Artist Similarity with Graph Attention Networks

Andrea Giuseppe Di Francesco
Dept. of Computer, Control and Management Engineering Antonio Ruberti
Sapienza University of Rome
Rome, Italy
difrancescoag00@gmail.com

Giuliano Giampietro
Dept. of Computer, Control and Management Engineering Antonio Ruberti
Sapienza University of Rome
Rome, Italy
giampietro.giuliano@gmail.com

***Abstract*—In this paper we show the effectiveness of Graph Neural Newtorks (GNNs) applied to the search for similarities within large collection of artists.**
The validity of this approach was earlier tested by [1] with the use of the GraphSAGE architecture which had already proved to be successful in various contexts [2], [3].
Since the artist similarities are easier to be modeled with a graph data structure, with respect to (w.r.t.) the non-grid-like datasets, we decided to test novel GNNs architectures in order to compare the different results obtained over the OLGA dataset [1]. These architectures include some of the state-of-the-art GNNs that were developed during the last decade.
Our purpose was to observe their performances on the artist similarity task and compare them with the already tested GraphSAGE layer.

I. Introduction

The artist similarity task consists into finding the most similar artist given a target artist, based on some aspects that relate them.
According to [4], there is not a clear and unique way of defining how two, or more artists could be similar, and which are the main characteristics that make them so. There could be culture-based aspects (e.g. their provenance, the meaning of their songs, the historical background etc.), or there also could be content-based reasons to argue about the similarity amongst artists (e.g. the common music progressions or the songs' keys [4]).
The ground-truth for artist similarity does not rely solely on content-based or culture-based aspects but rather on merging these together. Content-based aspects, extracted from elaborations of the music, are discretized and definite, thus easily handled in a scientific approach but cannot represent the common opinion concerning an artist. Furthermore, these widely accepted opinions are tough to encapsulate in numerical values but succeeding in doing so "can benefit from attempting to express innate non-acoustic and non-musical features about a specific piece of music" [4].
In our specific case we have decided to consider 2 or more artists as similar, based on their cultural aspects, but also by taking in consideration some aspects given by their music content. In particular every artist is encoded as an $n\text{-}dimensional$ vector, that captures the content-features of each artist. These vectors do represent the low-level features of the artist's music, or in other words they play the role of audio descriptors.
The procedure to obtain these vectors is explained in [1], the paper's authors use vectors of 2613 elements.
As ground-truth, to denote the binary artists relations, we considered the information extracted from the *OLGA* dataset [1]. These were selected according to human experts and/or the listeners feedback. From these we were able to design the graph and to conduct our experiments. Once defined the graph data structure and obtained the nodes' features we were able to design the GNNs architectures. At first, we tried the GraphSAGE configuration [2] as described in [1]. In particular, were objects of our study networks that comprised one, two, or three GraphSAGE layers.
Then we also tested other 4 architectures among which we obtained better results w.r.t those obtained in [1]. The common aspect amongst these configurations is the use of Graph layers together with fully connected layers. Our approach was an inductive one, namely we produce an embedding of artists such that (s.t.) their Euclidean distance is a synonym of music similarity. The embeddings are typically $d\text{-}dimensional$ vectors (where $d << n$).
At this point, it is easy to introduce new samples in the dataset and project them into the embedding space in order to assess the similarities. Since we have used different architectures (considering also the three GraphSAGE configurations), we have consequently obtained 7 different embeddings. Their quality has been evaluated through some metrics as shown in Section IV. We also decided to gauge the embeddings from a pragmatic point of view, in particular, given an artist, we have seen how much their $K\text{-}nearest\text{-}neighbors$ were pertinent to them in musicological terms. The results are shown in Figure 2. In three of our configurations, we have obtained better results than the GraphSAGE-based models, in particular, one of the Graph Attention Network-based (GATs) configurations outperformed them by 15% in accuracy.

II. The Olga Dataset

Olga is the dataset that we used to train our models, it was recently released for academic purposes by the same authors of [1], to perform the artist similarity task.
Although its sizes are limited, as already asserted by the authors, it is "significantly bigger than other datasets available for this task". In Table I are shown the differences of the main statistics measures between the original Olga Dataset and ours, indeed the data that we used for our experiments were reduced by 36.28% w.r.t. the original Olga because it

was not possible to recover all the information.

