POLITECNICO MILANO 1863

SCUOLA DI INGEGNERIA INDUSTRIALE E DELL'INFORMAZIONE

Executive Summary of the Thesis

Graphs of Objects: Innovating Data Modelling for Road Pavement Detection

Laurea Magistrale in Mathematical Engineering - Ingegneria Matematica

Author: Marco Poggi

Advisor: Prof. Simone Vantini

Co-advisor: Arianna Burzacchi

Academic year: 2022-2023

Introduction

A common issue in developing countries is the inefficiency and unreliability of infrastructures.
A concerning example of this is the fact that most of the roads in urban areas are paved without proper documentation. As a consequence of this, the optimization of public transport routes is hindered by the lack of knowledge regarding which roads are actually accessible and which are not.
The objective of this work was to solve this issue by producing new algorithms capable of performing the classification of road segments into "paved" and "unpaved", relying only on publicly available data.
We focused on the case of the Greater Maputo area, in Mozambique, but the results can be promptly extended to any other Sub-Saharan city.

1. Graphs of Objects

The data used for our analysis came from two different public sources, OpenStreetMap and Google Earth, respectively in the form of a network of polygons identified by spatial coordinates and a set of satellite images. In this way, for each road we had access to its shape, location, and appearance. In order to fully exploit the available information, without having to rely on synthetic indices, we decided to design a new data structure, that maintained this duality.
Firstly, inspired by the work in [1], we converted every satellite image into a 3D object. In practice, every picture was cleaned of unwanted areas (generally representing vegetation or vehicles), and a set of 150 of the remaining pixels was randomly sampled and used to identify an object ω_i in the RGB cube. The set of all objects was called Ω.
Secondly, a set of unweighted undirected edges was designed starting from the polygon network. Every pair of road segments whose respective polygons had a common boundary where assigned an edge η_{ij}. The set of all edges was called H.
These steps lead to the creation of a data structure called Graph of Objects (GoO), and identified as

$$\Gamma(\Omega, \mathrm{H})$$

where nodes contained the information that was included in the pictures, while edges incorporated knowledge coming from the polygons.
Because of the large dimension of the dataset,

we also extracted a second GoO,

$$\Gamma'(\Omega', \mathrm{H}')$$

representing a limited coastal area of Maputo. Γ' allowed us to tune our models with much shorter computational times, to find the optimal hyperparameters, and finally to run only the best models on Γ.

2. Classification Framework

After having produced a data structure that incorporated all the information we wanted to use, it was time to decide how to perform the classification of the unlabeled objects. This task was a clear example of Semi-Supervised Learning. In this framework, both labeled and unlabeled data were needed to improve accuracy, propagating information from the former to the latter throughout the graph structure.
First of all, we decided that our algorithms needed to account for uncertainty at a certain extent. Hence, they were written so that the predicted label $\hat{y}_i$ of object ω_i could assume three different values: 0 ("paved"), 1 ("unpaved"), or 2 ("uncertain"). This lead to the type of confusion matrix in Table 1.

		Predicted		
		0	1	2
True	0	N_{00}	N_{01}	N_{02}
	1	N_{10}	N_{11}	N_{12}

Table 1: Confusion matrix in the general case

Inspired once again by [1], we defined the misclassification costs in Table 2, where y_i refers to the true class of ω_i and $\hat{y}_i$ to its predicted class. Classifying an unpaved road as paved was

Misclassification	Cost
$y_i = 0, \hat{y}_i = 1$	2
$y_i = 0, \hat{y}_i = 2$	1
$y_i = 1, \hat{y}_i = 0$	2.5
$y_i = 1, \hat{y}_i = 2$	1

Table 2: Misclassification costs

considered more dangerous than vice versa. Furthermore, uncertainty was deemed less problematic than wrong predictions.
This allowed for the computation of an error, that was then normalized w.r.t. the error of a dummy classifier in order to obtain a percentage:

$$\mathcal{E} = \frac{2N_{01} + N_{02} + 2.5N_{10} + N_{12}}{\mathrm{err}_{\mathrm{dummy}}} \cdot 100\% \quad (1)$$

The objective of our GoO-based algorithms was therefore to minimize (1).
We then selected a traditional statistical approach, κ-Nearest Neighbors (κ-NN), and a model based on Deep Learning, a Graph Convolutional Network (GCN), and adapted them to this established framework.

