Forecasting citizenship acquisition flow

Nishant Aklecha
n.aklecha@student.tudelft.nl

Csanád Bakos
c.bakos@student.tudelft.nl

Elizaveta Evmenova
e.o.evmenova@tudelft.nl

Martin Michaux
m.michaux@student.tudelft.nl

Abstract

This project focuses on the prediction of citizenship acquisition flow between countries using graph-based models and compares their performance with baseline methods. The main objective is to assess whether graph-based approaches can outperform traditional models in this context. This work addresses the challenge of predicting edge values in a fully connected, directed graph, which is a departure from traditional node-level predictions. This requires a unique graph formulation that allows framing the problem as a node-level task. Multiple graph-based models, including Node2Vec [1] and a graph convolutional neural network (GCN), are employed to leverage the connectivity properties between countries. The connectivity is based on the similarity and dissimilarity of country features. The findings indicate that while the chosen country features play a significant role in predicting citizenship acquisition flow, the graph-based models did not surpass the performance of baseline methods. Nevertheless, some of the models are able to perform well, suggesting the relevance of the chosen features. The project provides insights into the potential and limitations of graph-based approaches for improving the accuracy of citizenship acquisition flow predictions.

1 Introduction

Citizenship acquisition flow, the process of changing one's citizenship from one country to another, is a crucial aspect of global migration. Accurately predicting the flow of citizenship acquisition between countries has significant implications for policymakers and stakeholders involved in immigration and integration efforts.

Exising migration prediction methods such as [2, 3] focus on local features exclusively. The main objective of this project is to assess whether incorporating connectivity properties between countries improves the accuracy of citizenship acquisition flow predictions. In particular, we construct similarity and dissimilarity graphs based on country features to capture the relationships and dependencies among countries. The performance of graph-based models are compared with traditional machine learning methods to evaluate their efficacy in predicting citizenship acquisition flow.

The findings of this project contribute to the understanding of the applicability of graph-based models in this domain and provide insights into the potential benefits of considering country connectivity.

This report presents the methodology, dataset, and implementation details of the project. The results and their implications are discussed, highlighting the performance of graph-based models. Lastly, the limitations of the study and suggestions for future research are also addressed.

Preprint. Under review.

2 Methods

2.1 Subject of study

In our setting, we start with a set of countries $C = \{c_1, ..., c_n\}$, where each country c_i has a corresponding feature vector per time step t, – $f_{i,t}$. A time step is one year for our study. The output, that we aim to predict for each directed pair of countries per year t, $(c_i, c_j, t), \forall i, j \in \{1, ..., n\}, i \neq j$ is $v_{i,j,t}$, the number of people changing their citizenship from country c_i to c_j in year t. We call $v_{i,j,t}$ the (citizenship) acquisition flow.

2.2 Techniques

We applied the following techniques to learn a mapping from (f_i, f_j, t) to $v_{i,j,t}, \forall i, j \in \{1, ..., n\}, i \neq j, t \in T$, where T is a set of timesteps, i.e. years.

For classical machine learning models, (f_i, f_j, t) was taken as a direct input (i.e. f_i, f_j, t were concatenated) such that varying i, j, t corresponded to separate samples. These methods included XGBoost and Random Forest which are ensemble methods based on decision trees. Furthermore, a time series model was also tried to investigate the relevance of time-based dependencies in the data. We used the ARIMAX (AutoRegressive Integrated Moving Average with Exogenous Variables) model, which combines ARIMA modeling with exogenous variables to capture external factors and provide interpretable coefficient estimation. The aim of these techniques was to establish a baseline for the task at hand.

To address to main objective of this study, graph-based models were also utilized. These included Node2Vec [1] and a Graph Convolutional Neural network (GCN) [4].

Node2Vec is a technique used for learning low-dimensional representations of nodes in a graph. For this approach we constructed a graph as follows. Each country c_i became a node and then edges were added following the k nearest neighbors principle. The distance between countries is based on the similarity of the country feature vectors f_i, calculated by the Euclidean distance metric. Using this graph, Node2Vec learns from the similarity topology and aims to preserve the network structure by generating random walks. By mapping each country to a low-dimensional vector representation, we aimed to extract meaningful patterns and similarities from the data. The learned node embeddings, $e_1, ...e_n$ were then combined with the country features $f_1, ..., f_n$ via concatenation so that each country c_i had a new feature vector composed of f_i and e_i. Finally, RandomForest and XGBoost was trained using these enriched features for directed pairs of countries (f_i, e_i, f_j, e_j, t).