Olga description	Original	Ours
Avg n°of connections/artist	11.43	11.20
Tot. n° of connections	101,029	63,096
1st quartile	3	3
2nd quartile	7	8
3rd quartile	16	16

Table I
THESE ARE THE STATISTICS OF THE OLGA'S VERSION THAT WE USED FOR OUR EXPERIMENTS. THE STATISTICS ARE COMPARED WITH THE ORIGINAL OLGA VERSION, THAT WAS PRESENTED FOR THE FIRST TIME IN [1]. THE MEASURES ARE DIFFERENT BECAUSE OF THE DIFFERENT DATASET LENGTHS (ORIGINAL: 17,673, OURS: 11,261.)

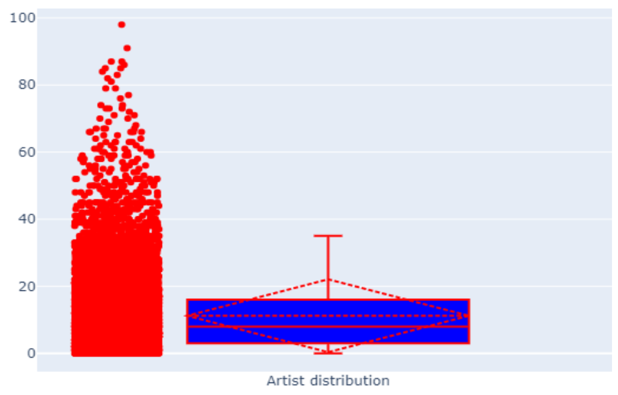

Figure 1. This box plot shows the number of connection per artist distribution in the OLGA dataset. In the left figure, each red point represent the number of relations for a given artist.

In Figure 1, there is a better visualization of our statistics measures. Even though there was a reduction in the data, we still have used the same splitting chosen by the authors, namely 80/10/10. The dataset contains information about the content of the artist's music and information about its connections with the other artists. The former is obtained in the procedure illustrated in [1] and represents the node features of the graph, the latter are obtained from the https://www.allmusic.com/ website, as described in the aforementioned paper, and are used to compute the adjacency matrix. It is noteworthy to say that the authors had also used a larger proprietary dataset for their experiments that was not available to us, in addition, they have shown that the GNN approach could obtain better results given the higher availability of data.

III. EXPERIMENTS

A. The architectures

In our experiments we first tried to replicate the authors' work in [1] by testing their three GraphSAGE configurations but with slight changes. Then we experimented with others Graph convolutional layers that were suitable for this task amongst some of the state-of-the-art GNNs. The main differences are the activation function in the graph layers and the aggregator function. As activation function we used the rectified linear unit (ReLU) instead of the exponential linear unit (ELU) [5] as in [1], nonetheless the ELU was applied at the end of each graph layer.
In the Equations 1, 2 we denote the difference between our aggregator function and the one used in the aforementioned paper.

$$\mathcal{N}^k \leftarrow \sigma(W^k \cdot MEAN(\{X^{k-1}_{\mathcal{V}_k}\} \cup \{X^{k-1}_{\mathcal{V}_{k-1}}\})), \tag{1}$$

