Identifying clusters using (Node2Vec Embedding, Spectral, and GCN) embeddings

## Ashish Kumar

EM23MTECH14005

[em23mtech14005@iith.ac.in](mailto:em23mtech14005@iith.ac.in)

# Abstract

*In this assignment, we use node2vec, spectral, and GCN embeddings to identify the clusters from the given data. Finding clusters in a financial transaction network is an important undertaking that can provide important informa- tion about the connections and patterns among different fi- nancial system components. By clustering network nodes according to their commonalities, this operation can iden- tify closely related groups of things, such as accounts, ven- dors, or consumers. In this work, we employ three dis- tinct techniques to embed nodes in the network: Graph Convolutional Networks (GCN), Spectral Embedding, and Node2Vec. The nodes are then clustered using these em- beddings to find patterns in the payment data.*

# Introduction

Clustering is a technique used to group similar nodes in a network, revealing communities or clusters of closely con- nected entities. This study aims to identify financial transac- tion network clusters using three node embedding methods: Node2Vec, Spectral Embedding, and Graph Convolutional Networks (GCN). Each method has its strengths in captur- ing network structures and relationships, making them valu- able tools for analyzing payment data.

* Node2Vec Embedding: Node2Vec is a random walk- based algorithm that learns low-dimensional represen- tations of nodes by simulating biased random walks on the network. It captures structural relationships and node similarities through these walks.
* Spectral Embedding: Spectral Embedding uses the graph’s Laplacian or adjacency matrices to project nodes onto a lower-dimensional space. This approach

effectively captures the underlying network structure and is known for its ability to identify clusters in the network.

* Graph Convolutional Networks (GCN) Embedding: GCNs apply neural network layers to the graph’s ad- jacency matrix and node features, learning meaningful node embeddings that represent relationships and pat- terns in the network.

The git-hub repository for this project can be found [here](https://github.com/Satyaditi/EE6310-Image-And-Video-Processing-Project/tree/main/Final_Report).

# Problem Statement

Identifying clusters using (Node2Vec Embedding, Spec- tral, and GCN) embeddings

# Data Description

The dataset consists of three columns: Sender, Receiver, and Amount.

* Sender: This column represents the entity that initi- ated the transaction. The dataset includes a total of 130,535 transactions involving different senders. The mean sender ID is approximately 158, with a standard deviation of around 157. Sender IDs range from 0 to 702.
* Receiver: This column represents the entity that re- ceived the transaction. Similar to the Sender column, there are 130,535 transactions involving different re- ceivers. The mean receiver ID is approximately 139, with a standard deviation of around 109. Receiver IDs range from 0 to 370.
* Amount: This column represents the monetary value of the transactions. The average transaction value is approximately 69,810, with a standard deviation of

around 56,967. Transaction values range from 1,501 to 2,124,500.

The data indicates that the distribution of sender and re- ceiver IDs covers a wide range, with varying transaction amounts. This dataset provides a broad view of financial in- teractions between different entities, making it suitable for analysis and clustering to identify patterns, relationships, and potential anomalies within the network.

# Implementation

1. Graph Construction:The first step involves construct- ing a directed graph (G) from the input data (df) us- ing NetworkX’s from pandas edgelist() function. The function accepts a DataFrame as input and the columns for the source (Sender), target (Receiver), and edge at- tribute (Amount). This graph will be used for the sub- sequent analysis.
2. Node2Vec Embeddings: Node2Vec embeddings are generated using the Node2Vec algorithm from the graph G. The Node2Vec algorithm performs biased random walks on the graph, and the resulting se- quences of nodes are used to train a Word2Vec model. The learned node embeddings are stored in the node embeddings array.
3. Spectral Embeddings: This is implemented both for directed and undirected graphs. for undirected graph Laplacian, eigen values and vectors are used for com- puting the embeddings.
4. GCN Method: The input to the model consists of: x: The feature matrix of the graph, where each row rep- resents a node and each column represents a feature. edge index: The adjacency information of the graph, represented as a list of tuples indicating the connec- tions between nodes. The model architecture consists of two layers of graph convolutions. This layer per- forms a graph convolution operation on the input fea- tures (x) and graph adjacency (edge index) and the next layer performs another graph convolution oper- ation on the output of the first layer.
5. Elbow Method:This method is used to find the opti- mal number of clusters for the KMeans algorithm. By computing the inertia (sum of squared distances) for different numbers of clusters, an ”elbow plot” is gen- erated. This plot helps to identify the optimal number of clusters by looking for the ”elbow” point.
6. KMeans: The KMeans clustering algorithm is applied to the Node2Vec embeddings. The nodes are grouped into n clusters clusters based on their similarity in the embedding space. Cluster assignments are stored in

the labels array. The code then constructs a dictio- nary of clusters, mapping each cluster label to the corresponding nodes. DBSCAN: The DBSCAN al- gorithm is used as an alternative clustering approach. The algorithm uses eps and min samples parameters to group nodes based on their proximity in the embedding space. DBSCAN identifies clusters as well as noise points, which are points that do not belong to any clus- ter. The clustering results are stored in the labels array.