GCNs are a class of graph neural networks specifically designed for node-level tasks. To be able to apply such a model we came up with the following graph framing of our data. We took ordered pairs of countries $(c_i, c_j) \forall i, j \in \{1, ..., n\}, i \neq j$ as nodes in the graph, yielding $n * (n - 1)$ nodes. Their ordered pairs of features (f_i, f_j, t) were concatenated and used to compute distances and hence similarities between the nodes. Afterward, a similarity/dissimilarity thresholding principle was applied to add edges between the nodes. Fig. 1 shows the change of the sparsity and connectivity with varying the similarity/dissimilarity threshold. While the behavior of the sparsity of the similarity-based graph is reversed to the dissimilarity-based one, the behavior of the connectivity is not. This approach allowed us to consider the relationships between specific country pairs while preserving the order between them. The resulting graph contains the ordered, concatenated features for the nodes which the GCN can utilize for the acquisition flow prediction, which is now a node-level task.

3 Numerical Experiments and Results

3.1 Data

The dataset used in this study consists of citizenship acquisition flow records between countries from the Organisation for Economic Co-operation and Development (OECD)[1] including 36 members. These records span a 20-year period and contain information on the flow value as well as features of the source and target countries. The features encompass various socio-economic indicators for each country per year. These include carbon emissions, education expenditure, foreign direct investment

[1] `https://stats.oecd.org/Index.aspx?DataSetCode=MIG`

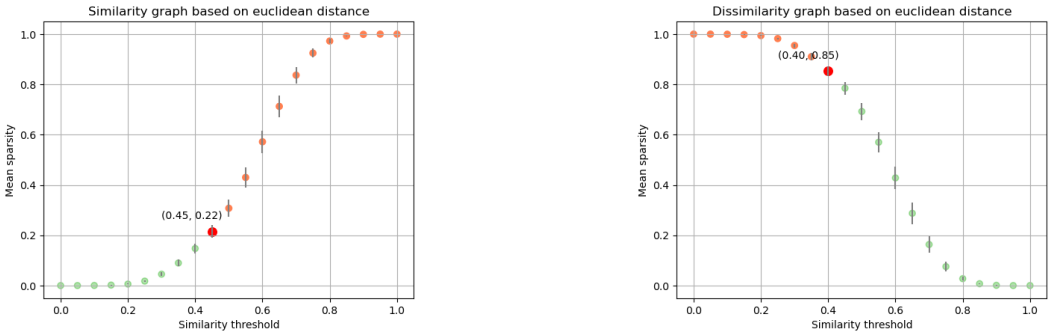


Figure 1: The mean sparsity of the obtained graphs using threshold cut-off for each year. On the left – similarity case; on the right – dissimilarity case. Green dots stand for graphs that are all connected. The orange dot means that at least one graph is disconnected. The red dot stands for graphs that have maximum sparsity while still being connected.

(FDI) inflows, GDP, health expenditure, inflation rate, internet penetration, life expectancy, renewable energy production and unemployment rate.

Regarding the acquisition flow, we faced the problem of data incompleteness, which is crucial for the stated objective. The decision was to linearly interpolate missing values for the countries, where the amount of the provided data allows to do so time-wise. The first step was to filter out the countries with insufficient amount of data points. According to Fig. 2, the following countries do not have enough information for interpolation and thus were erased from our dataset: Czech Republic (CZE), Estonia (EST), Japan (JPN), Korea (KOR), Portugal (PRT), Slovakia (SVK), Turkey (TUR) and Israel (ISR).

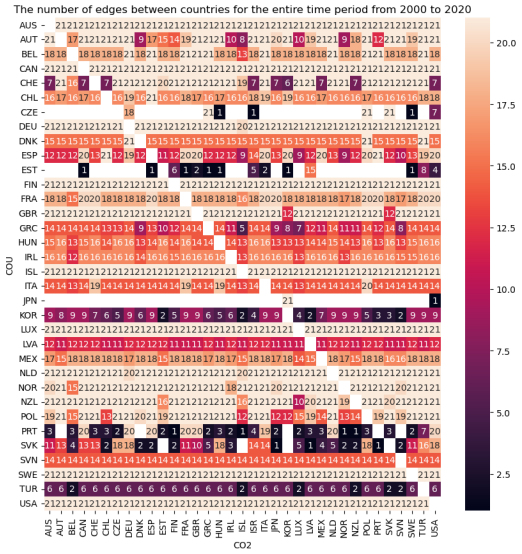


Figure 2: Number of provided time-wise data points between countries. The maximum is $T = 22$.

