Hydergraph – Mapping Culture Through Food Heritage

Sohan Thummala
2023101019
& $RudraChoudhary$
2024121012
& $SaiyamJain$
2023101135

November 18, 2025

Abstract

Unlike existing latent feature based recommenders that tend to be opaque, network-based recommenders store explicit relationships between nodes which in our case are elements of Hyderabadi culture. This project aims to build a network-based recommender for Hyderabad's culture which is transparent, interpretable and can provide deeper insights into the connections between cultural elements in Hyderabad.

1 Introduction

Recommender Systems (RS) are a type of information filtering system designed to predict and suggest items or content—such as products, movies, music, or articles—that a user might be interested in. These predictions are based on the user's past behavior, preferences, or the behavior of similar users [1]. This project aims to create a recommender system for cultural elements in Hyderabad.

An opaque Neural Recommender System(RS) is limited in the reasoning it can provide for suggestions. We propose Hydergraph, a network based recommender for cultural elements in Hyderabad which proposes suggestions based on network data analysis.

We aim to create a Network based RS for the city of Hyderabad. To this end we detail methods to generate network data and heuristic based recommendations. Fig.1 shows an example pipeline for a simple recommender based on adjacency. The user is directed towards elements in nodes that are direct neighbors of cultural elements that they like based on a simple keyword matching.

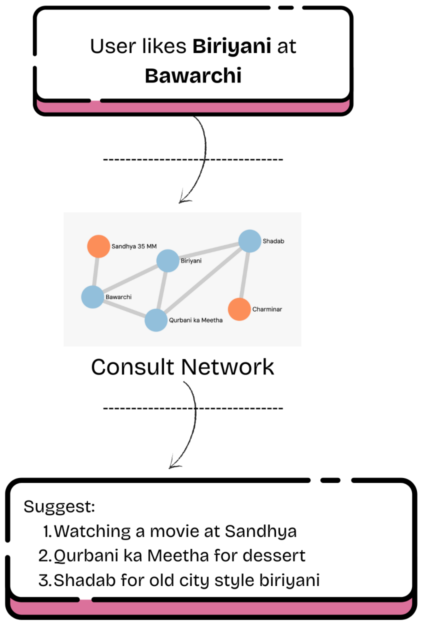

Figure 1: Basic Network based recommender workflow

1.1 Related Work

Our approach includes a network based recommender that does not involve neural network approaches.

In the industry, RSs improve customer satisfaction and drive revenue growth by providing tailored suggestions. Major corporations such as Amazon, Netflix, and Spotify integrate RS into their operations, significantly contributing to their business models. For example, Amazon reports that 35% of its revenue comes from its RS, while Netflix attributes revenues of approximately \$ 33.7 billion and its success in customer retention significantly to its RS [2]. The methods used in these recommender systems are: Content-Based Filtering(CBF), Collaborative Filtering(CF), graph based, reinforcement learning(RL), and mixed approaches.

CBF and CF methods started off with hand-crafted features with small interpretable pipelines to generate recommendations, but the current approaches are based on feature extraction using neural-networks resulting in methods based on feature vectors from these networks.

For a network based approach to building an RS, the choice of network is important, we borrow methodology from Satish et al. [3] where co-occurrence-based method was used to generate network data for citation network analysis which is also our approach for network data generation.

Our method uses the Named Entity Recognition tool from Stanford CoreNLP [4] to recognize entities of interest, elaborated further in section 4. We also use the PageRank algorithm for recommendations.

2 Objectives and Motivation

2.1 Motivation

Hyderabad is rich in cultural elements that interact with each other in interesting ways. These elements can be the subject of common discussion due to their geographical proximity, historical similarity, preferences of the people discussing them, etc. Analyzing these discussions via co-occurrence networks captures many of these interesting characteristics.