$$\mathcal{N}^k \leftarrow \sigma(W^k \cdot X^{k-1}) \cdot A^k. \tag{2}$$

Here k denotes the index of a graph layer and it refers to the respective batch of artists, whereas k - 1 refers to its neighbors. $\mathcal{N}^k$ stands for the aggregated sample features which are the input to the merging function. $\mathcal{V}_i$ is the set of artists in the batch i, and $X^k_{\mathcal{V}_i}$ is the latent representation of the artists in the $\mathcal{V}_i$ set at the k-th graph layer. W^k stands for the trainable parameters for the k-th layer.
A is the adjacency matrix and A^k is its sub-matrix (s.t. $A^k = A[\mathcal{V}_{k-1}; \mathcal{V}_k]$) computed in order to map the relations between the artists at the k-1-th and the k-th layer. σ is the non-linear activation function that we chose to be the ReLU.
In the other architectures we have included other types of Graph Convolutional layers, such as the GCNConv layer that was presented in [6], and the GraphConv layer that was recently introduced in [7].
Our best performances were reached with the architectures that include the Graph Attention (GATs) layer, which was introduced in [8]. All the architectures are illustrated in Table V.
The GAT and SAGE configurations are both inductive approaches, in particular, they can generalize even unseen graphs, which is not the case for spectral-based models, such as the GCNConv layer, indeed they are stuck to specific information belonging to the training graph.
The models contain also a set of fully connected (FC) layers, either in the network back-end or in its front-end. In [1] was shown how the Graph layers increase their performances w.r.t architectures that use only multiple FC layers. Moreover, we have seen empirically that adding these simple layers generally leads to an increase in results before the projection in the d-$dimensional$ space.

B. Input features

The input features that we used to feed our models are those provided by [1]. These represent the low-level features for each artist which embeds the content sampled from their artworks. These attributes are represented by vectors of size 2613, according to the procedure described by the authors, and give information about "the loudness, dynamics and spectral shape of the signal, but they also include more abstract descriptors of rhythm and tonal information, such as bpm and the average pitch class profile" [1].

In our view, these low-level features may results noisy for this specific type of application indeed the graph structure is built-up taking into account the feedback from music experts and listeners about each artist. In addition, the evaluation metric is based solely on the graph topology. This means that it does not take into consideration the artist's attributes.
To assess the relevance of these features we have also compared our best models with their random input versions. The results are shown in Table IV.
The same experiments with random features were attempted by [1] and they have found that in the case of GraphSAGE networks there is a significant drop in performances, which we found to be coherent with our own GraphSAGE trials.

C. Hyperparameter tuning

Since our dataset is reduced w.r.t. the original OLGA it was not sufficient to use the same hyperparameters of the original work. As matter of fact, we have obtained rather different results.
Hence we have defined a new hyperparameter space and tried it with our networks. The main concern of our optimization was the learning rate, in particular, the values that we have evaluated are illustrated in Table II.

γ	10^{-5}	10^{-6}	10^{-7}	10^{-8}	10^{-9}	10^{-10}

Table II
HYPERPARAMETER SPACE FOR THE MODEL SELECTION. γ IS THE LEARNING RATE.

In Table III are listed the best setting for each network.

Model	γ
SAGE (one layer)	10^{-5}
SAGE (two layer)	10^{-9}
SAGE (three layer)	10^{-7}
Conf.1	10^{-6}
Conf.2	10^{-10}
Conf.3	10^{-6}
Conf.4	10^{-10}

Table III
THESE ARE THE LEARNING RATES SETTING THAT HAVE BETTER PERFORMED ON THE VALIDATION SET.

The weight decay term that we found to be the most fitting for the training was 0.01.
All the architectures were trained just for a few epochs then the accuracy started to slightly decrease. The size for mini-batches was 512 as in [2], whereas the adopted optimizer was ADAM [9]. At each epoch, the learning rate was decreased according to the cosine learning rate decay scheduler [10].
As regards to the loss function we have used for all the trainings the triplet loss function [11] in the same fashion as [1].
This method has already proved to be successful in different contexts [12] and it consists in assigning to each sample a pair of artists in which: one is similar to it (the positive one) and the other is unrelated to it (the negative one). The optimization of this loss function aims to minimize the distance between the sample and its positive while maximizing the distance from its negative. Basically the samples are seen as vectors in the latent space, thus in a high-quality embedding we do expect to see positive samples closer and dissimilar samples further in distance. We do select our triplets through the distance weighted sampling as described in [13].