3.2 Experimental Setup

While predicting the acquisition flow data by default is a regression task, we have decided to also transform it to a classification problem for most of the experiments. This was done to ease result interpretation and avoid overfitting. To achieve this, the flow values were discretized into a number of bins based on percentiles, such that each bin has an (approximately) equal number of data points. This way, the resulting classes were balanced easing the learning of the classification models. Furthermore, the data were normalized to avoid issues coming from the different scales of the various features considered.

The dataset is divided differently based on the specific models used in the experiments. For the classification and regression baselines, as well as the Node2Vec model, a simple train-test split is performed. 80% of the data is randomly selected for training, while the remaining 20% is reserved for testing. On the other hand, for the time series and GCN models, the splitting is done based on the

time steps. Specifically, the data from 2000 to 2016 is used for training, while the data from 2017 to 2020 is set aside for testing.

For the classical machine learning models (XGBoost and Random Forest), we performed a grid search to optimize their hyperparameters using cross-validation on the training set. The hyperparameters considered for each model included the number of estimators, learning rate, maximum depth, and regularization parameters.

For the Node2Vec approach, we set the number of random walks to 10, the walk length to 80, the dimension of the node embeddings to 32 and k in for the KNN graph as 5. We trained the Node2Vec model on the graph constructed from the country features and obtained the node embeddings. The hyperparameters were determined by plotting the accuracy of the validation set accuracy with the hyperparameters as seen in Figure 3.

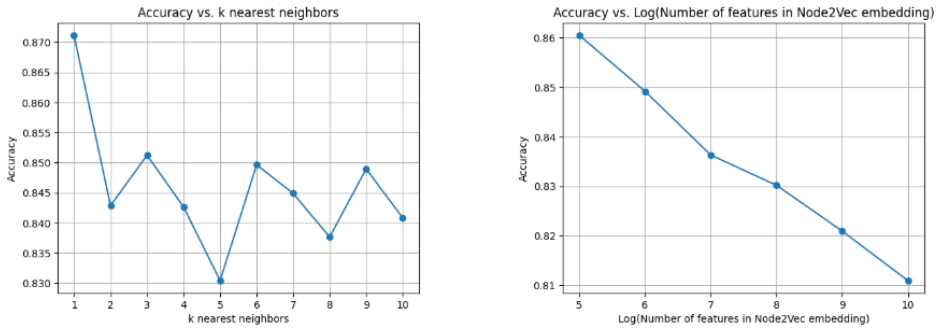


Figure 3: Hyperparameter tuning for Node2Vec

The GCN model was built using 3 hidden layers, each with 64 neurons using TAGConv layers [4] with PRelu activations after each layer. The number of hops, i.e. K hyperparameter, was set to 2. The model was trained on graphs constructed based on both similarity and dissimilarity, separately. An initial exploration showed that having different (dis)similarity thresholds to vary the sparsity had negligible effects on performance. Therefore for our experiments the threshold was chosen so that the sparsity would be fixed at approximately 95%, i.e. 95% of the possible edges are absent in the graph.

Our work is implemented in Python using the PyTorch (Geometric) [5, 6] and Scikit-learn [7] libraries. The code and implementation details can be found in our GitHub repository[2].

3.3 Results

We present the results obtained from the baselines, separately for regression, classification, and time series forecasting tasks, as well as the performance of the graph-based approaches.

3.3.1 Baseline Regression Results

For the regression task, we compared the performance of the baseline models: XGBoost, Random Forest and ARIMAX. The mean squared error (MSE) was used as the evaluation metric. The XGBoost model achieved an MSE of 7.196e-5, the Random Forest model achieved an MSE of 1.102e-4 and the time series model achieved 6.177e-5. The results are also visualized in Figure 4.

3.3.2 Baseline Classification Results

Both the XGBoost and Random Forest models achieved an accuracy of 0.85 when evaluated on the test set. However, the classification accuracy of ARIMAX was found to be 0.43. Due to the low performance of using temporal aspects, we have decided to limit our scope to graph approaches without explicit modelling of time-based dependencies in the data.

