Report on Gemsec Deezer Dataset *

*Note: Conducted an analysis of a Gemsec Deezer dataset(Stanford Large Network Dataset SNAP)

1st Vansh Lal Tolani
DSAI
IIIT DHARWAD
22bds061@iiitdwd.ac.in

2nd R Niharika
DSAI
IIIT DHARWAD
22bds050@iiitdwd.ac.in

3rd Dhrithi K
DSAI
IIIT DHARWAD
22bds018@iiitdwd.ac.in

4th G Leeladitya
DSAI
IIIT DHARWAD
22bds024@iiitdwd.ac.in

***Abstract*—In this study, the four algorithms Prims, Kruskal, Bellman Ford, and Dijkstra are compared and discussed.For Minimum Spanning Tree, Prims and Kruskal , and for Bellmanford and Dijkstra for shortest pathways.The four procedures mean, median, maximum, and minimum will be found out from the multigraph i.e Gemsec Deezer.**

I. Introduction

We will be using four datasets namely Prims,Kruskal,Bellmanford,Dijkstra on our dataset which is a multi layer graph to perform following operations such as mean,median,maximum and minimum. We have even analysed GEMSEC Algorithm which is used in research paper.GEMSEC is a sequence sampling-based learning model that learns node embeddings and node clustering at the same time.

II. Description of Datasets

A. Multi-Layer Travelling Network Dataset

Here we are taking three undirected graphs(G1,G2,G3) which is a part of transport multilayer network and we are finding the mean,mediun,maximum and minimum of the three MSTs of the Graph.Nodes represent cities and edges represent the distance between two cities.Where G1 represents trainways,G2 represents roadways and G3 represents railways.For convicence we take these as undirected graphs.

B. Gemsec Deezer Dataset

The information was gathered in November 2017 from the Deezer music streaming service. These statistics depict the social networks of users from three different European nations. Users are modeled as nodes, while mutual follower relationships are shown as edges. To achieve a certain level of anonymity, we reindexed the nodes. Based on the preferred song lists, favored genre lists were created. Hungary, Croatia, and Romania are the nations that are featured. We listed the number of nodes and edges for each dataset. The growth dynamics are highly influenced by the topology of the graph clusters, node embeddings, and how influencers connect with the rest.

The GEMSEC model was trained using real-world social-network datasets.

Identify applicable funding agency here. If none, delete this.

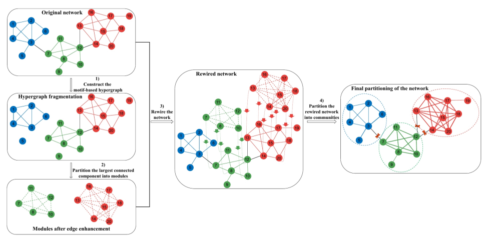
Original network
1) Construct the motif-based hypergraph
Hypergraph fragmentation
2) Partition the largest connected component into modules
Modules after edge enhancement
3) Rewire the network
Rewired network
4) Partition the rewired network into communities
Final partitioning of the network

Fig. 1.

Deezer user-user friendship networks: Friendship networks from Deezer, a music streaming service.

After running the GEMSEC graph embedding algorithm, the two datasets are visualized by the top 3 principal components.

TABLE I
Gemsec Deezer Dataset information

Counrty	***Nodes***	***Edges***
Romania	41773	125826
Croatia	54573	498202
Hungary	47538	222887

III. Analysis of Dataset

Here we are doing statistical measures such as mean,median,maximum,minimum on our dataset MSTs and Shortest paths

A. Multilayer Travelling Network

Properties of Minimum Spanning Tree

Average MST: It gives us the average time/distance taken to reach from one city the other city.

Median MST: It gives us the average time/distance taken to reach the other city.

Maximum MST: It provides us the maximum time/distance taken to reach the other city.

Minimum MST: It displays us the minimum time/distance taken to reach the other city.

Shortest Paths

Average Travelling Time:It provides us the average time/distance to travel from one city to another using any of travelling ways.

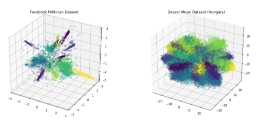

Fig. 2.

