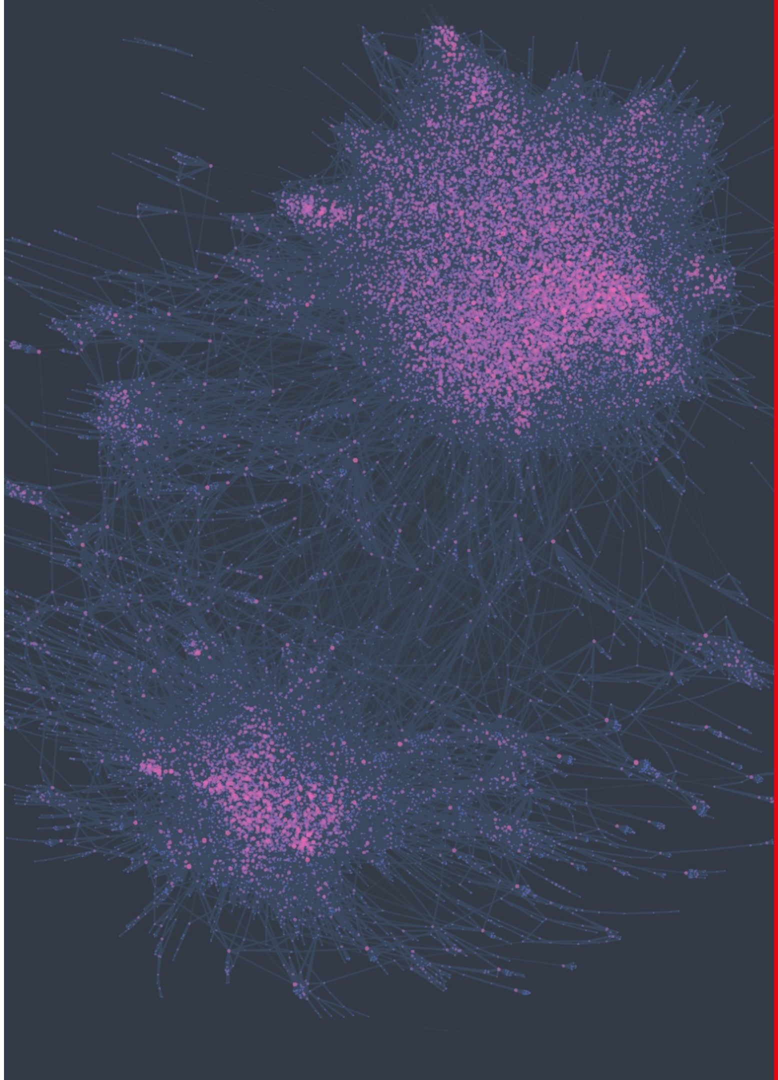


Gender Classification of Deezer Europe Users with Graph Neural Networks

Group 24

Xinyu Liu,

Weijiang Xiong

Contents

- Network exploration
- Feature embedding
- User gender classification

Dataset

- Social network of Deezer users
- Nodes: Deezer users (28, 281 nodes)
- Node features: list of artists liked by the users
- Edges: Mutual followership between users (92, 752 edges)

