Member-only story

GraphRAG + GPT-4o-Mini is the RAG Heaven

Part 1: Introduction to GraphRAG

Vatsal Saglani · Follow

Published in Towards AI

10 min read · 3 days ago

Listen Share More

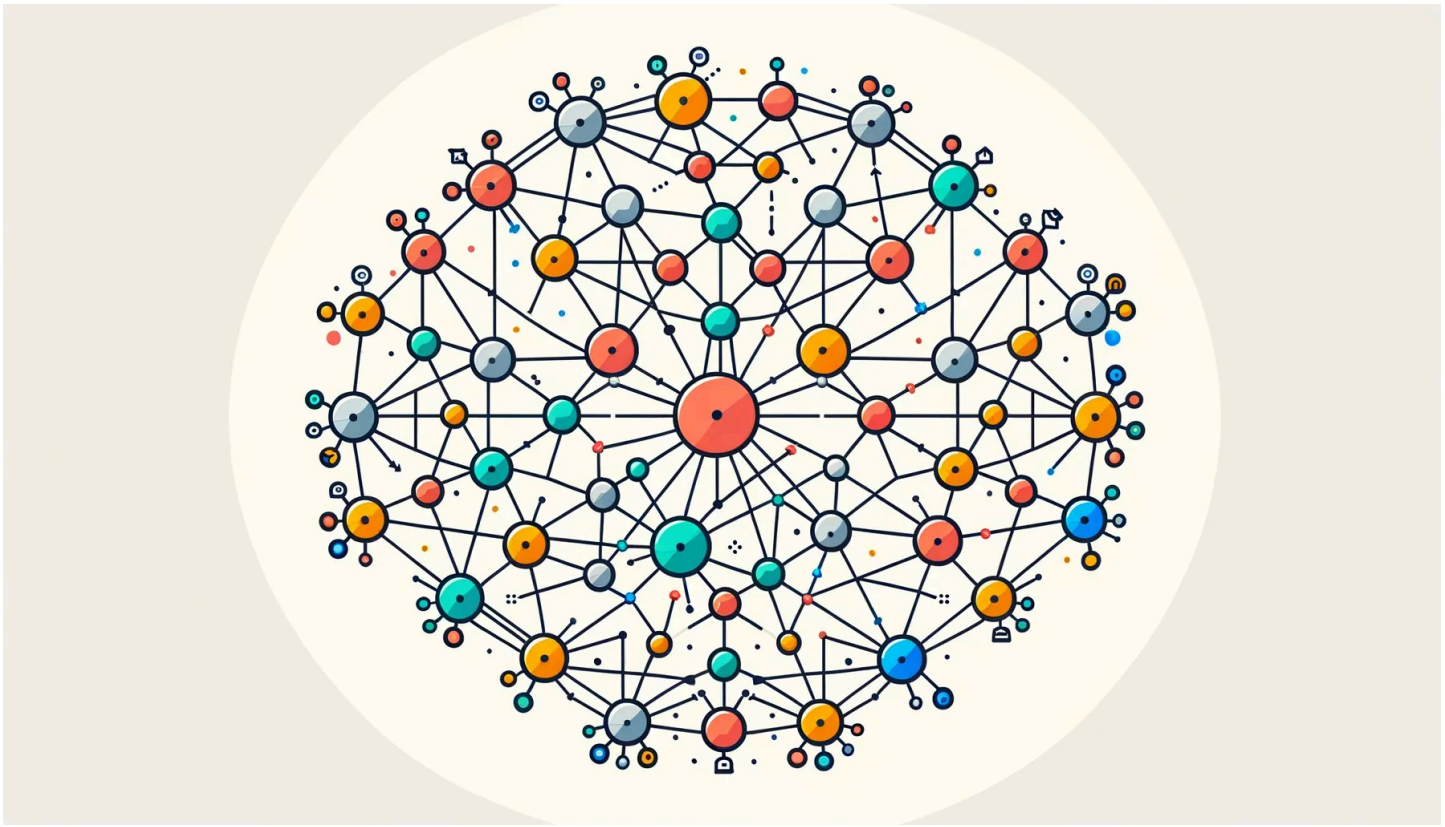


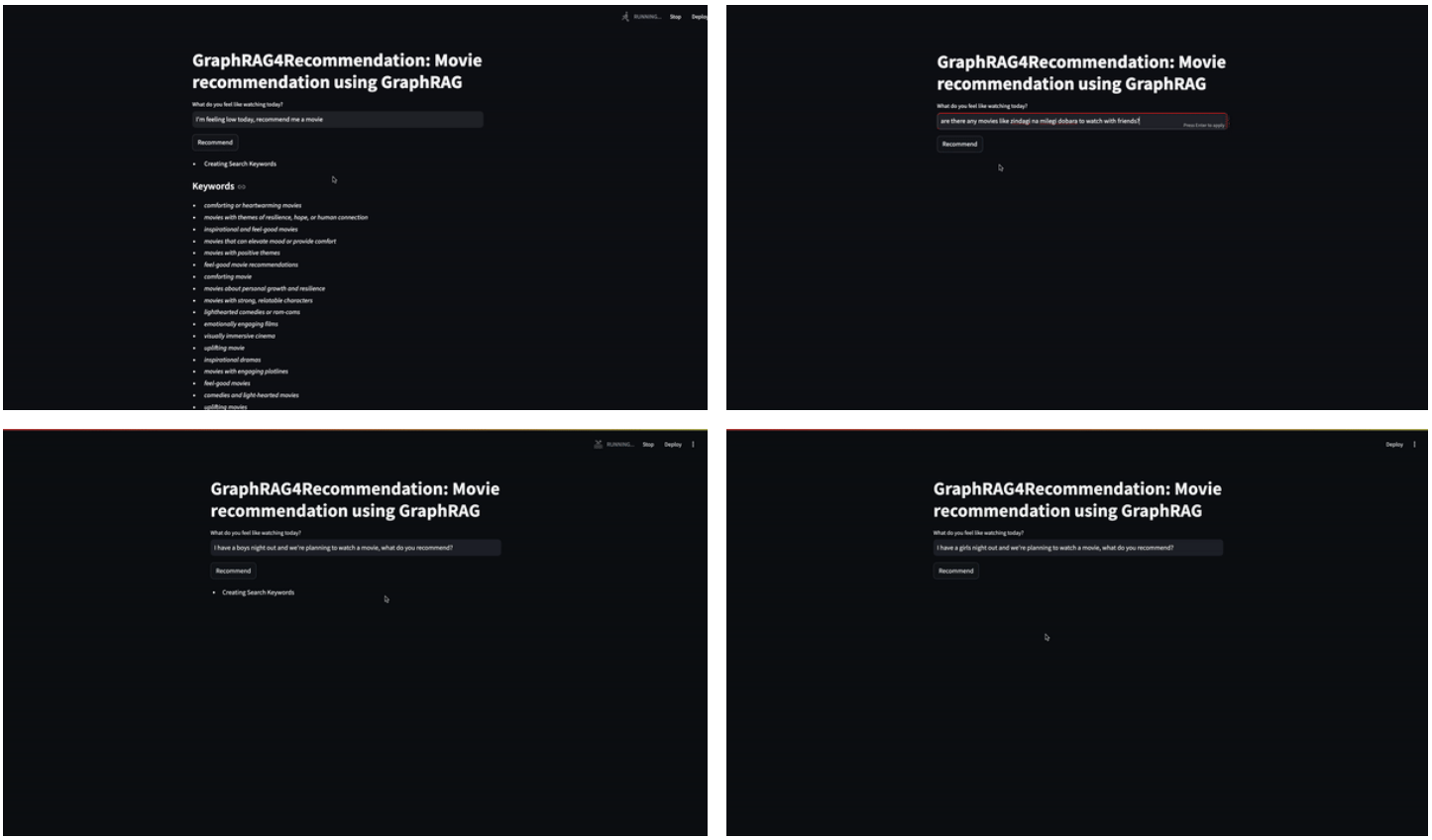
Image by DALL-E 3

> **Disclaimer:** *This implementation of GraphRAG is inspired by the paper From Local to Global: A Graph RAG Approach to Query-Focused Summarization by Darren Edge et. al. The code is not entirely similar to the paper's codebase, though the prompts for certain tasks are taken from the paper's codebase.*