1. Visualization: After clustering, the code uses the TSNE algorithm to reduce the dimensionality of the node embeddings to 2D for visualization. The clus- tering results are visualized using scatter plots, where different colors represent different clusters, and noise points are highlighted if DBSCAN is used.

# Results and Plots

### Clustering using Node2vec Embedding

Figure-1 shows the node2vec embedding-based cluster- ing with two methods: Kmeans and BDSCAN. The number of clustering for Kmeans is computed by fitting the embed- dings into the KMean. This clustering will help us find the anomalies and patterns between the nodes.

### Clustering using Spectral Embedding

Figure-2 shows the spectral embedding method result. This is computed by considering the undirected graph as the directed graph is not very effective in producing the clusters here.

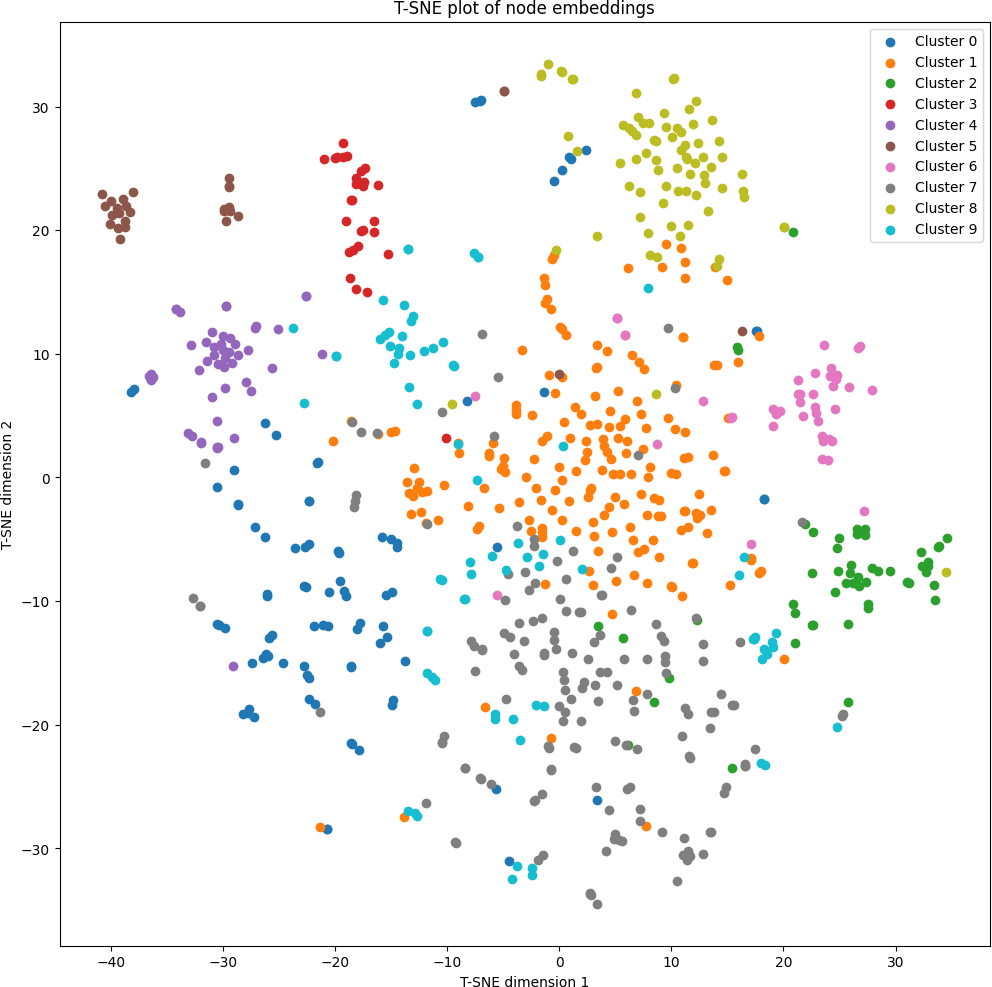
### Clustering using GCN Embedding

Figure-3 shows the GCN embedding method outcome.