Dataset

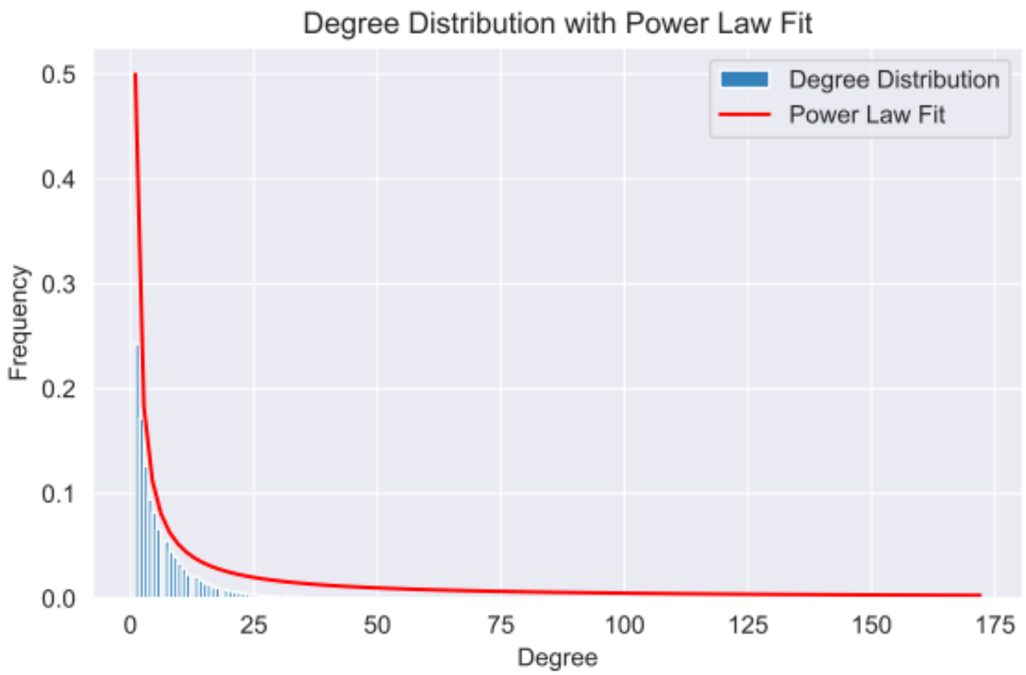
- Social network of Deezer users
- Nodes: Deezer users (28, 281 nodes)
- Node features: list of artists liked by the users
- Edges: Mutual followership between users (92, 752 edges)

- Low clustering coefficient and high diameter (not a small world)

Nodes	Edges	Density	Clustering Coefficient	Diameter	Features	Classes
28281	92752	0.00023	0.141	21	31240	2