Maximum Travel Path/Time:It provides us the maximum time/distance to travel from one city to another using any of travelling ways.It happens in the the worst scenario where the distance is too long and it takes longer time to reach another city.

Minimum Travel Path/Time): It provides us with the shortest paths and shortcuts without spending much time to travel from one city to another in a network.

Median Travel Path/Time: If the value is above the median it suggests that it takes long time and distance to travel through any transport from one city to another. f the value is below the median it suggests that it takes lesser time and distance to travel through any transport from one city to another.

B. Gemsec Deezer Dataset

Analysis on Prims and Kruskal algorithm

Average users: It tells us the average number of mutual users of a particular country following a music streaming service.

Median users: If the median value of the user is above the median level it shows that many people are intrested for a particular music streaming service of a particular country.

Maximum users: If tells that maximum number of mutual users follow a particular music streaming service in a particular country.

Minimum users: If tells that minimum number of mutual users follow a particular music streaming service in a particular country.

Analysis on BellmanFord and Dijkstra algorithm

Average songs: It tells us the average number of songs that are heard by mutual users in a particular country.

Median songs: If the value of median is above average songs then many mutual users are listening to a particular song in a particular country.If the value of median is below average then very few mutual users are listening to a particular song in a particular country.

Maximum songs: It tells us the maximum number of songs that are heard by mutual users in a particular country.

Minimum songs: It tells us the minimum number of songs that are heard by mutual users in a particular country.

TABLE II
DATASET VALUES

Algorithms	***Mean***	***Median***	***Maximum***	***Minimum***
Prims	47690.33	47537	54572	41772
Kruskal	47690.33	47537	54572	41772
BellmanFord	5.17	5	16	0
Dijkstra	5.17	5	16	0

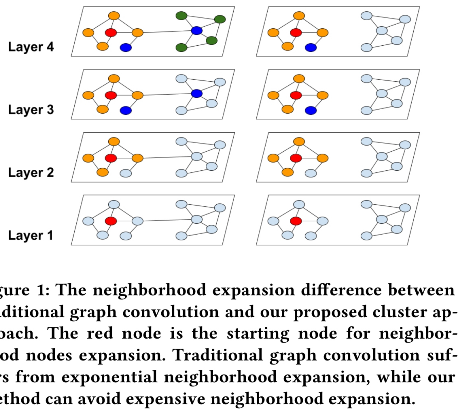

Figure 1: The neighborhood expansion difference between traditional graph convolution and our proposed cluster approach. The red node is the starting node for neighborhood nodes expansion. Traditional graph convolution suffers from exponential neighborhood expansion, while our method can avoid expensive neighborhood expansion.

Fig. 3.

TABLE III
MULTINETWORK DATSET VALUES

Algorithms	***Mean***	***Median***	***Maximum***	***Minimum***
Prims	10.67	10	12	10
Kruskal	10.67	10	12	10
BellmanFord	4	3	10	0
Dijkstra	4	3	10	0

ACKNOWLEDGMENT

I would like to extend our sincere thanks to our Assistant Professor Dr.Animesh Chaturvedi for providing us guidance throughout our project.This project has increased our technical knowledge

CONCLUSION

In summary, our work contributed to the development of GEMSEC, a machine learning model that handles both network embedding and community detection. GEMSEC outperforms existing methods in clustering by using smoothness regularization and taking into account the social community structure. In addition, additional social network datasets are given, and experimental results demonstrate GEMSEC's efficiency in embedding quality, cluster quality, and recommendation tasks.

REFERENCES

[1] https://snap.stanford.edu/data/gemsec-Deezer.html
[2] https://algs4.cs.princeton.edu/40graphs/
[3] https://towardsdatascience.com/https-towardsdatascience-com-the-elements-of-viral-growth-7f364aec64eb
[4] https://github.com/Niharika0804/Lab11DSA.git
[5] https://github.com/Niharika0804/DSA-Project-Case1.git
[6] https://github.com/Niharika0804/DSA-Project-Case2.git

LAB 11 Questions

Question 1: We are supposed to maintainn a linked list of student records. We need to create, traverse, insert, delete a linked list.