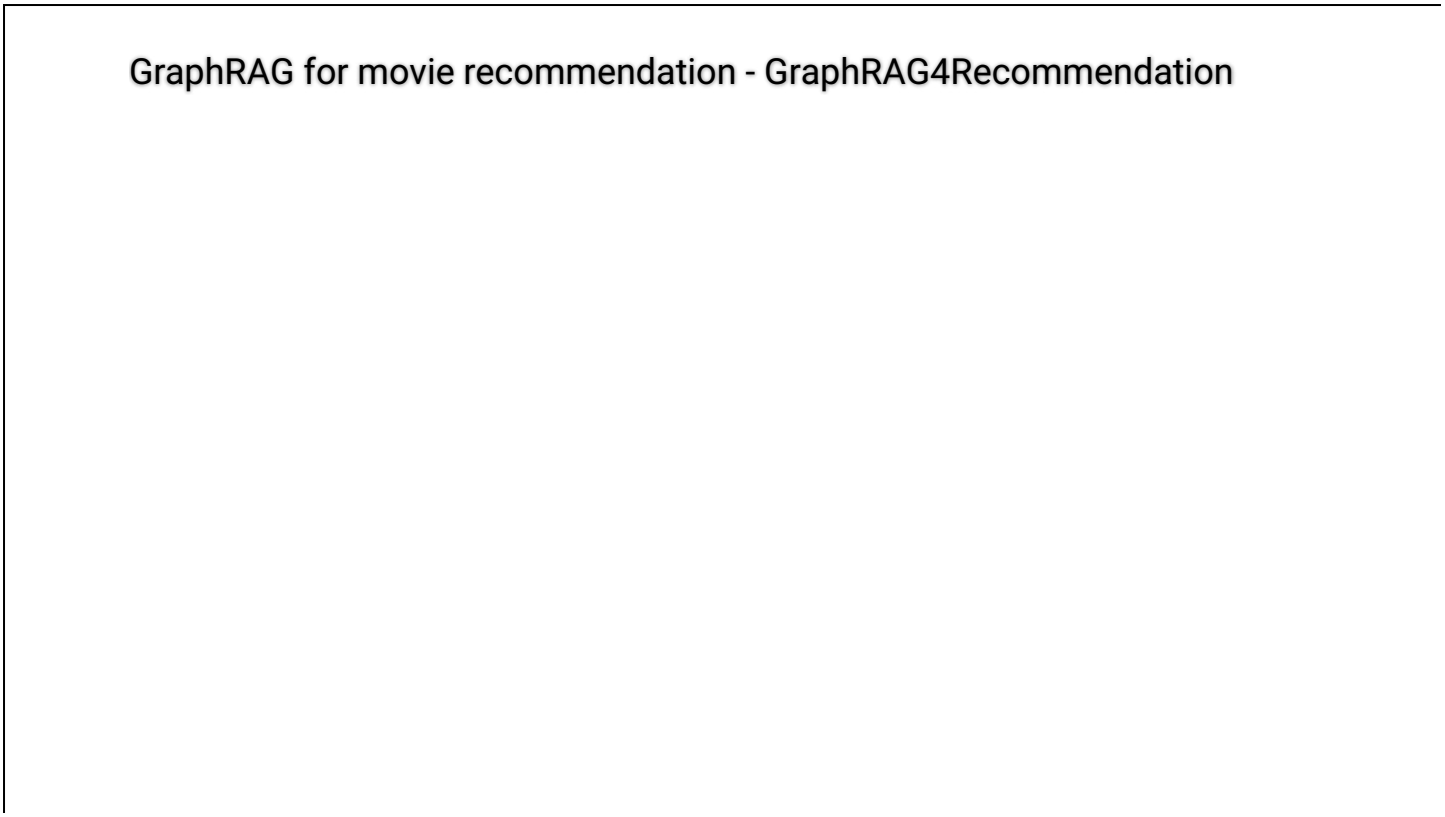
This is the ***first blog*** in a multi-part blog series series about GraphRAG. In this blog series, our goal is to achieve the following,

1. **Understand the fundamentals of GraphRAG**
2. The need for GraphRAG: GraphRAG vs. Semantic/Keyword-based RAG
3. Implement GraphRAG components from scratch in Python
4. Apply GraphRAG for Content-Based Movie Recommendation: “GraphRAG4Reccomendation”
5. Use GPT-4o-Mini for creating the graph and providing recommendations

We will achieve the following output by the end of this multi-part blog series.



Implementation Output by Author



Video by Author

The following is the GitHub repository for the GraphRAG4Rec codebase.

GitHub - vatsalsaglani/GraphRAG4Rec: A naive implementation of GraphRAG for Movie Recommendation on...