3. κ-NN

3.1. Theory

In [1] an Object-Oriented κ-NN was designed, where distance between objects was chosen to be the energy distance. Keeping this in mind, we stored the energy distance between every $\omega_i \in \Omega$ and $\omega_j \in \Omega_{\mathcal{L}}$, where $\Omega_{\mathcal{L}}$ was the subset of labeled objects, in a matrix E. We then normalized it, computing the so-called energy dissimilarity matrix:

$$\Delta_{E,ij} = \frac{E_{ij} - \min E}{\max E - \min E} \quad (2)$$

Afterwards, we computed multiple other dissimilarity matrices to include information contained in H, rather than just Ω.
The first one was the adjacency dissimilarity matrix:

$$\Delta_{A,ij} = 1 - A_{ij} \quad (3)$$

where A was the adjacency matrix of the GoO. Then, starting from A, we computed P, a matrix consisting of the length of the shortest path between each pair $\omega_i \in \Omega$ and $\omega_j \in \Omega_{\mathcal{L}}$. For comparison purposes we normalized this one too:

$$\Delta_{P,ij} = \frac{P_{ij} - \min P}{\max P - \min P} \quad (4)$$

Finally, we performed four different community detection algorithms over Γ. These algorithms were capable of finding hidden structures within the graph, similarly to what a clustering algorithm does, starting only from the adjacency matrix.
The idea was that communities within the network were likely reflecting neighborhoods of the city, and the pavement of roads in the same neighborhood was likely to be similar.

We then counted how many of these four algorithms classified each pair of ω_i and ω_j as members of the same community in matrix C:

$$C_{ij} = \sum_{a \in \mathcal{A}} \delta_a(\omega_i, \omega_j) \tag{5}$$

where $\mathcal{A}$ was the set of the four community detection algorithms. We finally normalized C, obtaining the community dissimilarity matrix:

$$\Delta_{C,ij} = \frac{\max C - C_{ij}}{\max C - \min C} \tag{6}$$

Multiple tests were made to find the best combination of sums and element-wise products of these four dissimilarity matrices to find the best performing one. It was proven that this optimal dissimilarity was given by:

$$\Delta_{ij} = \Delta_{E,ij}^{\zeta} \cdot \Delta_{C,ij}^{1-\zeta} \tag{7}$$

where $\zeta \in (0,1)$ was an hyperparameter to be tuned.
Finally, we decided to compare and contrast two models:

- **$\Omega\kappa$-NN**, the Object-Oriented κ-NN proposed in [1], using (2) as dissimilarity;
- **$\Gamma\kappa$-NN**, a new GoO-Oriented κ-NN, using (7) as dissimilarity.

3.2. Results

In the final part of our work on the κ-NN, we tried to find the optimal sets of hyperparameter values both for the Object-Oriented model ($\Omega\kappa$-NN), and the new GoO-Oriented one ($\Gamma\kappa$-NN). These hyperparameters where:

- κ: number of neighbors;
- θ_1 and θ_2: thresholds such that:

$$\hat{y}_i = \begin{cases} 0 & \text{if } f_{\mathcal{P},i} > \theta_2 \\ 1 & \text{if } f_{\mathcal{P},i} \leq \theta_1 \\ 2 & \text{otherwise} \end{cases} \tag{8}$$

  where $f_{\mathcal{P},i}$ was the frequency of paved neighbors of ω_i;
- ζ (only for $\Gamma\kappa$-NN): value in (7).

We decided to tune these hyperparameters with 5-fold cross-validation over multiple different train-test splits of Γ'. This process allowed to iteratively enhance our models and get the most realistic results.
The values of the error $\mathcal{E}$ for the two optimal models over five splits are reported in Table 3,

Seed	$\Omega\kappa$-NN	$\Gamma\kappa$-NN
110	41.79%	22.01%
120	36.94%	25.75%
130	34.7%	19.78%
140	34.33%	15.67%
150	36.19%	18.28%

Table 3: Compared $\mathcal{E}$ obtained by optimal $\Omega\kappa$-NN and $\Gamma\kappa$-NN on Γ'

where it is clear the dominance of our GoO-Oriented approach.
A plot of the classification performed by the optimal $\Gamma\kappa$-NN is reported in Figure 1. The seed used in this case is 0.