2.2 Objectives

1. Collect text data by scraping blogs, social media, news articles, etc.
2. Create network data using co-occurrence methods from text data.
3. Explore centrality based, and similarity based methods to analyze network data.
4. Create a simple recommender augmented by network data

3 Materials, Methods and Results

3.1 Text Data

Text data was collected from the following sources:

1. Wikipedia articles
2. Travel blogs like TripAdvisor, Travel Triangle, etc
3. Food blogs like Taste Atlas, Eating Asia, etc
4. Zomato

3.2 Network construction

Entities are elements of hyderabad culture that form nodes. These were manually identified from the scraped text. Later additions to this can include using entity recognition tools from NLP toolkits like Apache OpenNLP and Stanford CoreNLP. Entity co-occurrence was the method used to obtain connectivity data, entities are said to have an edge between them(undirected) if the occur together. 2 different networks were constructed as follows:

1. Sentence co-occurrence
2. Paragraph co-occurrence

3.3 Recommendation Pipeline

Recommendations are based on remarks by users. Given a user comment, Named Entity Recognition(NER) [4] is applied to identify entities of interest which is the starting point for the recommenders. The chosen recommender takes the entities from the previous step and gives the user suggestions on other entities which are put together as natural language text using an LLM.

We explored recommenders with varying levels of complexity. Below are the recommender algorithms used:

1. **Simple Co-Occurrence Recommender.** For a seed entity e and neighbor v, let w_{ev} denote their co-occurrence weight. The recommendation score is:

   $$\text{score}(v \mid e) = w_{ev}. \tag{1}$$

   Returns top-k neighbors ranked by direct co-occurrence. *Example:* in Fig.2

2. **Personalized PageRank Recommender.** Captures indirect and globally influential entities through Personalized PageRank (PPR). Let $G = (V, E)$ be a directed or undirected graph with adjacency matrix A where A_{ij} is the edge weight from node i to node j. The transition matrix P is the *row-stochastic*(Markov) matrix defined by

   $$P_{ij} = \frac{A_{ij}}{\sum_k A_{ik}}, \tag{2}$$

   With restart probability $\alpha \in (0, 1)$ and seed distribution s, the PPR vector r is given by

   $$r = \alpha s + (1 - \alpha)P^{\top} r. \tag{3}$$

   Then the top K Nodes with the highest PPR values relative to the seed are recommended.

   $$score(v_i|seed) = r_i.$$

   Reasoning: Surfaces influential landmarks beyond immediate neighbors. Votes from influential nodes are weighted more, similar to the PageRank algorithm used by Google [5].

   Restart probability is the probability with which the random walk teleports back to the seed rather than following an outgoing edge. Seed supplied in our case is given by

   $$s = \begin{cases} 1 & \text{Entity found in sentence} \\ 0 & \text{Otherwise} \end{cases} \tag{4}$$

3. **Inverse-Frequency (IDF-Weighted) Recommender.** This method augments the simple co-occurrence score by incorporating a measure of global *rarity* for each entity. The motivation is that certain entities appear very commonly across the dataset (high-degree nodes), making them less informative for recommendation. Conversely, rare or highly specific entities should be prioritized.

   IDF Weight Computation. Let G be the entity graph with N nodes, and let $\deg(v)$ denote the degree of node v, which serves as a proxy for how frequently the entity appears across all contexts. We assign each node a rarity (IDF-like) weight computed as

   $$\mathrm{IDF}(v) = \log\left(\frac{N}{1+\deg(v)}\right). \tag{5}$$

   The term $1 + \deg(v)$ prevents division by zero and ensures numerical stability. Under this definition:

   - high-degree (generic) entities receive a low IDF value,
   - low-degree (rare) entities receive a high IDF value.

   These weights are pre-computed once for all nodes in the graph.

   IDF-Based Recommendation Score. For a given seed entity e, let v be one of its neighbors and let w_{ev} denote their co-occurrence edge weight. The recommender assigns the final score

   $$\mathrm{score}(v \mid e) = w_{ev} \cdot \mathrm{IDF}(v), \tag{6}$$

   thereby promoting neighbors that are both strongly connected to the seed and globally rare.

   Operationally, the algorithm iterates over all neighbors of e, retrieves the edge weight w_{ev}, looks up the pre-computed $\mathrm{IDF}(v)$, multiplies the two to form the final score, and returns the top-k highest-scoring neighbors. This produces recommendations that are distinctive and meaningful, rather than dominated by overly common or generic entities.

Path-Based Recommendation Methods.

Unlike the previous recommenders—which assign scores to individual entities—these methods produce *paths* in the graph. A path recommender outputs a sequence of nodes

$$(u_0, u_1, u_2, \ldots, u_T),$$

starting from a chosen entity.

1. **Weighted Random Walk.** This algorithm generates a path by walking through the graph with transition probabilities proportional to edge weights:

   $$\mathbb{P}(u \to v) \propto w_{uv}.$$

   It returns a sequence, not a score vector. The walk tends to follow stronger co-occurrence edges, producing a locally coherent trail from the start node.