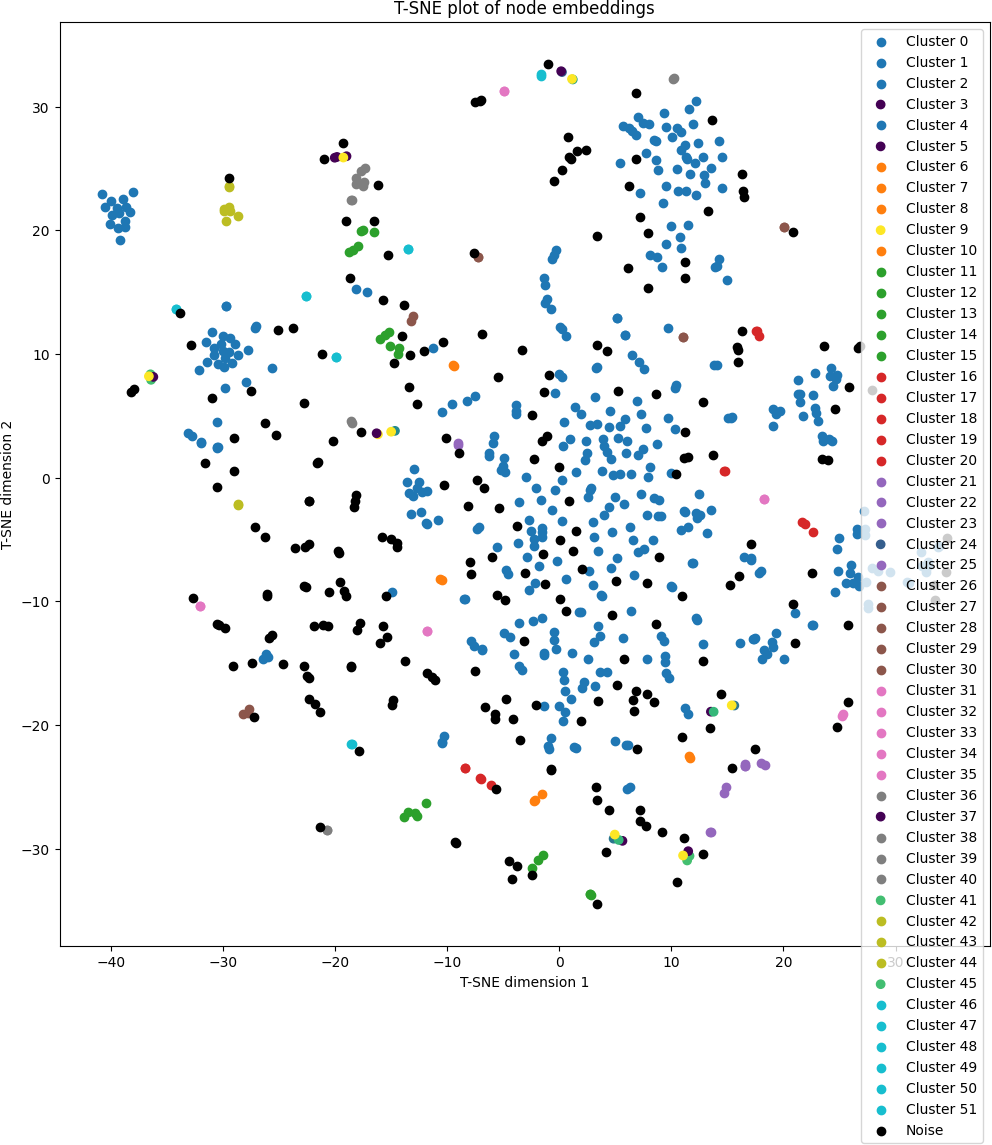
We can conclude that node2vec and GCN embedding maps are significantly better than the spectral maps.