D. Evaluation metrics

To evaluate our embeddings we used the normalized discounted cumulative gain (nDCG) metric [14].
We first compute the embedding of artists using all the connections between the training and test set, then, for each sample in the evaluation set, all the mutual distances are computed. The distances are ranked and, for each instance, we have its K-$nearest$-$neighbors$. If its neighbors are also connected to the target artist, then the prediction is recognized as a true positive. The nDCG for each sample is computed as follows:

$$nDCG_K(a, \hat{s}, s) = \frac{\sum_{k=1}^{K} g(\hat{s_k}, a) \cdot d(k)}{\sum_{k=1}^{K} g(s_k, a) \cdot d(k)}, \quad (3)$$

where:

$$d(k) = log_2^{-1}(k+1). \quad (4)$$

Where $g(i,k)$ is equal to one if artist i is similar to k in the graph, otherwise is 0. Namely, if the artists are similar the adjacency matrix would be s.t. $A[i,k] = 1$.
As we can see if a related artist is detected amongst the K-$nearest$-$neighbors$ it will be ranked based on its closeness to the considered artist (i.e. the closer it is, the higher is the accuracy). For our experiments, we set K equal to 200.

IV. RESULTS

The obtained results are listed in Table IV and plotted in Figure 2.
As previously mentioned our experiments on GraphSAGE are not comparable to the already tested networks for the artist similarity task. Nonetheless, we were able to show that by increasing the number of GraphSAGE layers, performances improved over the accuracy function as well. Compared to the other artist similarity with GNNs' work, we outperformed the previous best performing architecture by 15%. As shown in Table IV, [1] obtained the 55% with the nDCG metric, which was exceeded by our three best models, namely the Conf.3, Conf.4 and Conf.1 have gained respectively the following accuracy scores: 69%, 62%, 57%.
It is remarkable to notice that Conf.3 and Conf.4 are both GAT-based and so they are inductive networks. As in the case of the GraphSAGE layers, they attempt to sample the neighborhood features and aggregate them in an embedded representation, but in addition they follow also a self-attention mechanism, which means that each graph layer assigns attention scores to the neighbors of each sample. These scores

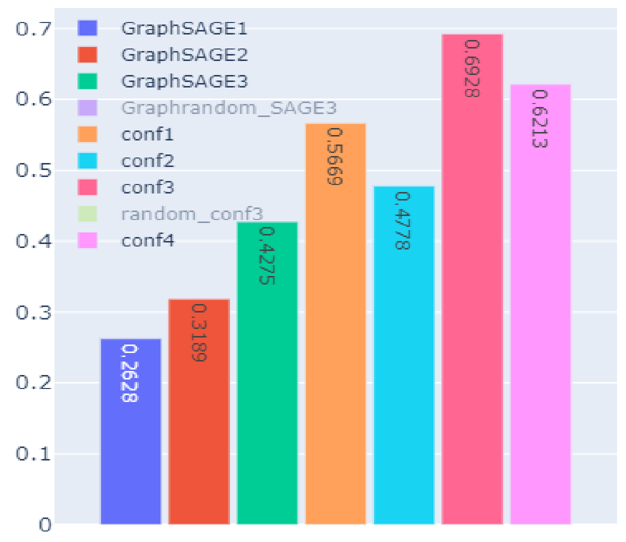

Figure 2. In this bar plot are displayed the different results obtained from our trained embeddings. The best configuration is the third one, which is better described in Table V

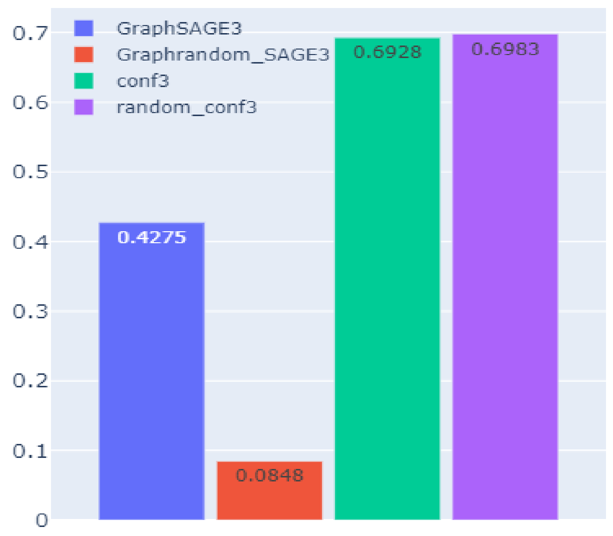

Figure 3. In this plot, we illustrate the performances of our best GAT network. It outperforms three GraphSAGE layers, and its performances appear to be robust also when it is fed with random input features.

