Report GNN Project

by Guilhem DUPUY, Anh Duy VU, Artus BLETON

Study of airport graph: Link Prediction on the Plane Network

1 Dataset Description

The dataset represents an **airport network**, where each node corresponds to a city or airport characterized by several features:

- Longitude (`lon`)
- Latitude (`lat`)
- Population
- Country (categorical, one-hot encoded)
- City name (string attribute)

After preprocessing, we obtained **3363 nodes** and **27,094 undirected edges**. Each node is described by a **215-dimensional feature vector** combining numerical and categorical attributes. The task is **link prediction**: predicting whether a connection (route) exists between two airports using graph-based methods.

2 Models and Baselines

We compared **learning-based methods** (GAE and VGAE) against **classical topological heuristics** commonly used for link prediction.

2.1 Learning-based models

Graph Autoencoder (GAE): A deterministic encoder-decoder model using two GCN layers for node embedding and a dot-product decoder for edge reconstruction.

Variational Graph Autoencoder (VGAE): Similar to GAE but introduces Gaussian latent variables (μ, $\log \sigma$) and a Kullback-Leibler (KL) divergence term for probabilistic embedding regularization.

2.2 Classical heuristics

Jaccard Coefficient: Measures the ratio of common neighbors to total neighbors between two nodes.

$$J(u, v) = \frac{|N(u) \cap N(v)|}{|N(u) \cup N(v)|}$$

Preferential Attachment (PA): Models hub tendency, based on the product of node degrees.

$$PA(u, v) = |N(u)| \times |N(v)|$$

These baseline methods rely purely on graph connectivity and require no training or node features.

3 Features sets analysis

In order to verify which dataset allows for the best performance, we first performed an ablation study on the Features themselves.

3.1 Feature sets definition

We decomposed our original node features into 3 distinctive sets:

- **Full features**: all the data available for each airport (each graph node): population, lat, lon, country.
- **Numerical features only**: a subset of our original features, without the country name.
- **No node features**: removal of all features for all nodes.

3.1.1 No node features

In order to simulate the "no node feature" mode, we entered an identity matrix defined as `np.eye(df.shape[0], dtype=np.float32)` as node features in our models. This is equivalent to representing each node by a **unique one-hot vector** with no other information.

The idea behind this "empty" feature set is to remove all semantic information from our graph. This forces the model to learn embeddings based on **connectivity patterns alone**, hence from pure **graph structure**. This baseline will help us understand the impact of our features on model performance.

3.2 Experimental setup

This study was performed using the following setup:

- **Data split:** 80% train edges, 10% test edges, 10% validation edges (using `train_test_split_edge` from PyTorch Geometric)
- **Latent dimension:** 16
- **Optimizer:** Adam

- **Learning rates tested:** 0.01, 0.001 (best result retained)
- **Epochs:** 600
- **Evaluation metrics:** Area Under ROC Curve (AUC) and Average Precision (AP)

Both AUC and AP were computed on the test link set (`test_pos_edge_index`, `test_neg_edge_index`).

3.3 Results

Feature Set	Model	AUC	Avg Precision (AP)
Full features	VGAE-Linear	0.8840	0.8787
	VGAE	0.9220	0.9119
	GAE	**0.9461**	**0.9439**
	Jaccard (classical)	0.9346	0.9307
	Preferential Attachment (classical)	0.9101	0.9213
	1-Layer-GAE	0.8312	0.7614
Numerical features only	VGAE-Linear	0.8302	0.8193
	VGAE	0.8869	0.8746
	GAE	**0.9117**	**0.9089**
	Jaccard (classical)	0.9346	0.9307
	Preferential Attachment (classical)	0.9101	0.9213
No node features	VGAE-Linear	0.9195	0.9357
	VGAE	0.8702	0.8820
	GAE	0.9188	0.9328
	Jaccard (classical)	**0.9346**	**0.9307**
	Preferential Attachment (classical)	0.9101	0.9213

4 Models comparison

4.1 Experimental setup

The setup of this experiment is similar to the one previously used:

- **Data split:** different variations compared
- **Latent dimension:** 16
- **Optimizer:** Adam
- **Learning rates tested:** 0.01, 0.001 (best result selected)
- **Epochs:** 600
- **Evaluation metrics:** AUC and AP

Both AUC and AP were computed on the test link set (`test_pos_edge_index`, `test_neg_edge_index`).

