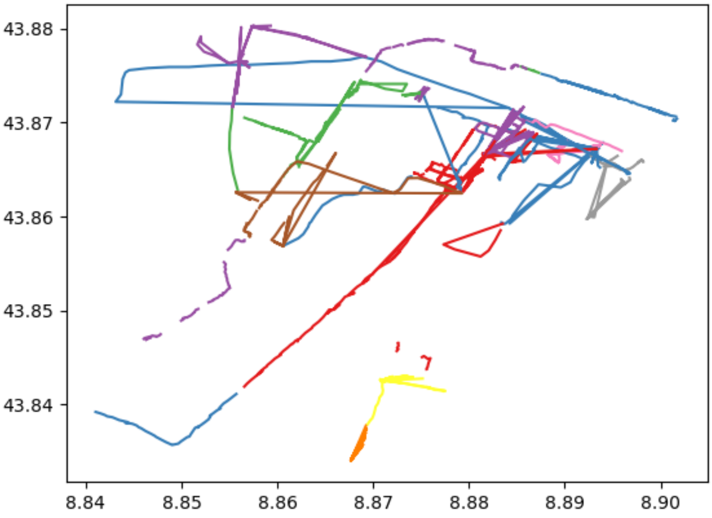
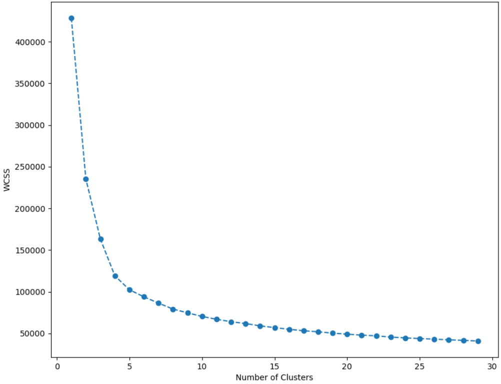


Figure: Rentals showed: 50.

WCSS and Elbow method

Within Cluster Sum of Squares (WCSS) graph for *Elbow method* with number of clusters in range from 1 to 30 and *K-Means* algorithm.



Number of clusters estimated: **5**.

Ward Hierarchical Agglomerative

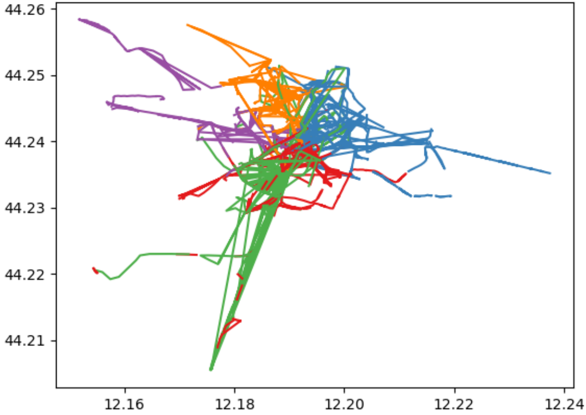


Figure: Silhouette: 0.28. Rentals showed: 200.

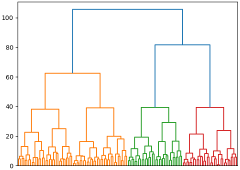


Figure: Dendrogram up to level 5 of merge

K-Means clustering

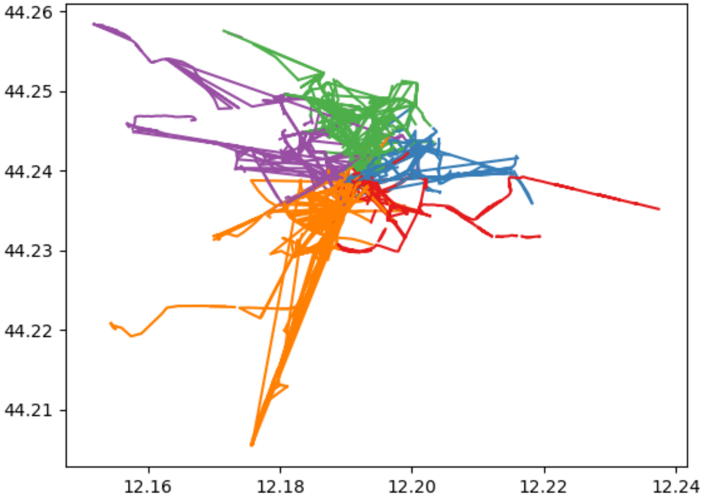


Figure: Silhouette: 0.352. Rentals showed: 200.

K-Means applied to the latent space extracted.

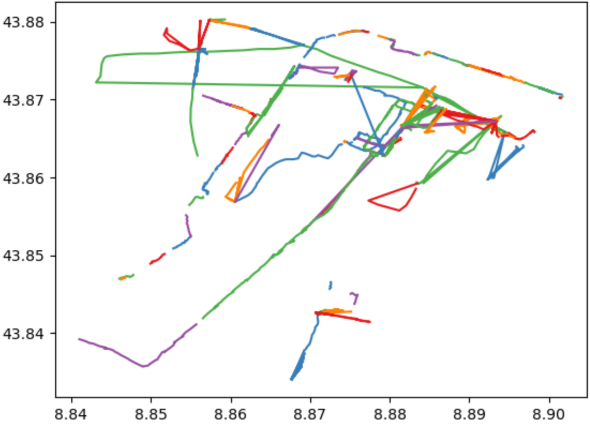


Figure: Silhouette: 0.112. Positions showed: ~1000.

Conclusion

Final considerations

- *K-Means* is the best result in terms of plot representation and *Silhouette score*;
- Time useful for partition and group, but not for bottom-up clustering techniques;
- Clustering with *PCA* shows better results in variance terms;
- Clustering with heuristic features maintains the rental information;
- Clustering has always to be performed on a specific region of interest in order to optimize the results;
- *Silhouette score* is not a validation methodology so reliable, because it depends a lot on the data you are dealing with;
- Deep Clustering doesn't perform well, but can be improved...

Improvements

- Test the autoencoder models on well known datasets;
- Test the autoencoder models with more epoch;
- Design more complex autoencoder models (with convolutional layer);
- Trying to augment the number of positions in a trajectory with some techniques;
- Improve the edge and coord heuristic applying Mean Shift, in order to avoid the bimodal distribution problem;
- Improve the autoencoder training with specific loss functions focused on clustering;

K-Means 3D