A naive implementation of GraphRAG for Movie Recommendation on IMDB Top 1000 movies dataset. ...

github.com

Other Parts

- Part 2: GraphRAG vs Semantic/keyword-based RAG
- Part 3: Extract — entities, relations, and claims to build the graph *(coming soon)*
- Part 4: Batch communities and prepare summarization reports *(coming soon)*
- Part 5: Query processing and recommendation generation via map-reduce prompting *(coming soon)*

In this blog,

We'll understand the fundamentals of GraphRAG with an example.

What is GraphRAG?

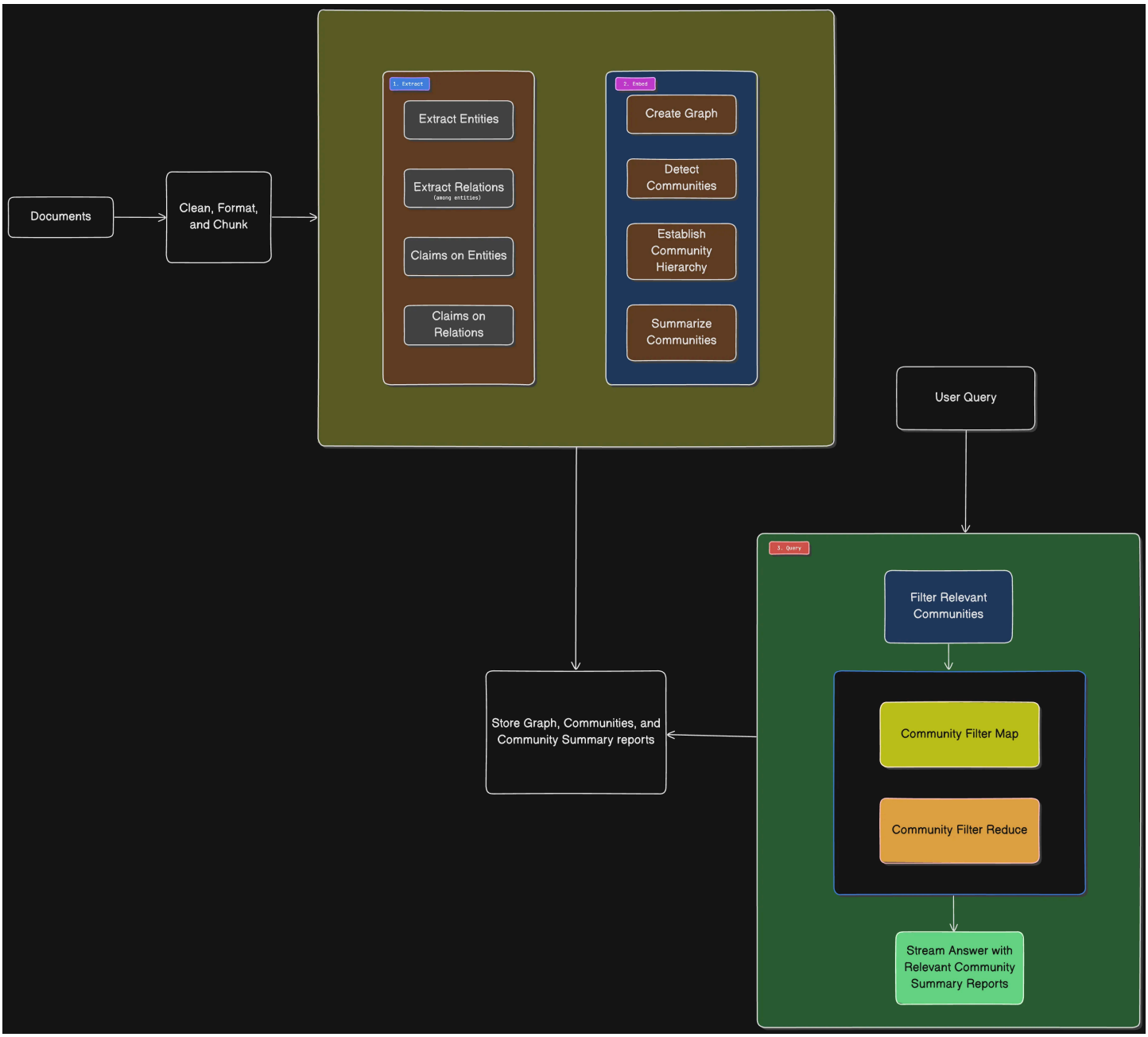
GraphRAG is an advanced Graph-based Retrieval Augmented Generation (GraphRAG) approach introduced in the paper ***From Local to Global: A Graph RAG Approach to Query-Focused Summarization*** by Darren Edge et. al. This approach combines graph theory, information retrieval, and LLMs.