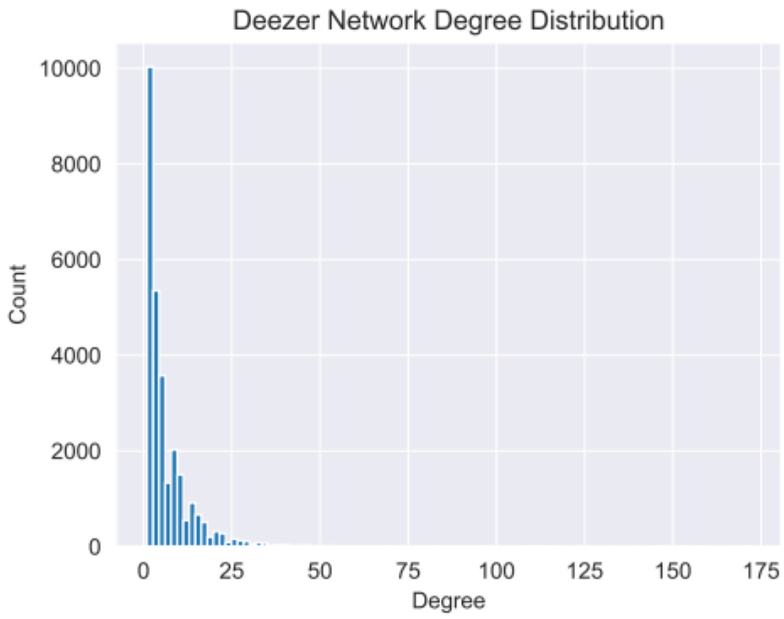
Degree Distribution

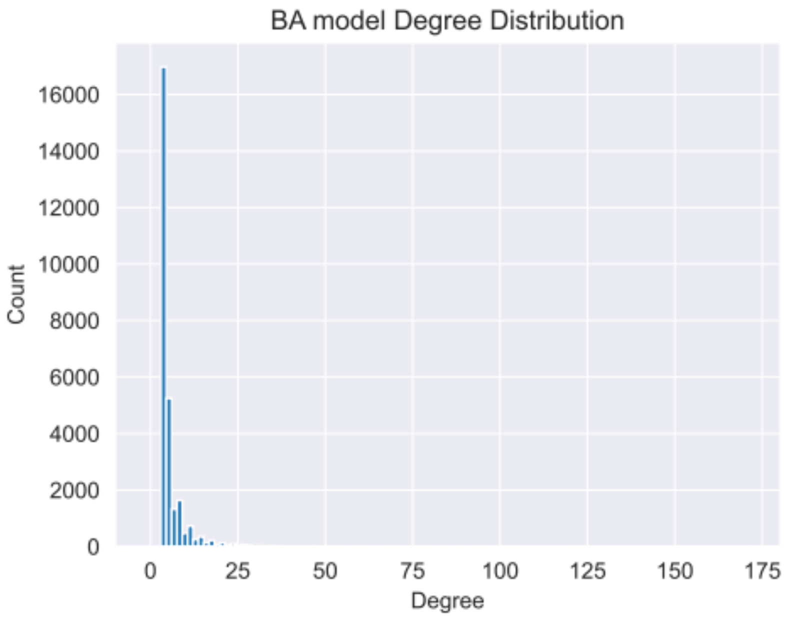
- Degree distribution can be fitted by a power-law distribution



Random Network Model

- The degree distribution resembles a Barabasi-Albert (BA) model.



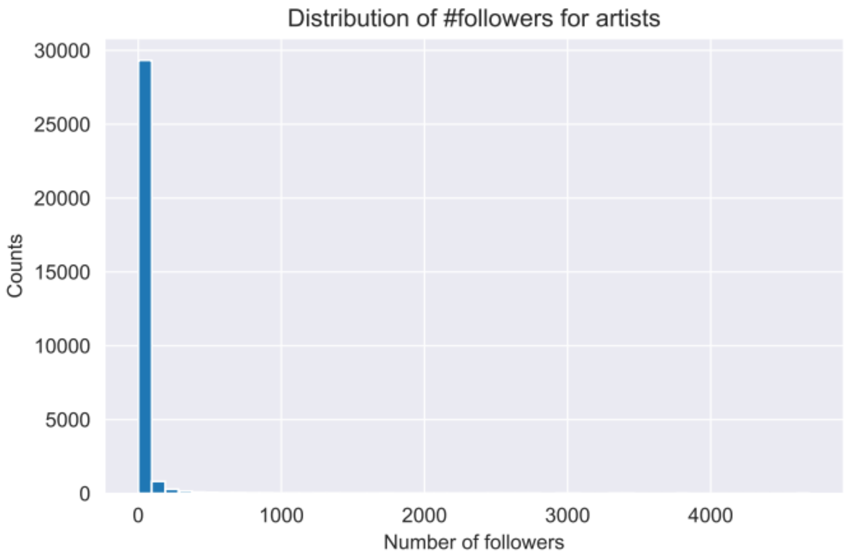


Most liked Artists

- Features: artists followed by users.
- Most artists have few followers.
- A few popular artists exists

TABLE II: Artists with more than 3k followers

Artist ID	505	12	251	675	21342	87
#Follower	4693	3777	3401	3376	3311	3013



Feature embedding

- Binary: 0/1 vector to represent liked artists of the user

$$\mathbf{x}_i = [b_i^0, b_i^1, ..., b_i^M], \quad b_i^k = 1 \quad \text{if } k \in \mathcal{A}_i,$$

Feature embedding

- Binary: 0/1 vector to represent liked artists of the user

$$\mathbf{x}_i = [b_i^0, b_i^1, ..., b_i^M], \quad b_i^k = 1 \quad \text{if } k \in \mathcal{A}_i,$$

- Feather: compose embeddings characteristic functions to node

```
X = SVD(X)
X = concatenate([X*theta
        for theta in linspace(0.01, 2.5, 25)])
X = concatenate([cos(X), sin(X)])
X = concatenate([matrix_power(L, i) @ X
        for i in range(5)])
```

Normalized Laplacian

Data: $\widehat{\mathbf{A}}$ – Normalized adjacency matrix.
$\mathcal{X} = \{\mathbf{x}^1, \ldots, \mathbf{x}^k\}$ – Set of node feature vectors.
$\widetilde{\Theta} = \{\Theta^{1,1}, \ldots, \Theta^{1,r}, \Theta^{2,1}, \ldots, \Theta^{k,r}\}$ – Set of evaluation point vectors.
r – Scale of empirical graph characteristic function.

Result: Node embedding matrix $\mathbf{Z}$.