Input	**Low level features**		**Random features**	
Dataset	**Original**	**Ours**	**Original**	**Ours**
SAGE (1 layer)	49%	26%	28%	—
SAGE (2 layer)	53%	32%	42%	—
SAGE (3 layer)	55%	43%	45%	8%
Conf.1	—	57%	—	—
Conf.2	—	48%	—	—
Conf.3	—	**69%**	—	**70%**
Conf.4	—	62%	—	—

Table IV
THIS IS THE COMPARISON OF OUR RESULTS WITH THE ONES OBTAINED BY [1], TOGETHER WITH OUR NOVEL ARCHITECTURES.
CONF.3 IS ABLE TO OBTAIN BETTER RESULTS W.R.T THE AUTHORS' SAGE NETWORKS.

determine the hidden representation of each instance by giving the right attention to its connections in the graph. Conversely from GraphSAGE, for each edge in the graph are computed different attention scores. As asserted by the authors of [8] the attention scores lead to some possible interpretability of the model. The other characteristic of the GAT-based configuration is that the neighborhood is sampled in its entirety which is not the case in GraphSAGE where the neighborhood size is fixed. Since the training is not so computationally expensive, using a non-fixed size gives a higher prediction power without affecting the hardware resources. To test the robustness of our best architecture we decided to use as input a random vector of features instead of the content-based vector in order to see if it would affect our model's performances.
As depicted in Figure 3 GraphSAGE network suffers due to the different inputs given, but the GAT-based architecture still keeps the same results. It is interesting to see how the latter stays flexible despite the random features.

This is a convenient property because it is required less effort to represent the input artist. In fact to get the low-level features must be followed a specific procedure which could be avoidable with our approach. Moreover, we can notice that if GraphSAGE loses accuracy without the hand-crafted attributes, it means that it relies mainly on those. In our case we can infer that our method depends more on the graph topology, indeed the attention scores are trained on the graph's edges. Although we have used a different dataset, the effectiveness of the low-level features was also clear in [1]. Taking these last statements into account, we strongly think that with the Conf.3 model applied to the original OLGA dataset our result would remain the same even with the random feature setting. Another important aspect that we would like to stress is that this achievement could lead to a reconsideration of the task's nature.
Let us assume that we can choose the instance matrix arbitrarily, then it would be relevant to the model just the graph structure, in other words the artist similarity task could be addressed in an unsupervised manner.

$$ENC_{\Theta} : X, A \rightarrow Z \qquad (\textit{Supervised learning}), \qquad (5)$$

$$ENC_{\Theta} : A \rightarrow Z \qquad (\textit{Unsupervised learning}). \qquad (6)$$

Where ENC is the model, X, and A are respectively the instance and the adjacency matrix, while Z is the embedding produced by the encoder.
In addition to the accuracy metric, we have decided to evaluate the embeddings quality by assessing its recommendation efficacy. In particular, we have chosen several well-known artists and look for their K-$nearest$-$neighbors$. Some of these experiments are reported in Tables VI, VII, VIII, IX, X.
We can notice that the quality of the recommendation is proportional to the accuracy score obtained by each model.

V. Conclusion

In this paper we illustrated how GNNs are effective in the artist similarity task. This new approach had already been

introduced in [1], but we have also decided to experiment with 4 GNNs architectures to further verify the validity of Graph Convolutional (GC) layers in modeling relationships between artists. We observed that GATs networks outperformed GraphSAGE-based architectures and were more flexible with respect to input attributes.
We have concluded that they rely more on graph data structure than node features. Our method could easily create large-scale datasets, by augmenting those currently available, indeed to introduce new samples, it would be sufficient to specify some similar ground-truth artists to project the sample into the latent space. Another possible approach could be the improvement of the features' quality and the simplification of their extraction. The former could lead to the detection of more patterns and low-level information from the data, while the latter could make the use of the supervised method more preferable in terms of time.
We firmly believe that improving accuracy scores with these GC networks is possible and would help to obtain even deeper and more detailed music recommendations.