[2]Link to GitHub repository: `https://github.com/MartinMichaux/ML-for-Graph-Data-Project`

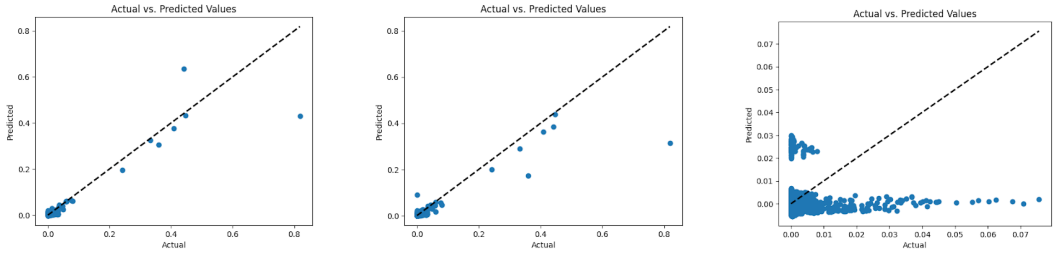


Figure 4: XGBoost (left), ARIMAX (middle) and Random Forest (right) regression predicted vs true values

For the classification task, we employed the same XGBoost, Random Forest models and ARIMAX models. The goal was to predict the class label of the acquisition flow based on the country features. The class boundaries were defined as follows: Class 0 – [0, 4); Class 1 – [4, 57); Class 2 - [57, 231752).

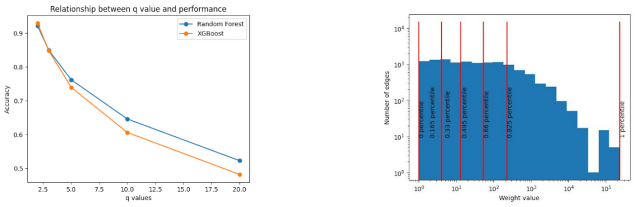


Figure 5: Accuracy of the model with different number of classes (left) and percentiles of the flow distribution with 6 classes, when the bin overlapping happens (right).

As for choosing the number of classes, we faced the following issue. Since we used percentiles as boundaries for the classes, after dividing the data into 6 parts we noticed a class overlap (see Fig 5). That means that, while increasing the number of classes, at a certain point (6 classes) the boundaries of two classes (0-percentile and 0.165-percentile) overlap so that the values that are between these two percentiles are related to different classes while being indistinguishable between each other.

3.3.3 Graph-based Approach Results

The Node2Vec approach achieved an accuracy of 0.85, which was identical to the accuracy achieved by the baseline models. This observation is depicted in Figure 3, where the accuracy of the Node2Vec approach is illustrated.

Based on the results, it is hypothesized that the Node2Vec approach did not effectively utilize the Node2Vec features. Instead, it appears to have relied on the country features themselves to achieve comparable accuracy to the baseline models. Consequently, no improvement in accuracy was observed when using the Node2Vec approach.

The performance of the GCN model was evaluated using node classification. Its performance is showcased in Figure 6. We can see that the accuracy using similarity and dissimilarity graphs does not differ by much. Overall the similarity graph input yields slightly better performance. Unfortunately, this approach did not outperform the baseline methods in terms of accuracy.

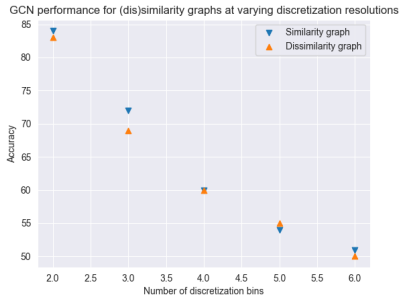
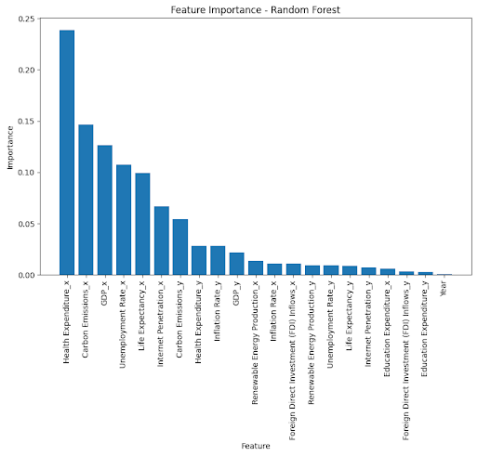