$\mathbf{Z}_{Re} \leftarrow$ Initialize Real Features()
$\mathbf{Z}_{Im} \leftarrow$ Initialize Imaginary Features()
for i in $1:k$ **do**
  for j in $1:r$ **do**
    for l in $1:j$ **do**
      if $l = 1$ **then**
        $\mathbf{H} \leftarrow \mathbf{x}^i \otimes \Theta^{i,j}$
        $\mathbf{H}_{Re} \leftarrow \cos(\mathbf{H})$
        $\mathbf{H}_{Im} \leftarrow \sin(\mathbf{H})$
      $\mathbf{H}_{Re} \leftarrow \widehat{\mathbf{A}}\mathbf{H}_{Re}$
      $\mathbf{H}_{Im} \leftarrow \widehat{\mathbf{A}}\mathbf{H}_{Im}$
    end
    $\mathbf{Z}_{Re} \leftarrow [\mathbf{Z}_{Re} \mid \mathbf{H}_{Re}]$
    $\mathbf{Z}_{Im} \leftarrow [\mathbf{Z}_{Im} \mid \mathbf{H}_{Im}]$
  end
end
$\mathbf{Z} \leftarrow [\mathbf{Z}_{Im} \mid \mathbf{Z}_{Re}]$
Output $\mathbf{Z}$.

Rozemberczki, Benedek, and Rik Sarkar. "Characteristic functions on graphs: Birds of a feather, from statistical descriptors to parametric models." *Proceedings of the 29th ACM international conference on information & knowledge management*. 2020.

Feature embedding

- Binary: 0/1 vector to represent liked artists of the user

$$\mathbf{x}_i = [b_i^0, b_i^1, ..., b_i^M], \quad b_i^k = 1 \quad \text{if } k \in \mathcal{A}_i,$$

- Feather: compose embeddings characteristic functions to node

```
X = SVD(X)
X = concatenate([X*theta
        for theta in linspace(0.01, 2.5, 25)])
X = concatenate([cos(X), sin(X)])
X = concatenate([matrix_power(L, i) @ X
        for i in range(5)])
```

Normalized Laplacian

Data: $\widehat{\mathbf{A}}$ – Normalized adjacency matrix.
$\mathcal{X} = \{\mathbf{x}^1, \ldots, \mathbf{x}^k\}$ – Set of node feature vectors.
$\widetilde{\Theta} = \{\Theta^{1,1}, \ldots, \Theta^{1,r}, \Theta^{2,1}, \ldots, \Theta^{k,r}\}$ – Set of evaluation point vectors.
r – Scale of empirical graph characteristic function.

Result: Node embedding matrix $\mathbf{Z}$.

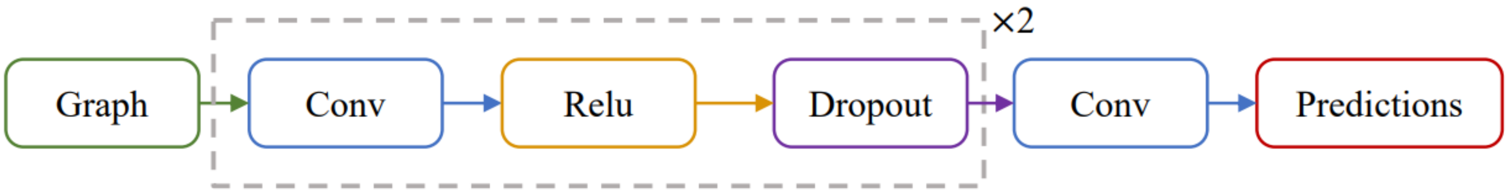
$\mathbf{Z}_{Re} \leftarrow$ Initialize Real Features()
$\mathbf{Z}_{Im} \leftarrow$ Initialize Imaginary Features()
for i in $1:k$ **do**
  for j in $1:r$ **do**
    for l in $1:j$ **do**
      if $l = 1$ **then**
        $\mathbf{H} \leftarrow \mathbf{x}^i \otimes \Theta^{i,j}$
        $\mathbf{H}_{Re} \leftarrow \cos(\mathbf{H})$
        $\mathbf{H}_{Im} \leftarrow \sin(\mathbf{H})$
      $\mathbf{H}_{Re} \leftarrow \widehat{\mathbf{A}}\mathbf{H}_{Re}$
      $\mathbf{H}_{Im} \leftarrow \widehat{\mathbf{A}}\mathbf{H}_{Im}$
    end
    $\mathbf{Z}_{Re} \leftarrow [\mathbf{Z}_{Re} \mid \mathbf{H}_{Re}]$
    $\mathbf{Z}_{Im} \leftarrow [\mathbf{Z}_{Im} \mid \mathbf{H}_{Im}]$
  end
end
$\mathbf{Z} \leftarrow [\mathbf{Z}_{Im} \mid \mathbf{Z}_{Re}]$
Output $\mathbf{Z}$.