References

[1] F. Korzeniowski, S. Oramas, and F. Gouyon, "Artist similarity with graph neural networks," *CoRR*, vol. abs/2107.14541, 2021. [Online]. Available: https://arxiv.org/abs/2107.14541

[2] W. L. Hamilton, R. Ying, and J. Leskovec, "Inductive representation learning on large graphs," *CoRR*, vol. abs/1706.02216, 2017. [Online]. Available: http://arxiv.org/abs/1706.02216

[3] R. Ying, R. He, K. Chen, P. Eksombatchai, W. L. Hamilton, and J. Leskovec, "Graph convolutional neural networks for web-scale recommender systems," *CoRR*, vol. abs/1806.01973, 2018. [Online]. Available: http://arxiv.org/abs/1806.01973

[4] D. Ellis, B. Whitman, A. Berenzweig, and S. Lawrence, "The quest for ground truth in musical artist similarity." 01 2002.

[5] D.-A. Clevert, T. Unterthiner, and S. Hochreiter, "Fast and accurate deep network learning by exponential linear units (elus)," *arXiv preprint arXiv:1511.07289*, 2015.

[6] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," *CoRR*, vol. abs/1609.02907, 2016. [Online]. Available: http://arxiv.org/abs/1609.02907

[7] C. Morris, M. Ritzert, M. Fey, W. L. Hamilton, J. E. Lenssen, G. Rattan, and M. Grohe, "Weisfeiler and leman go neural: Higher-order graph neural networks," *CoRR*, vol. abs/1810.02244, 2018. [Online]. Available: http://arxiv.org/abs/1810.02244

[8] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio, "Graph attention networks," 2017. [Online]. Available: https://arxiv.org/abs/1710.10903

[9] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[10] I. Loshchilov and F. Hutter, "SGDR: stochastic gradient descent with restarts," *CoRR*, vol. abs/1608.03983, 2016. [Online]. Available: http://arxiv.org/abs/1608.03983

[11] E. Hoffer and N. Ailon, "Deep metric learning using triplet network," in *International workshop on similarity-based pattern recognition*. Springer, 2015, pp. 84–92.

[12] J. Wang, Y. Song, T. Leung, C. Rosenberg, J. Wang, J. Philbin, B. Chen, and Y. Wu, "Learning fine-grained image similarity with deep ranking," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2014, pp. 1386–1393.

[13] C.-Y. Wu, R. Manmatha, A. J. Smola, and P. Krahenbuhl, "Sampling matters in deep embedding learning," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 2840–2848.

[14] K. Järvelin and J. Kekäläinen, "Cumulated gain-based evaluation of ir techniques," *ACM Transactions on Information Systems (TOIS)*, vol. 20, no. 4, pp. 422–446, 2002.

SAGE (1,2,3 layers)	**Conf.1**	**Conf.2**	**Conf.3**	**Conf.4**
SAGEConv(2613,256)	FC(2613,256)	GCNConv(2613,256)	FC(2613,256)	GATConv(2613,256)
SAGEConv(256,256) [1]	FC(256,256)	GraphConv(256,256)	FC(256,256)	GATConv(256,256)
SAGEConv(256,256)[2]	GCNConv(256,256)	GCNConv(256,256)	FC(256,256)	FC(256,256)
FC(256,256)	GCNConv(256,256)	GCNConv(256,256)	GATConv(256,256)	FC(256,256)
FC(256,256)	-	FC(256,256)	GATConv(256,256)	-
FC(256,100)	-	FC(256,256)	-	-

Table V

IN THIS TABLE ARE DISPLAYED THE 7 ARCHITECTURES THAT WERE USED FOR THE EXPERIMENTS, WRITTEN IN THE FORM: *layer name(input size,output size)*. SAGE NETWORKS RETURN 100-DIMENSIONAL EMBEDDINGS, WHEREAS OURS RETURN 256-DIMENSIONAL EMBEDDINGS.