The core concept is that the entities in our text are represented as nodes in the graphs and the relations between these entities represent the edges between the nodes. The graph is then hierarchically divided into communities and summarized into community reports.

At query time, we've to decide how deep should we explore the communities to find relevant communities. The more the depth the more the computations/LLM calls.

Once the relevant communities are retrieved we can answer the user query using the summary reports of those communities.

The following diagram depicts the entire process.



GraphRAG Diagram by Author

Key components of GraphRAG

As shown in the above image, we've divided the GraphRAG process into the following three key components.

1. Extract
2. Embed
3. Query

Let's understand these components individually.

Extract

In the extract component, we do the following things,

1. Extract Entities
2. Extract Relations among Entities
3. Extract Claims on Entities
4. Extract Claims on Relations

We'll understand this with an example.

Suppose we have the following text.

> *Movie: 'The Shawshank Redemption' Two imprisoned men bond over a number of years, finding solace and eventual redemption through acts of common decency.\nYear: 1994\nDirector: Frank Darabont\nCast: ['Tim Robbins', 'Morgan Freeman', 'Bob Gunton', 'William Sadler']\nCertificate: A*

Step 1: Extract Entities

- The Shawshank Redemption (Movie)
- Frank Darabont (Person — Director)
- Tim Robbins (Person — Actor)
- Morgan Freeman (Person — Actor)
- Bob Gunton (Person — Actor)
- William Sadler (Person — Actor)
- 1994 (Year)
- A (Certificate)

Step 2: Extract Relations

- The Shawshank Redemption — *directed by* — Frank Darabont
- The Shawshank Redemption — *stars* — Tim Robbins
- The Shawshank Redemption — *stars* — Morgan Freeman
- The Shawshank Redemption — *stars* — Bob Gunton

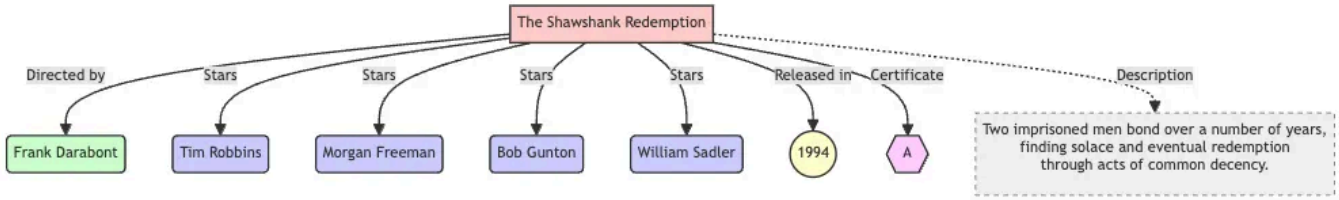
- The Shawshank Redemption — *stars* — William Sadler
- The Shawshank Redemption — *released in* — 1994
- The Shawshank Redemption — *has certificate* — A

Steps 3–4: Extract Claims for Entities and Relations

- The Shawshank Redemption: “Two imprisoned men bond over a number of years, finding solace and eventual redemption through acts of common decency.”
- The Shawshank Redemption: Released in 1994
- The Shawshank Redemption: Has certificate A

The central node will be the name of the movie — *The Shawshank Redemption.*

If we were to plot the entities, relations, and claims it would look something like the image below.



Extract: Nodes, Edges, and Claims (By Author)

Embed

After processing the required steps for the first component — Extract — for all the documents, the extracted information will be embedded into a graph.

As we need to embed movies into a graph let's take two more movie texts.