- Both VERY HIGH (~30k) dimensions

Rozemberczki, Benedek, and Rik Sarkar. "Characteristic functions on graphs: Birds of a feather, from statistical descriptors to parametric models." *Proceedings of the 29th ACM international conference on information & knowledge management*. 2020.

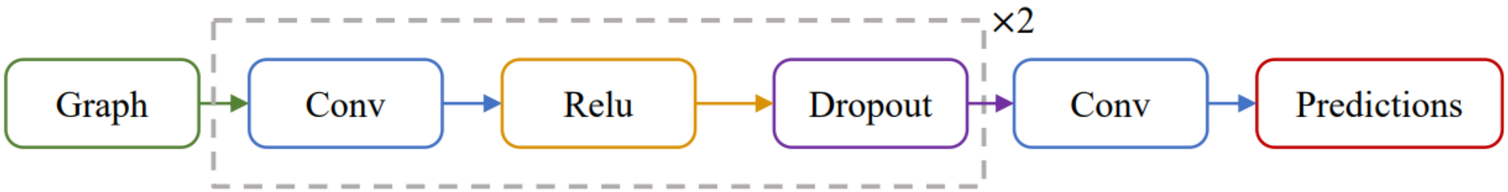
Node Classification Model

- Conv: Graph convolution (GCN) or Transformer convolution (TFC)
- TFC weights neighborhood messages by an attention score



Node Classification Model

- Conv: Graph convolution (GCN) or Transformer convolution (TFC)
- TFC weights neighborhood messages by attention scores



- Train-validation-test splits with 70%, 10%, 20% nodes
- Trained for long-enough (500) epochs
- Implemented by PyTorch Geometric

https://pytorch-geometric.readthedocs.io/en/latest/modules/nn.html#convolutional-layers

Experiment Results

- Metric: ROC AUC Score (random guessing = 0.5)
- SVD: light-weight features; Node2Vec: structural information

TABLE III: ROC AUC Scores of different feature embeddings.

No.	Feature	SVD	N2V	Dim.	Model	
					GCN	TFC
1	Binary	×	×	31241	0.640	0.725
2	Binary	✓	×	128	0.642	0.713
3	Binary	✓	✓	128	0.637	0.692
4	Binary	×	✓	31369	0.640	0.717
5	Feather	×	×	32000	0.594	0.559
6	Feather	✓	×	128	0.634	0.637
7	Preset	-	-	128	0.640	0.698
8	None	-	✓	128	0.532	0.529

generally better

https://scikit-learn.org/stable/modules/generated/sklearn.metrics.roc_auc_score.html
https://en.wikipedia.org/wiki/Receiver_operating_characteristic#ROC_space

Experiment Results

- Metric: ROC AUC Score (random guessing = 0.5)
- SVD: light-weight features; Node2Vec: structural information

TABLE III: ROC AUC Scores of different feature embeddings.

No.	Feature	SVD	N2V	Dim.	Model	
					GCN	TFC
1	Binary	×	×	31241	0.640	0.725
2	Binary	✓	×	128	0.642	0.713
3	Binary	✓	✓	128	0.637	0.692
4	Binary	×	✓	31369	0.640	0.717
5	Feather	×	×	32000	0.594	0.559
6	Feather	✓	×	128	0.634	0.637
7	Preset	-	-	128	0.640	0.698
8	None	-	✓	128	0.532	0.529

generally better

pytorch geometric built-in

almost random guessing

https://scikit-learn.org/stable/modules/generated/sklearn.metrics.roc_auc_score.html
https://en.wikipedia.org/wiki/Receiver_operating_characteristic#ROC_space

Experiment Results

- Metric: ROC AUC Score (random guessing = 0.5)
- SVD: light-weight features; Node2Vec: structural information

TABLE III: ROC AUC Scores of different feature embeddings.