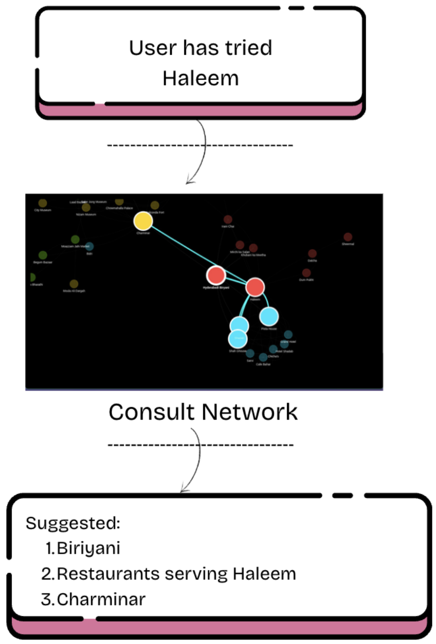

Figure 2: Recommender flow for simple top K Neighbors based recommendation

2. **Guided Start-to-End Path.** This method constructs a path from a specified start entity to a specified end entity. At each step it scores neighbors by combining:

   $$\text{edge strength } w_{un} \quad \text{and} \quad \text{progress toward the end node } d(n),$$

   via the function

   $$\text{score}(n) = w_{un} \cdot \frac{1}{1 + d(n)}.$$

   The resulting path is goal-directed but still stochastic weighted by the score.

3. **Exploratory Walk with Teleportation.** This produces an exploratory path rather than a strictly local walk. At each step:

   $$\text{with probability } \tau: \text{ teleport to a random node;} \qquad \text{else: take a weighted step.}$$

   The teleportation allows the path to jump to new regions of the graph, generating diverse trajectories. *Example:* Fig. 3

4. **Guided Exploratory Start-to-End Walk.** This combines goal-direction with exploration and is a combination of the previous two methods. With probability τ the walker teleports; otherwise it chooses a neighbor using the guided score

   $$\text{score}(n) = w_{un} \cdot \frac{1}{1 + d(n)}.$$

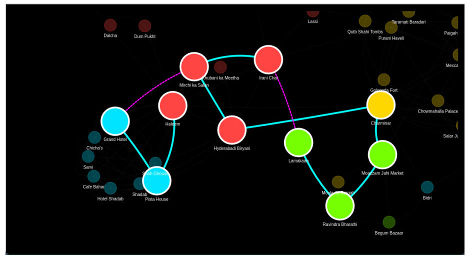

Figure 3: Exploratory Walk with teleportation starting at Haleem

3.4 Negative results

While the algorithm works smoothly for expected cases, we observed some negative results. NER is case sensitive, and treats some entities strangely. For example if a user remarks "I want to drink Irani chai and eat some Osmania biscuits", 'Irani' and 'Osmania' are recognised as entities while we expect "Irani chai" and Osmania biscuits" as entities.

The random walk methods do not always give feasible itineraries because there is no geo-spatial data integrated into the network.

4 Conclusion

The sentence co-occurrence network provides a network that has desirable properties like linking places that are close together, food that is available at the same restaurants, restaurants that have similar cuisines, etc. Some data is also based on bloggers' personal recommendations/preferences. These properties make the network useful for a recommender system.

The recommender system is simple but powerful, requiring little compute.

References

[1] Charu C. Aggarwal. Recommender Systems The Textbook. Number 4. Springer Cham, 2016.

[2] Shaina R. and Mizanur R. and Safiullah K. and Armin T. and Ananya R. and Farshad N. and Amirmohammad K. (2024) A Comprehensive Review of Recommender Systems: Transitioning from Theory to Practice. arXiv:2407.13699

[3] Satish Muppidi and K. Thammi Reddy. 2020. Co-occurrence analysis of scientific documents in citation networks. Int. J. Know.-Based Intell. Eng. Syst. 24, 1 (2020), 19–25. https://doi.org/10.3233/KES-200025

[4] Manning, Christopher D., Mihai Surdeanu, John Bauer, Jenny Finkel, Steven J. Bethard, and David McClosky. 2014. The Stanford CoreNLP Natural Language Processing Toolkit In Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations, pp. 55-60.

[5] Sergey Brin and Lawrence Page, The Anatomy of a Large-Scale Hypertextual Web Search Engine, Computer Networks 1998 link