Conf.3					
Snoop Dogg		**Nancy Sinatra**		**Rod Stewart**	
Artist name	**Distance**	**Artist name**	**Distance**	**Artist name**	**Distance**
2Pac	0.1406	Dusty Springfield	0.1921	Fleetwood Mac	0.1170
Tha Dogg Pound	0.1415	Cilla Black	0.2103	Ringo Starr	0.1201
Mack 10	0.1471	Harpers Bizarre	0.2162	Phil Collins	0.1244
Luniz	0.1524	Helen Shapiro	0.2193	Dire Straits	0.1410
Benzino	0.1555	Sonny & Cher	0.2256	Paul Carrack	0.1430

Table VI

THESE ARE THE 5-*nearest-neighbors* FOR THESE ARTISTS. CONF.3 HAS REACHED THE 69% IN THE OLGA DATASET.

Conf.3 (random)					
Snoop Dogg		**Nancy Sinatra**		**Rod Stewart**	
Artist name	**Distance**	**Artist name**	**Distance**	**Artist name**	**Distance**
Luniz	0.1744	Dusty Springfield	0.2107	Paul McCartney	0.1334
Tha Dogg Pound	0.1768	Cilla Black	0.2116	Steve Winwood	0.1357
Spice 1	0.1833	Harpers Bizarre	0.2116	Phil Collins	0.1379
Warren G	0.1867	Air Supply	0.2141	Fleetwood Mac	0.1387
Ice Cube	0.1913	The 5th Dimension	0.2151	Bruce Hornsby	0.1389

Table VII

THESE ARE THE 5-*nearest-neighbors* FOR THESE ARTISTS. CONF.3 (RANDOM) HAS REACHED THE 70% IN THE OLGA DATASET.

Three Layers GraphSAGE					
Snoop Dogg		**Nancy Sinatra**		**Rod Stewart**	
Artist name	**Distance**	**Artist name**	**Distance**	**Artist name**	**Distance**
Ice Cube	0.0266	Bobby Vinton	0.0367	Argent	0.0355
Kurupt	0.0266	Ferlin Husky	0.0383	The Doobie Brothers	0.0406
Petey Pablo	0.0268	Dusty Springfield	0.0393	Levon Helm	0.0406
Obie Trice	0.0277	Tammy Wynette	0.0394	Delaney & Bonnie	0.0408
2Pac	0.0279	Irma Thomas	0.0394	The Moody Blues	0.0415

Table VIII

THESE ARE THE 5-*nearest-neighbors* FOR THESE ARTISTS. THREE LAYERS GRAPHSAGE HAS REACHED THE 43% IN THE OLGA DATASET.

Three Layers GraphSAGE (random)					
Snoop Dogg		**Nancy Sinatra**		**Rod Stewart**	
Artist name	**Distance**	**Artist name**	**Distance**	**Artist name**	**Distance**
Lars Winnerbäck	0.0555	Chris Spheeris	0.0537	The Box Tops	0.0475
The Afghan Whigs	0.0581	Chris Connor	0.0544	OneRepublic	0.0484
Bill Wyman's Rhythm Kings	0.0585	Divinyls	0.0549	Guns N' Roses	0.0493
Marisa Monte	0.0585	Lydia Lunch	0.0553	Big Wreck	0.0501
Fukkk Offf	0.0587	Sophie B. Hawkins	0.0553	MV & EE	0.0511

Table IX
THESE ARE THE 5-*nearest-neighbors* FOR THESE ARTISTS. THREE LAYERS GRAPHSAGE (RANDOM) HAS REACHED THE 8% IN THE OLGA DATASET.

Conf.4					
Snoop Dogg		**Nancy Sinatra**		**Rod Stewart**	
Artist name	**Distance**	**Artist name**	**Distance**	**Artist name**	**Distance**
Tha Dogg Pound	0.3131	Cilla Black	0.3588	Ringo Starr	0.3591
Luniz	0.3823	Sonny & Cher	0.4853	Steve Winwood	0.3985
Mack 10	0.3872	The Mamas & the Papas	0.5645	The Doobie Brothers	0.4132
Eminem	0.4575	Adam Faith	0.6053	Tom Petty	0.4228
Fler	0.4586	Bobbie Gentry	0.6174	Dave Mason	0.4556

Table X
THESE ARE THE 5-*nearest-neighbors* FOR THESE ARTISTS. CONF.4 HAS REACHED THE 62% IN THE OLGA DATASET.