No.	Feature	SVD	N2V	Dim.	Model GCN	Model TFC
1	Binary	×	×	31241	0.640	0.725
2	Binary	✓	×	128	0.642	0.713
3	Binary	✓	✓	128	0.637	0.692
4	Binary	×	✓	31369	0.640	0.717
5	Feather	×	×	32000	0.594	0.559
6	Feather	✓	×	128	0.634	0.637
7	Preset	-	-	128	0.640	0.698
8	None	-	✓	128	0.532	0.529

generally better

best performance

good balance

pytorch geometric built-in

almost random guessing

https://scikit-learn.org/stable/modules/generated/sklearn.metrics.roc_auc_score.html
https://en.wikipedia.org/wiki/Receiver_operating_characteristic#ROC_space

Experiment Results

- Metric: ROC AUC Score (random guessing = 0.5)
- SVD: light-weight features; Node2Vec: structural information

TABLE III: ROC AUC Scores of different feature embeddings.

No.	Feature	SVD	N2V	Dim.	Model GCN	TFC
1	Binary	×	×	31241	0.640	0.725
2	Binary	✓	×	128	0.642	0.713
3	Binary	✓	✓	128	0.637	0.692
4	Binary	×	✓	31369	0.640	0.717
5	Feather	×	×	32000	0.594	0.559
6	Feather	✓	×	128	0.634	0.637
7	Preset	-	-	128	0.640	0.698
8	None	-	✓	128	0.532	0.529