4.2 Results

Split Configuration	Model	AUC	AP	Train %
10% Test / 10% Val	VGAE-Linear	0.8840	0.8787	80%
	VGAE	0.9220	0.9119	80%
	GAE	**0.9461**	**0.9439**	80%
	Jaccard	0.9346	0.9307	-
	PA	0.9101	0.9213	-
20% Test / 10% Val	VGAE-Linear	0.8847	0.8814	70%
	VGAE	0.9219	0.9151	70%
	GAE	**0.9480**	**0.9443**	70%
	Jaccard	0.9369	0.9340	-
	PA	0.9073	0.9189	-
20% Test / 20% Val	VGAE-Linear	0.8834	0.8777	60%
	VGAE	0.9147	0.9087	60%
	GAE	0.9367	0.9383	60%
	Jaccard	0.9173	0.9137	-

	PA	0.9125	0.9214	-
30% Test / 10% Val	VGAE-Linear	0.8817	0.8770	60%
	VGAE	0.9126	0.9027	60%
	GAE	0.9346	0.9330	60%
	Jaccard	0.9360	0.9326	-
	PA	0.9094	0.9201	-

Summary:

Best split: **20% test / 10% validation** (70% training) achieves the best overall performance, with GAE reaching AUC 0.9480 and AP 0.9443.

Key Findings:

- **Training data matters**: Models with 70–80% training data significantly outperform those with only 60% training data. Reducing training data from 80% to 60% causes GAE to drop from AUC 0.9461 to 0.9346–0.9367.
- **GAE consistently dominates**: GAE outperforms VGAE across all splits, confirming that deterministic embeddings work better for this clean, structured airport network.
- **VGAE degrades faster**: VGAE shows higher sensitivity to reduced training data, dropping from AUC 0.9220 (80% train) to 0.9126 (60% train).
- **Classical heuristics remain stable**: Jaccard and PA maintain relatively consistent performance (AUC 0.91–0.93) regardless of split ratio since they don't require training.

5 Experiments Analysis

5.1 Feature Sets Analysis

5.1.1 a. Full Features

- **GAE (AUC 0.9461, AP 0.9439)** performs best, showing that deterministic embeddings from GCNs with full node attributes excel.
- **VGAE (AUC 0.9220, AP 0.9119)** lags slightly behind GAE, likely due to the KL regularization that adds noise.
- **Linear encoder (AUC 0.8840, AP 0.8787)** is weaker but surprisingly strong, demonstrating that some linear feature transformations capture useful info.
- **Classical heuristics (Jaccard AUC 0.9346, PA AUC 0.9101)** do well too, but GAE improves further by incorporating node features.

5.1.2 b. Numerical Features Only

- All scores drop compared to full features, indicating **categorical features (e.g., country encoding) add predictive value**.
- GAE (AUC 0.9117) outperforms classical PA (AUC 0.9101), but VGAE underperforms classical.
- Linear drops more steeply (AUC 0.8302), showing the benefit of nonlinear modeling for numeric-only features.

5.1.3 c. No Node Features (Identity matrix)

- Linear encoder surprisingly performs very well (AUC 0.9195, AP 0.9357), showing that unique node identifiers plus simple linear maps can still encode connectivity well.
- GAE (AUC 0.9188, AP 0.9328) also performs strongly, close to heuristics.
- VGAE slightly worse (AUC 0.8702); stochastic sampling may not help here.
- Classical heuristics remain very competitive.