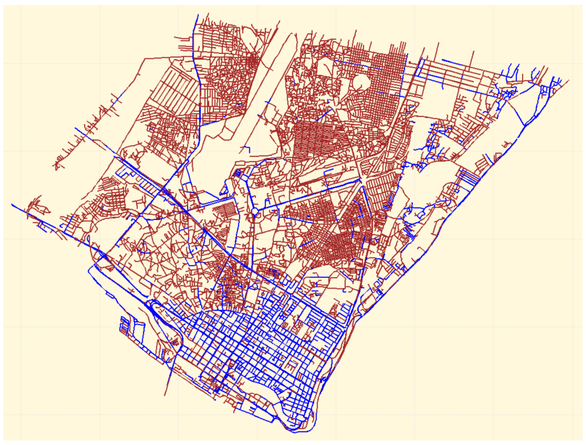

Figure 1: Optimal $\Gamma\kappa$-NN model on Γ'
🟦 Paved 🟥 Unpaved

The same model was then applied to the entire Greater Maputo area, represented by Γ, leading to the results in Table 4.

Seed	$\mathcal{E}$
100	25.34%
200	29.11%
300	30.48%
400	35.96%
500	25%

Table 4: Results of optimal $\Gamma\kappa$-NN model on Γ

These final results were obviously slightly worse in terms of $\mathcal{E}$ than the ones obtained on Γ', but this was an expected behavior, since hyperparameters were tuned on the limited dataset. This allowed for low computational times (on average 39.2 s were needed for an entire classification of Γ), without hindering too much the efficiency of the classification.

4. GCN

4.1. Theory

This second approach was mainly based on the work in [2], that presented an idea for a convolutional neural network, called GCN, capable of spreading information throughout a graph. The original structure performed the following computation:

$$Z = \text{softmax}\left(\hat{A}\ \text{ReLU}\left(\hat{A}XW^{(0)}\right)W^{(1)}\right) \tag{9}$$

where $\hat{A}$ was a slightly modified version of the adjacency A, X was a matrix containing features, and Z was an $N \times 2$ matrix containing for each ω_i the probability of belonging to class 0 and class 1. By definition, $\forall \omega_i \in \Omega$

$$\begin{aligned} Z_{i,1}, Z_{i,2} &\in [0,1] \\ Z_{i,1} + Z_{i,2} &= 1 \end{aligned} \tag{10}$$

The idea of this approach was to feed the neural networks with the training set for multiple iterations, adjusting the weights $W^{(0)}$ and $W^{(1)}$ according to gradient descent. Afterwards, (9) was applied to the test set using the optimized $W^{(0)}$ and $W^{(1)}$ to get a final classification.
We decided to test the different performances of three models:

- **HGCN**, a classical Edge-Oriented convolutional network, with no information taken from the shape of the objects. This was achieved by setting $X = I$ in (9);
- **ΓGCN₁**, a first GoO-Oriented network, with information regarding objects given in the input. More specifically, X was set to be the one-hot-encoded version of $\underline{f_{\mathcal{U}}}$, the frequency of unpaved neighbors obtained with a traditional $\Omega\kappa$-NN;
- **ΓGCN₂**, a second GoO-Oriented network, with information regarding objects given in the output. A new output function, called oo-softmax, was defined to substitute the softmax in (9), resulting in

$$\begin{aligned} B &:= [\underline{b_0}, \underline{b_1}] \\ &= \hat{A}\ \text{ReLU}\left(\hat{A}W^{(0)}\right)W^{(1)} \end{aligned} \tag{11}$$

$$\begin{aligned} Z &= \text{oo-softmax}(B) \\ &= \text{softmax}([\alpha \cdot \underline{b_0} + (1-\alpha)\cdot \underline{f_{\mathcal{P}}}, \\ &\qquad \beta \cdot \underline{b_1} + (1-\beta) \cdot \underline{f_{\mathcal{U}}}]) \end{aligned} \tag{12}$$

where α and β were two new hyperparameters in $[0,1]$, used to weigh the contribution of edges and objects.

4.2. Results

Similarly to what was done for the κ-NN, we searched the optimal hyperparameter values for the three GCN models, for later comparison. These hyperparameters where:

- θ_1 and θ_2: thresholds such that:

$$\hat{y}_i = \begin{cases} 0 & \text{if } Z_{i,1} > \theta_2 \\ 1 & \text{if } Z_{i,1} \le \theta_1 \\ 2 & \text{otherwise} \end{cases} \tag{13}$$

- κ (only for ΓGCN_1 and ΓGCN_2): number of neighbors;
- α and β (only for ΓGCN_2): weights in (12).

We once again performed cross-validation to tune the hyperparameters and iteratively reached the three different optimal sets. In Table 5 we report the error $\mathcal{E}$ for the same train-test splits used for the κ-NN.