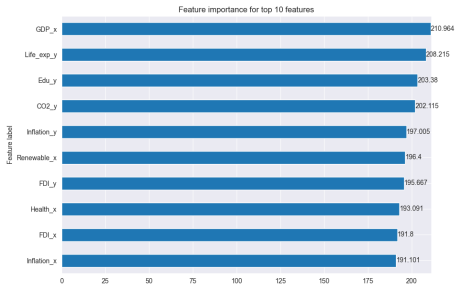
Figure 6: GCN performance

3.3.4 Explanability

To gain further insights into the importance of country features in predicting citizenship acquisition flow, we analyzed the feature importances for the baseline models and GCN. In the case of GCN,



(a) Random Forest feature importances



(b) GCN Similarity Graph top 10 features via GNNExplainer for acquisition flow from Australia to Canada

Figure 7: Feature importances for explainability

GNNExplainer [8] was used to obtain feature importances for each node in the graph. This allows one to investigate for each country pair, the most important factors for citizenship transfers. These factors varied greatly across different pairs of nodes, therefore we saw no point in creating aggregations, thus we only present an example in Figure 7b, to showcase the feature importance regarding acquisition flow originating from Australia transferring to Canada.

4 Discussion

The project results show that the chosen features of countries are important in predicting acquisition flow, especially, the features of the source country (see Fig. 7a). One of the interpretations could be that the patterns of citizenship migration might depend not on the benefits of a destination, but rather on the disadvantages of one's present location, and that is what affects people's decision on changing their nationality in the first place.

One of the possible future research directions is to explore different types of graph constructions of this problem. For example, one can build connectivity between countries based on different phenomenons such as geographical distance, trading alliances, transport connectivity, historical migration patterns, etc. Furthermore, besides GCN there are various GNN models available that may perform better for the different graph formulations.

While applying the GCN model on graphs constructed by different principles (similarity- and dissimilarity-based), we obtained an alike performance (see Fig. 6). That leads us to the conclusion that the topology of a graph, in our case, does not add any significant information to the model, but, also, does not interfere with it.

5 CRediT author statement

The team members main contributions were as follows.

Nishant Aklecha: Node2Vec, XGBoost

Csanád Bakos: GCN, GNNExplainer, graph formulations

Elizaveta Evmenova: Data exploration and preparation, graph formulations

Martin Michaux: Random Forest, ARIMAX

The other tasks such as writing the report, idea formulation, etc. were split equally among the team members.

References

[1] Aditya Grover and Jure Leskovec. "node2vec: Scalable feature learning for networks". In: *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*. 2016, pp. 855–864.

[2] Caleb Robinson and Bistra Dilkina. "A Machine Learning Approach to Modeling Human Migration". In: *Proceedings of the 1st ACM SIGCAS Conference on Computing and Sustainable Societies*. COMPASS '18. Menlo Park and San Jose, CA, USA: Association for Computing Machinery, 2018. ISBN: 9781450358163. DOI: 10.1145/3209811.3209868. URL: https://doi-org.tudelft.idm.oclc.org/10.1145/3209811.3209868.

[3] Caleb Robinson, Bistra Dilkina, and Juan Moreno-Cruz. "Modeling migration patterns in the USA under sea level rise". In: *Plos one* 15.1 (2020), e0227436.

[4] Jian Du et al. "Topology adaptive graph convolutional networks". In: *arXiv preprint arXiv:1710.10370* (2017).

[5] Matthias Fey and Jan E. Lenssen. "Fast Graph Representation Learning with PyTorch Geometric". In: *ICLR Workshop on Representation Learning on Graphs and Manifolds*. 2019.

[6] Adam Paszke et al. "PyTorch: An Imperative Style, High-Performance Deep Learning Library". In: *Advances in Neural Information Processing Systems 32*. Curran Associates, Inc., 2019, pp. 8024–8035. URL: http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf.

[7] F. Pedregosa et al. "Scikit-learn: Machine Learning in Python". In: *Journal of Machine Learning Research* 12 (2011), pp. 2825–2830.

[8] Zhitao Ying et al. "Gnnexplainer: Generating explanations for graph neural networks". In: *Advances in neural information processing systems* 32 (2019).