> *"Movie: 'Inception'\nGenre: ['Sci-Fi', 'Action', 'Thriller']\nYear: 2010\nDirector: Christopher Nolan\nCast: ['Leonardo DiCaprio', 'Joseph Gordon-Levitt']\nCertificate: PG-13"*
>
> *"Movie: 'The Matrix'\nGenre: ['Sci-Fi', 'Action']\nYear: 1999\nDirector: The Wachowskis\nCast: ['Keanu Reeves', 'Laurence Fishburne']\nCertificate: R"*

Extract output for Inception

Step 1: Extract Entities

- Inception (Movie)
- Christopher Nolan (Person — Director)
- Leonardo DiCaprio (Person — Actor)
- Joseph Gordon-Levitt (Person — Actor)
- 2010 (Year)
- PG-13 (Certificate)
- Sci-Fi (Genre)
- Action (Genre)
- Thriller (Genre)

Step 2: Extract Relations

- Inception — *directed by* — Christopher Nolan
- Inception — *stars* — Leonardo DiCaprio
- Inception — *stars* — Joseph Gordon-Levitt
- Inception — *released in* — 2010
- Inception — *has certificate* — PG-13
- Inception — *has genre* — Sci-Fi
- Inception — *has genre* — Action
- Inception — *has genre* — Thriller

Steps 3–4: Extract claims on Entities and Relations

- Inception: “A skilled thief with the rare ability to ‘extract’ information from people’s minds is offered a chance to regain his old life as payment for a task considered to be impossible: ‘inception’, the implantation of another person’s idea into a target’s subconscious.”
- Inception: Released in 2010
- Inception: Has certificate PG-13

- Inception: Is a Sci-Fi film
- Inception: Is an Action film
- Inception: Is a Thriller film

Extract output for The Matrix

Step 1: Extract Entities

- The Matrix (Movie)
- The Wachowskis (Person — Directors)
- Keanu Reeves (Person — Actor)
- Laurence Fishburne (Person — Actor)
- 1999 (Year)
- R (Certificate)
- Sci-Fi (Genre)
- Action (Genre)

Step 2: Extract Relations

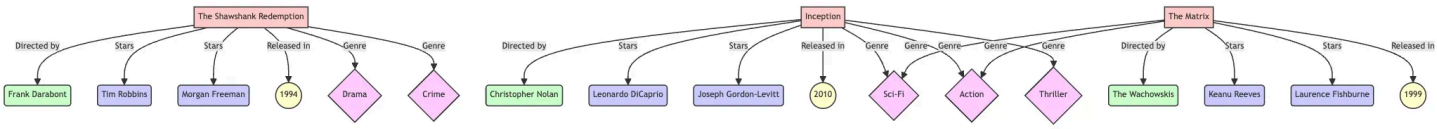
- The Matrix — *directed by* — The Wachowskis
- The Matrix — *stars* — Keanu Reeves
- The Matrix — *stars* — Laurence Fishburne
- The Matrix — *released in* — 1999
- The Matrix — *has certificate* — R
- The Matrix — *has genre* — Sci-Fi
- The Matrix — *has genre* — Action

Steps 3–4: Extract claims on Entities and Relations

- The Matrix: “A computer programmer discovers that reality as he knows it is a simulation created by machines to subjugate humanity, and joins a rebellion to overthrow them.”
- The Matrix: Released in 1999
- The Matrix: Has certificate R
- The Matrix: Is a Sci-Fi film
- The Matrix: Is an Action film

Embed Step 1: Create a Graph

Now that we have the *entities, relations, and claims* from all three movies we can ***embed*** these into a graph like the following.

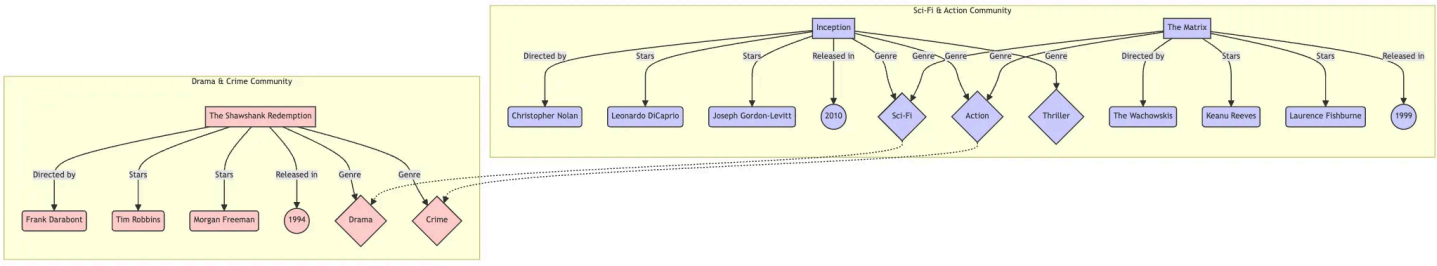


Embedding entities, relations, and claims (By Author)

Embed Step 2–3: Detect Communities and Establish Hierarchy

We can divide the graph into the following two communities based on the genres.