Seed	HGCN	ΓGCN_1	ΓGCN_2
110	24.63%	17.16%	26.12%
120	31.34%	30.97%	25%
130	28.36%	30.22%	22.76%
140	21.27%	20.9%	15.67%
150	21.64%	17.54%	16.04%

Table 5: Compared $\mathcal{E}$ obtained by optimal HGCN, ΓGCN_1 and ΓGCN_2

Both GoO-Oriented models outperformed the Edge-Oriented one, showcasing their capability of exploiting all available information. Moreover, ΓGCN_2 generally outperformed ΓGCN_1, making it our best GCN model. From now on, it will be referred to as ΓGCN.
In Figure 2 a plot of the classification performed by the optimal ΓGCN is available.
Furthermore, in Table 6 results of such model over the entire Γ for five different train-test splits are available. Like before, the errors over the entire dataset were obviously higher than over the limited one, but the gain in terms of computational time was impressive: on average, only 21.02 s were needed for the classification.

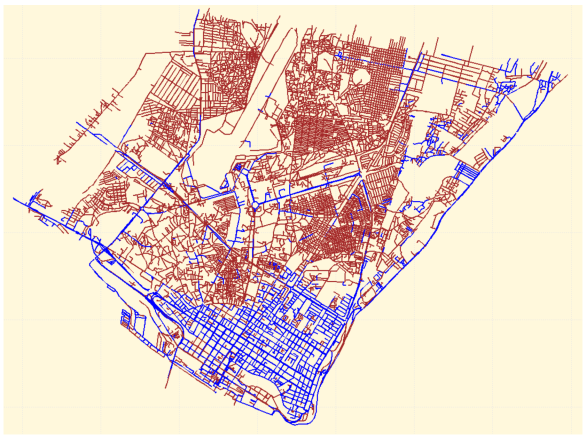

Figure 2: Optimal ΓGCN model on Γ'
🟦 Paved 🟥 Unpaved

Seed	$\mathcal{E}$
100	31.16%
200	38.7%
300	35.45%
400	34.93%
500	41.78%

Table 6: Results of optimal ΓGCN model on Γ

5. Compared Approaches

To conclude our work, we decided to compare and contrast the two algorithms designed for the analysis of Graphs of Objects according to four indicators:

- **Performance:** ability to minimize the average $\mathcal{E}$ on the limited dataset Γ'.
  By looking at the results in Table 3 and Table 5, it was clear that the two algorithms were equivalent and it was not possible to identify the best one.
- **Generalizability:** ability to minimize the average $\mathcal{E}$ on the complete dataset Γ, without re-tuning the hyperparameters.
  According to Table 4 and Table 6, Γκ-NN outperformed ΓGCN four out of five times and was deemed the winning approach.
- **Computational Time:** time needed for the classification of the entire road network represented by Γ.
  Here, ΓGCN was proven to be almost twice as fast as Γκ-NN and nominated the best approach.
- **Visual Result:** visual analysis of Figure 1 and Figure 2, to highlight possible mistakes made by the classifiers, taking into account a priori information on the structure of the city of Maputo.
  Both models presented some imperfections, but the general trend was the same and coherent with the real world scenario. The region immediately adjacent to the coast featured a substantial number of paved roads, indicative of a densely populated neighborhood. Moving away from this area, the prevalence of short unpaved roads increased exponentially, pointing towards the suburbs.
  It was not possible to consider one model better than the other only according to this inspection.

To sum up, the two models performed equivalently and none of them clearly outperformed the other. The choice was then left to the user, according to their needs in terms of dimension of the full dataset and computational time.

Conclusion

We proved the efficiency of a GoO-based approach for the classification of a road network, as compared to more simple Object-Oriented and Edge-Oriented models. Furthermore, the proposed models effectively utilized both sources of information to enhance their results. It was not possible to choose one single best model, but this allows for much more freedom for the user and adaptability to different problems.
This thesis was a first step in the world of GoO-Oriented Data Analysis, but multiple new algorithms could be developed in the future to try and exploit effectively this complex type of data.

Code

The presented results can be reproduced using the code available at https://github.com/bertrandpouget/goo.

References

[1] Arianna Burzacchi, Matteo Landrò, and Simone Vantini. Object-oriented classification of road pavement type in Greater Maputo from satellite images. *MOX Report*, 38, 2022.

[2] Thomas N. Kipf and Max Welling. Semi-supervised classification with Graph Convolutional Networks, 2017. https://arxiv.org/abs/1609.02907.