generally better

best performance

good balance

node2vec doesn't help

not good for GNNs

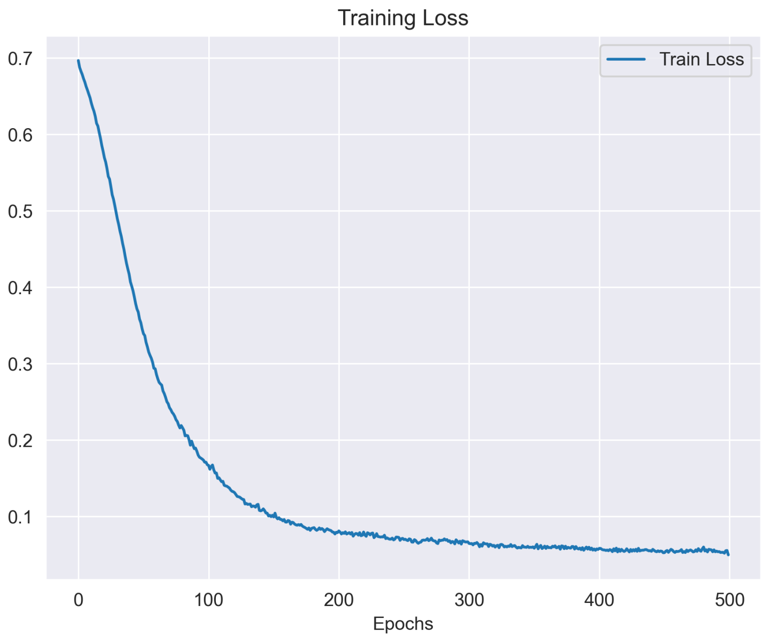
pytorch geometric built-in

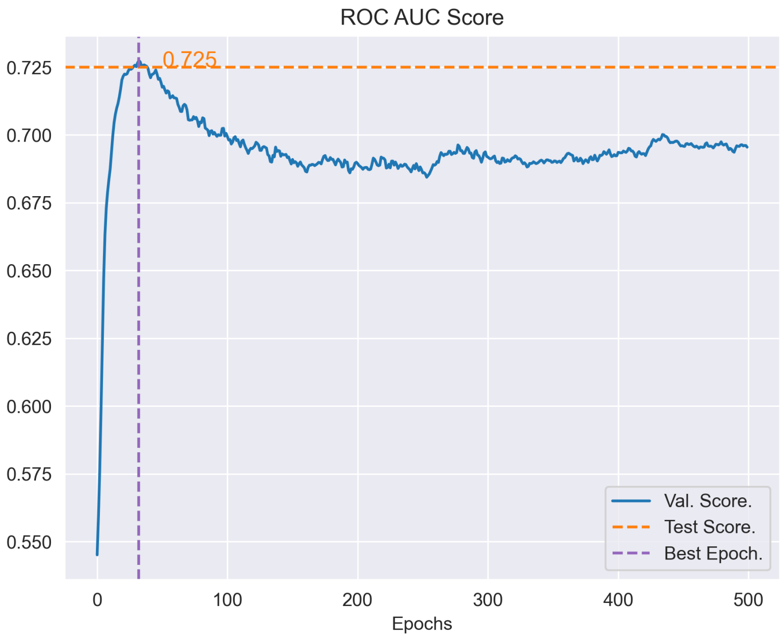
almost random guessing

https://scikit-learn.org/stable/modules/generated/sklearn.metrics.roc_auc_score.html
https://en.wikipedia.org/wiki/Receiver_operating_characteristic#ROC_space

Training Progress

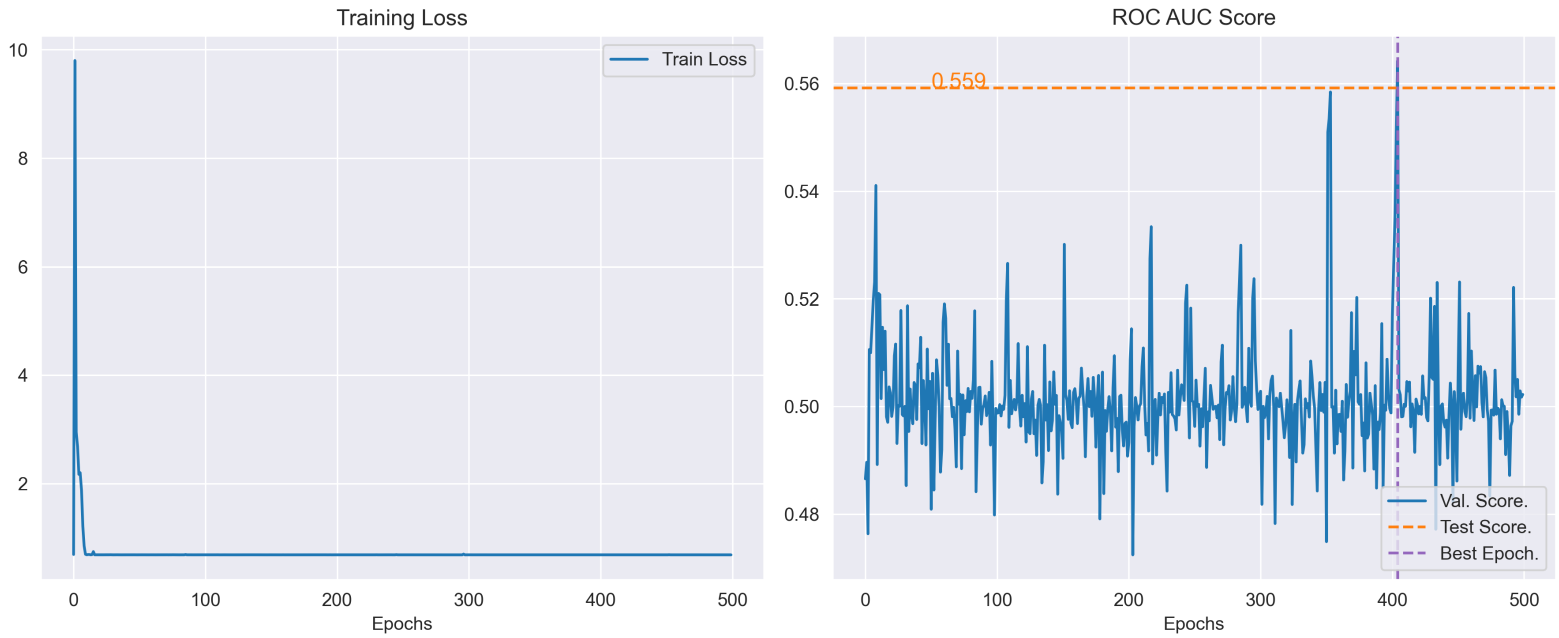
- Best performance: raw binary feature + TFC model, AUC 0.725
- Model begin to overfit from ~30 epoch