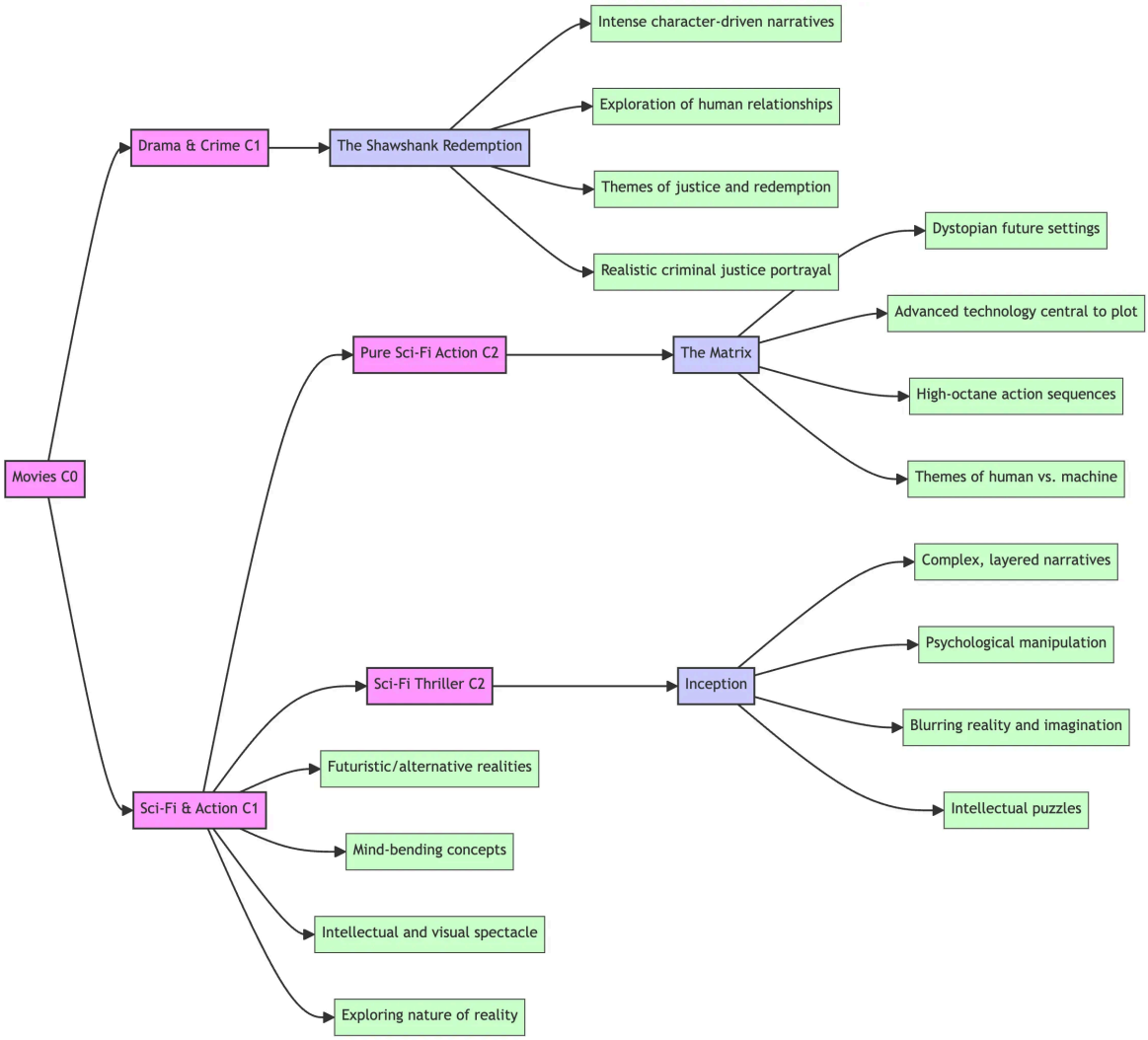
1. Drama and Crime Community
2. Sci-Fi and Action Community



Community based on Genre (by Author)

We can use a hierarchical community detection algorithm — the Leiden Algorithm — to cluster the nodes into separate communities.