This program is intended to maintain a linked list of student records. It employs the ”student” struct to hold information on each student, such as roll number, name, and age. A typedef ”node” is also used to define a node in the linked list.

(1) Creating a Linked List: The program begins by allocating memory dynamically for each node in the linked list and fills it with student data entered by the user via the createlist() function.

(2) Displaying the Linked List: The display() function is used to display the contents of the linked list, i.e., student records, in tabular format.

(3) Inserting a new student record: Using the insert() function, the user can create a new node (student record) with data before a particular roll number. If the supplied roll number is not found in the linked list, the new node is appended to the end.

(4)Deleting a student record: The delete() function allows the user to remove a node from the linked list with a given roll number.

The Inputs given would be

How many elements to enter: 3

Enter the rollno, name and age of the student 1:

10 Ash 17

Enter the rollno, name and age of the student 2:

18 Brook 18

Enter the rollno, name and age of the student 3:

21 Misty 17

Data to be inserted:

20 Harry 19

Insert the added data before rollno (-ve for end):21

Delete the data or replace the Rollno :18

The Outputs we obtain would be

Rollno Name Age

Student 1: 10 Ash 17 Student 2: 18 Brook 18 Student 3: 21 Misty 17

Rollno Name Age Student 1: 10 Ash 17 Student 2: 20 Harry 19 Student 3: 21 Misty 17

Question 2: a) Concatenate two given list into a big list.

- The program works with singly linked lists and allows you to create, display, and concatenate them. It specifies a struct Node to represent each linked list node, with each node containing an integer value and a pointer to the next node.
- The Create function is in charge of constructing a linked list from an array of numbers. It takes as input the array A of size n and dynamically allocates memory for each node while linking them together to produce the linked list.
- The Display function prints the elements of a specified linked list.
- Concate accepts two linked lists as arguments, head1 and head2, and appends the second linked list (head2) to the end of the first linked list (head1). It accomplishes this by traversing the first linked list to its last node and then connecting that node to the beginning node of the second linked list. Following the concatenation, it sets the head of the second linked list (head2) to NULL, resulting in an empty list.
- The user is invited to enter the size of the two linked lists, n and m, in the main function. The user is next prompted to enter the elements of both connected lists, A and B. The Create function is called twice to generate the two linked lists, head1 and head2, from the elements that have been entered. Following that, the Display function is used to display the elements of both linked lists.
- The Concate function is then run to concatenate the second linked list (head2) to the end of the first linked list (head1). The Display function is used to display the concatenated linked list once it has been concatenated.

The Inputs given would be like

Enter the size of first linked list 6

Enter the size of second linked list 8

Enter the elements of the first linked list 8 4 1 9 9 0

Enter the elements of the second linked list 1 8 1 2 2 0 1 7

The Outputs obtained would be like

1st Linked List: 8 4 1 9 9 0

2nd Linked List: 1 8 1 2 2 0 1 7

Concatenated Linked List: 8 4 1 9 9 0 1 8 1 2 2 0 1 7

b) Insert an element in a linked list in sorted order

- First, we will see how the insertion function works
- To maintain the list sorted, this code uses a singly linked list data structure and an insertion sort algorithm. The struct Node represents the linked list, with each node holding an integer data value and a pointer to the next node.
- The main function assigns data values 10, 15, and 20 to three nodes (head, node2, and node3) and connects them to build a linked list. The display function is used to print the original linked list’s elements.
- The user is then invited to provide a new data value (k) to insert into the linked list. The insertion sort function is then called with the address of the head pointer and a pointer to the new node holding the inserted data. This function ensures that fresh data is added into the linked list while keeping the sorted order.
- Finally, the modified linked list is presented using the display method to demonstrate the result of the insertion. The complete procedure explains how the insertion sort algorithm is used to put elements into a linked list while keeping it sorted.

The input of the code is

Linked list before insertion : 10 15 20

Enter data you want to insert: 18

The output would be like

Linked list after insertion : 10 15 18 20

c) Always insert elements at one end, and delete elements from the other end (first-in first-out QUEUE).

```
Output: 10 20 30 Queue is empty
```