# References

[https://snap.stanford.edu/class/cs224w-](https://snap.stanford.edu/class/cs224w-2017/projects/cs224w-38-final.pdf) [2017/projects/cs224w-38-final.pdf](https://snap.stanford.edu/class/cs224w-2017/projects/cs224w-38-final.pdf)

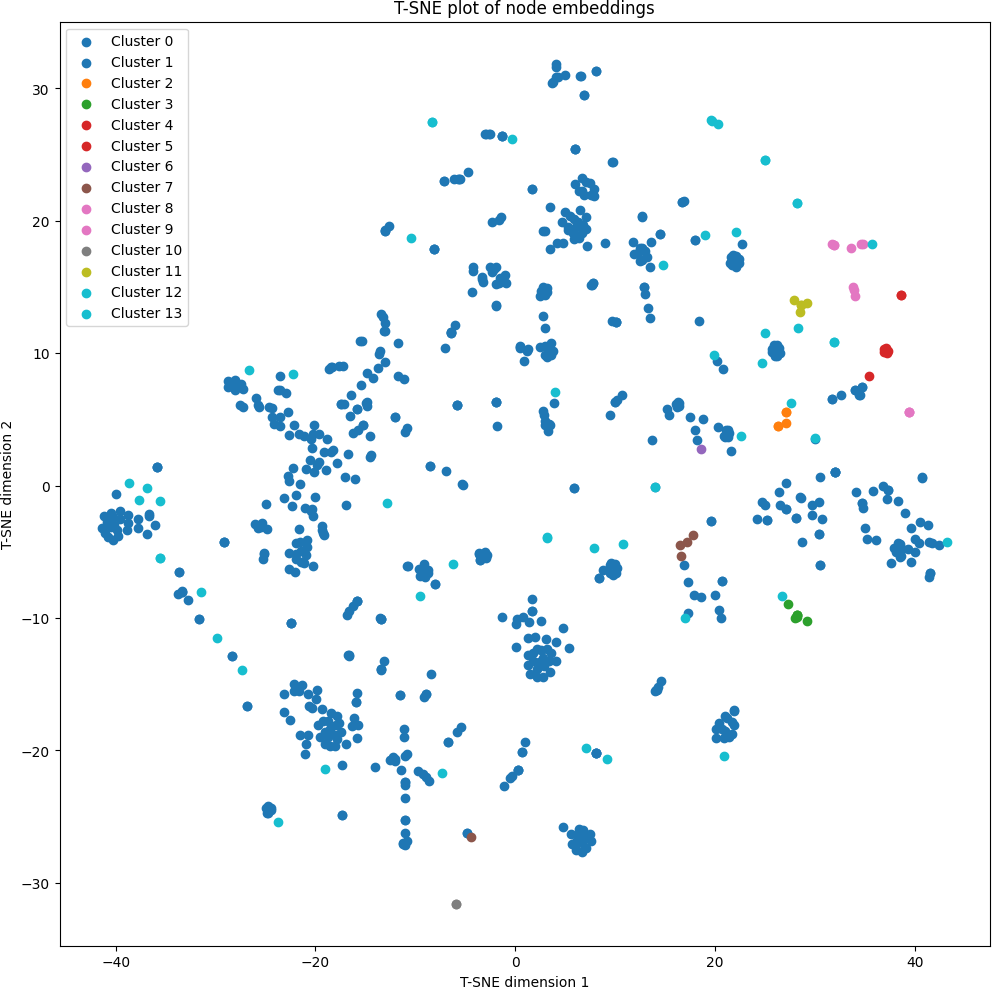


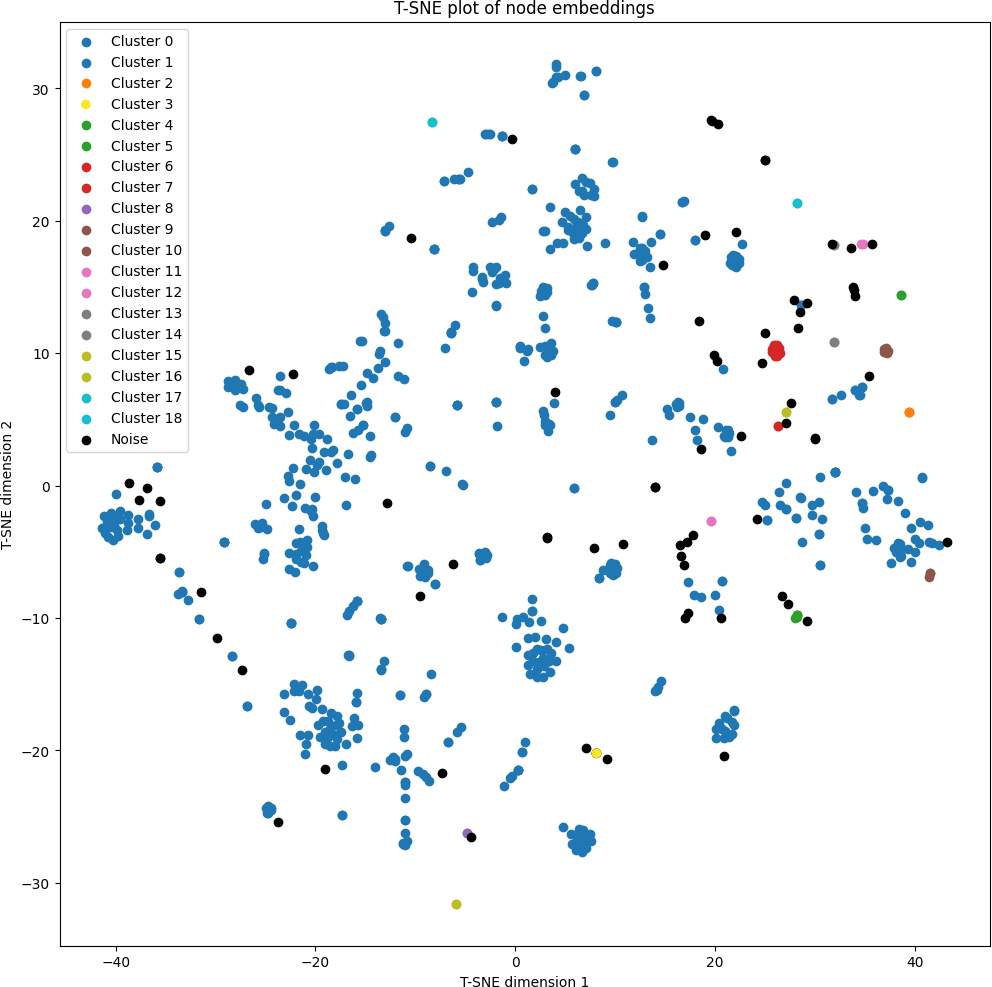
1. Node2vec clustering using kmeans.