5.1.4 Overall Insights

- We can validate that our **node features enhance prediction:** full features > numerical only > none.
- However, we remark that **topology alone (no features) remains very powerful**.
- **Linear encoder is a strong baseline**, particularly with unique node features (identity matrix). It can surprisingly rival nonlinear VGAE, highlighting the importance of choosing strong baselines to properly evaluate the relevance of a model.

5.2 Overall Models Comparison

5.2.1 a. GAE Consistently Outperforms VGAE

Observation:

- GAE outperforms VGAE in **all feature settings** (full, numerical, none).
- With full features: GAE AUC 0.9461 vs VGAE 0.9220 (difference of ∼0.024).
- With numerical only: GAE 0.9117 vs VGAE 0.8869 (∼0.025 difference).
- With no features: GAE 0.9188 vs VGAE 0.8702 (∼0.049 difference).

Our analysis:

- VGAE introduces a **KL divergence regularization** term that enforces the latent space to match a prior Gaussian distribution.
- According to our research, this regularization is beneficial for **sparse, noisy or incomplete data**, but our airport network is clean, deterministic, and highly structured.
- As a result, in our case, the noise added by stochasticity in VGAE **degrades the predictions** instead of improving generalization.

5.2.2 b. Classical Heuristics Are Remarkably Strong

Observation:

- Jaccard coefficient consistently achieves **AUC 0.9346 and AP 0.9307** regardless of feature configuration.
- Preferential Attachment (PA) achieves **AUC 0.9101 and AP 0.9213**, also stable.

Our analysis:

- Both heuristics rely **purely on graph topology**: neighborhood overlap (Jaccard) and degree product (PA).
- This is why they remain unchanged, whatever the model used. They are **featureless baselines** because they don't use node attributes at all.
- Since the airport network connectivity is highly **topological** (airports connect based on existing routes and hub structures), these heuristics naturally capture the dominant link formation mechanism.
- Thanks to the strong baseline they offer for link prediction, we know that any learned model must surpass ∼0.93 AUC to justify its complexity.
- Still, we managed to outperform these heuristics using rich features and GCN embeddings.

5.3 Ablation Study

The goal of this ablation study is to isolate the contribution of each architectural and data-related component in our best link prediction model: the GAE model with 2 GCN layers.

5.3.1 a. Methodology

We compared through our experiments the following variations:

Feature ablation: By removing or restricting node attributes (Full features → Numerical only → None). → Evaluates the contribution of semantic information.

Model ablation: By replacing GCN layers with a Linear encoder. → Tests whether message passing truly helps beyond a simple linear projection.

Depth ablation: By reducing the number of GCN layers from 2 to 1. → Checks if deeper architectures are beneficial or redundant for this topology.

5.3.2 b. Ablation Study Table

Feature Set	**Model**	**AUC**	**Average Precision (AP)**
Full features	Linear	0.8840	0.8787
Full features	**GAE GCN**	**0.9461**	**0.9439**
Full features	GAE (1-layer)	0.8312	0.7614
Numerical only	GAE	0.9117	0.9089
No node features	GAE	0.9188	0.9328

Table 3: Ablation study results across feature and model configurations.

5.3.3 c. Analysis

We therefore justified the use of the following features for our model:

- The "full features" model gives the best results.
- The use of a GCN encoder outperforms the use of a linear one.
- The 2-layer GCN model outperforms the 1-layer one.

6 Main Conclusions

Here, a quick recap of the main findings of this study:

1. **Need for a powerful baseline**: Classical heuristics and no-feature models performed surprisingly well in our experiments. It allowed us to put into perspective the added efficiency of our trained models.
2. **Node features add value when rich**: Full features (including categorical) push GAE to its highest performance, numerical features alone are insufficient.
3. **Deterministic models > Variational models**: We verified that for clean, structured graphs, like airport networks, GAE consistently outperforms VGAE.
4. **Ablation study** allows us to validate a clean, minimalist model design which only includes features proven to be beneficial to model efficiency.