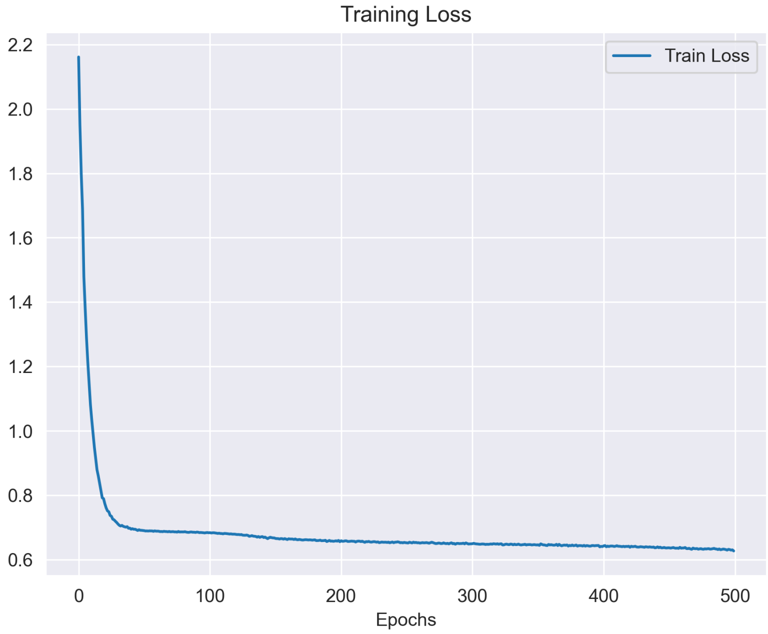
Training Progress

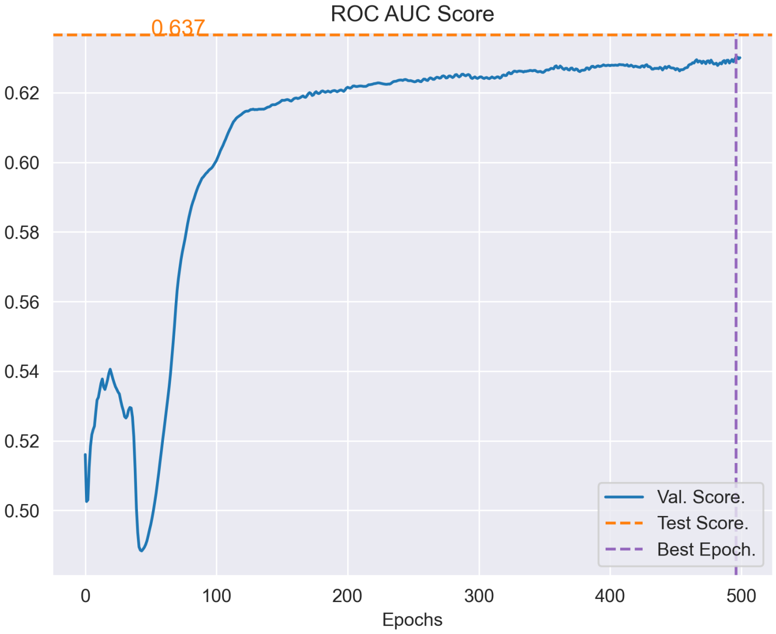
- Training with feather embedding is very unstable
- Observed both with TFC (below) and GCN



Training Progress

- SVD-reduced Feather embedding stabilizes training
- Both for TFC (below) and GCN





Conclusion

- Deezer Europe network has approximately power-law degree distribution and weak small-world properties
- Binary feature embedding is good for gender classification with GNN, and SVD balances performance and computation
- Attention scores can improve message passing
- Code at https://github.com/Weijiang-Xiong/NML23-Project

Thank you

Xinyu Liu
Weijiang Xiong