1. Node2vec clustering using DBSCAN.

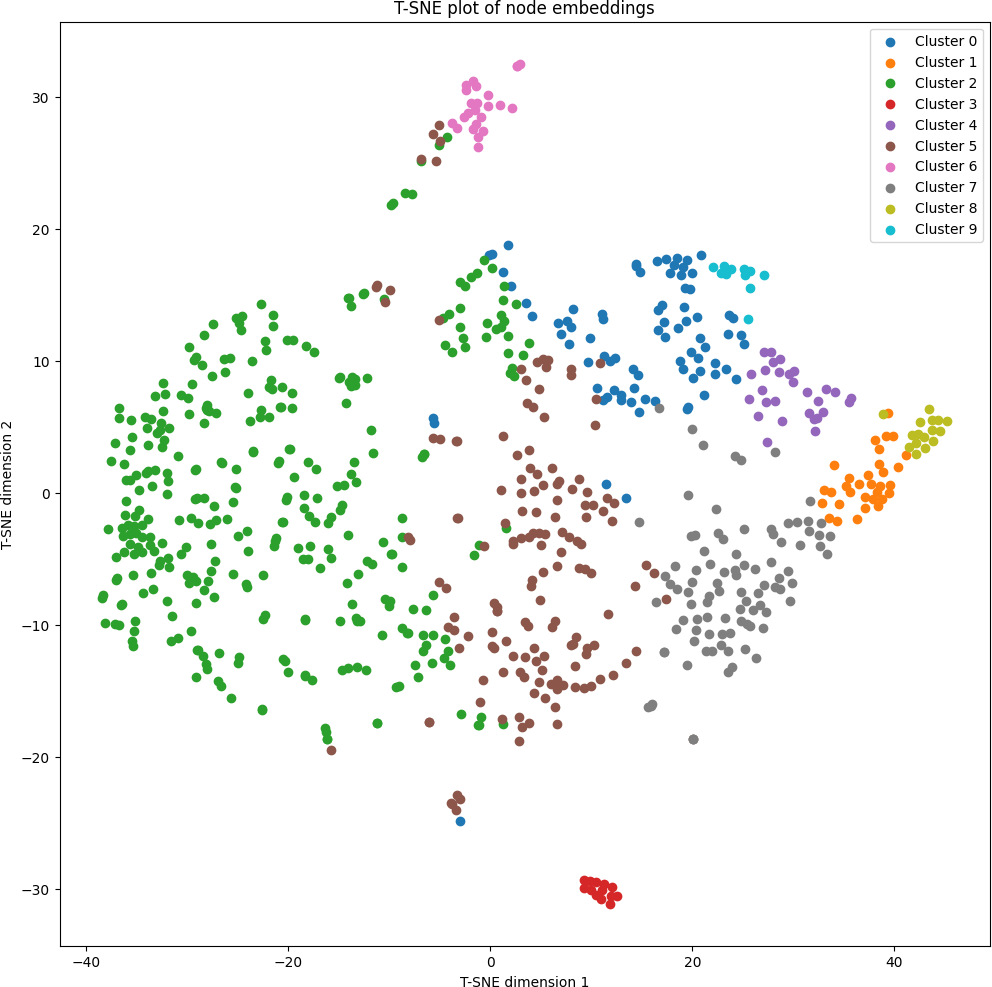
Figure 1. Node2Vec clustering

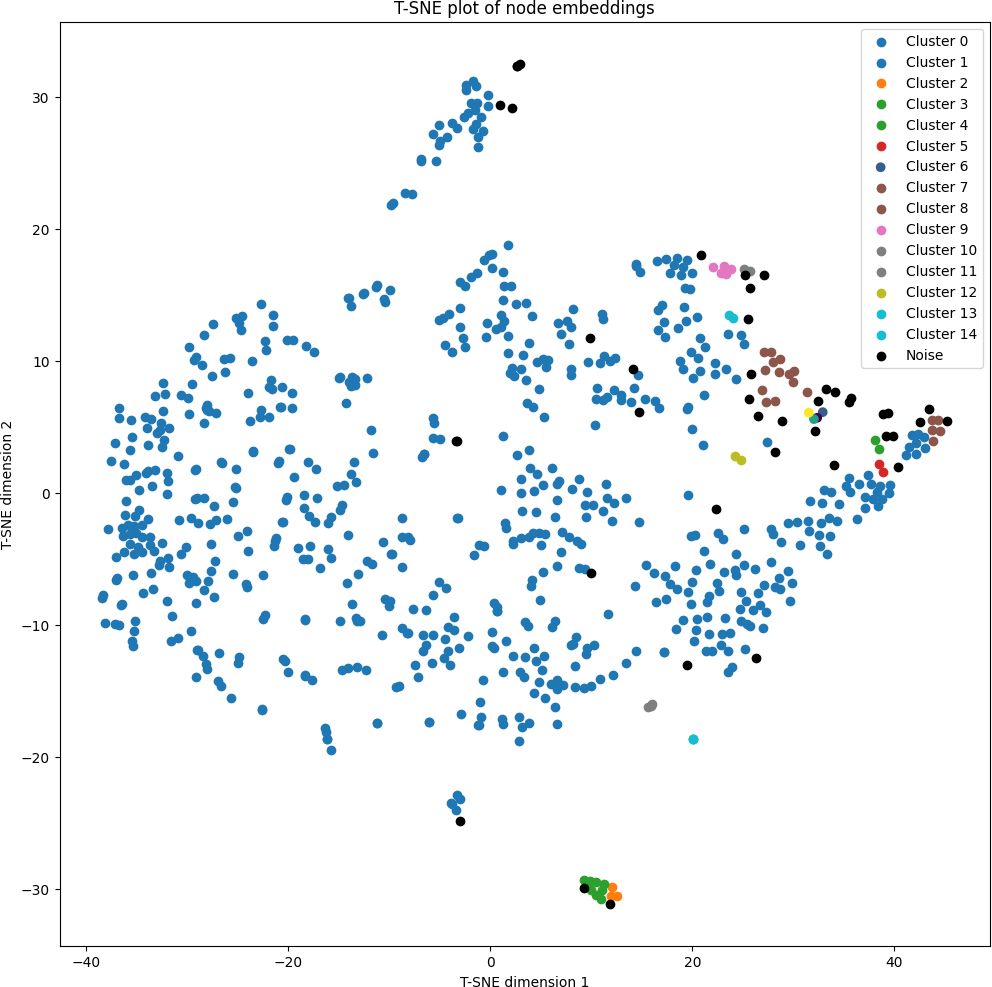
* 1. Spectral clustering using kmeans.



* 1. Spectral clustering using DBSCAN.

Figure 2. Spectral clustering

* + 1. GCN clustering using kmeans.



* + 1. GCN clustering using DBSCAN.

Figure 3. GCN clustering