First, let's look at how the hierarchical communities will come out.



Hierarchical communities (by author)

We have the following hierarchies.

1. C0 — Movies: This community contains all the movies in our dataset. It represents a diverse range of movies spanning across different genres, periods, and themes. The movies share common elements such as directors, actors, genres, and release year, but differ in their specific content and style.

2. C1 — Drama and Crime: This community focuses on dramatic storytelling with elements of crime.

3. C1 — Sci-Fi and Action: This community combines elements of science fiction and action.

4. C2 — Pure Sci-Fiction: This sub-community, represented by “The Matrix,” focuses on science fiction concepts with a heavy emphasis on action.

5. C2 — Sci-Fi Thriller: This sub-community, represented by “Inception,” combines science fiction elements with psychological thriller aspects.

With this hierarchy, we have both global and local categorization. The C0 and C1 clusters/groups/communities are very broad — global — and the C2 cluster/group/community are very specific — local.

Embed Step 4: Summarize Communities

1. C1 — Drama and Crime

- Intense character-driven narratives
- Exploration of human relationships and emotions
- Themes of justice, redemption, and perseverance
- Realistic portrayals of criminal justice systems

2. C1 — Sci-Fi and Action

- Futuristic or alternative reality settings
- Mind-bending concepts and technologies
- Blend of intellectual stimulation and visual spectacle
- Exploration of the nature of reality and consciousness

3. C2 — Pure Sci-Fi Action

- Dystopian future settings
- Advanced technology as a central plot element
- High-octane action sequences
- Themes of human vs. machine

4. C2 — Sci-Fi Thriller

- Complex, layered narratives
- Psychological manipulation and exploration
- Blurring lines between reality and imagination
- Intellectual puzzles and mind-bending concepts

> *Summary reports can also contain awards, performance by actor directory, box office results, etc. as well.*

We'll take a short detour and understand the Leiden algorithm with the movie example.

About Leiden Algorithm

> *The Leiden algorithm is an improved version of the Louvain method for community detection. It works by optimizing modularity — a measure of the density of links inside communities compared to links between communities.*

First, let's understand *modularity.*

Modularity is a measure of how well a network is divided into communities. We can think of it as,

- High modularity means there are many connections within communities and few connections between different communities.
- Low modularity means connections are more evenly spread, with no clear community structure.

For our movie example, high modularity would mean movies within a community have many shared characteristics like the Sci-Fi and Action community. Low modularity means fewer characteristics in common like the Drama and Crime community.

Hierarchical Community Detection Steps

Let's look at the community detection steps for the movie example.

Step 1: Start with individual nodes

Begin with each movie as its own community.

- Community 1: The Shawshank Redemption
- Community 2: Inception
- Community 3: The Matrix

Step 1 (by author)

Step 2: Merge nodes into communities

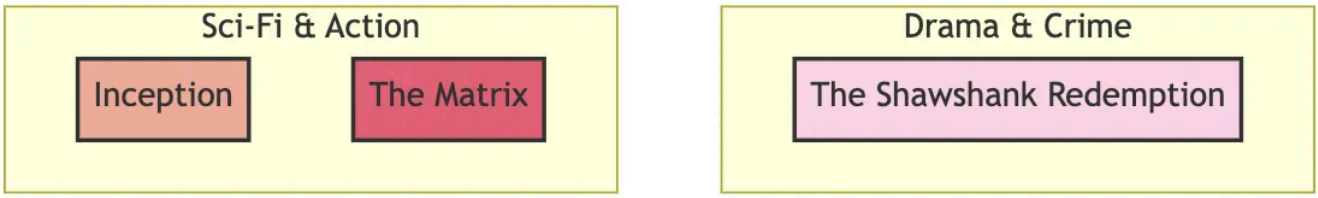
Look at the connections between movies like shared genres or themes and merge them if it improves modularity.

- Merge Inception and The Matrix into a Sci-Fi and Action community.
- The Shawshank Redemption remains in its own Drama and Crime community.

Step 2 (by author)

Step 3: Create the first level of hierarchy (C1):

- C1 Community 1: Drama & Crime (The Shawshank Redemption)
- C1 Community 2: Sci-Fi & Action (Inception, The Matrix)



Step 3 (by author)

Step 4: Communities as Nodes

Now consider communities as nodes.

Step 4 (by author)

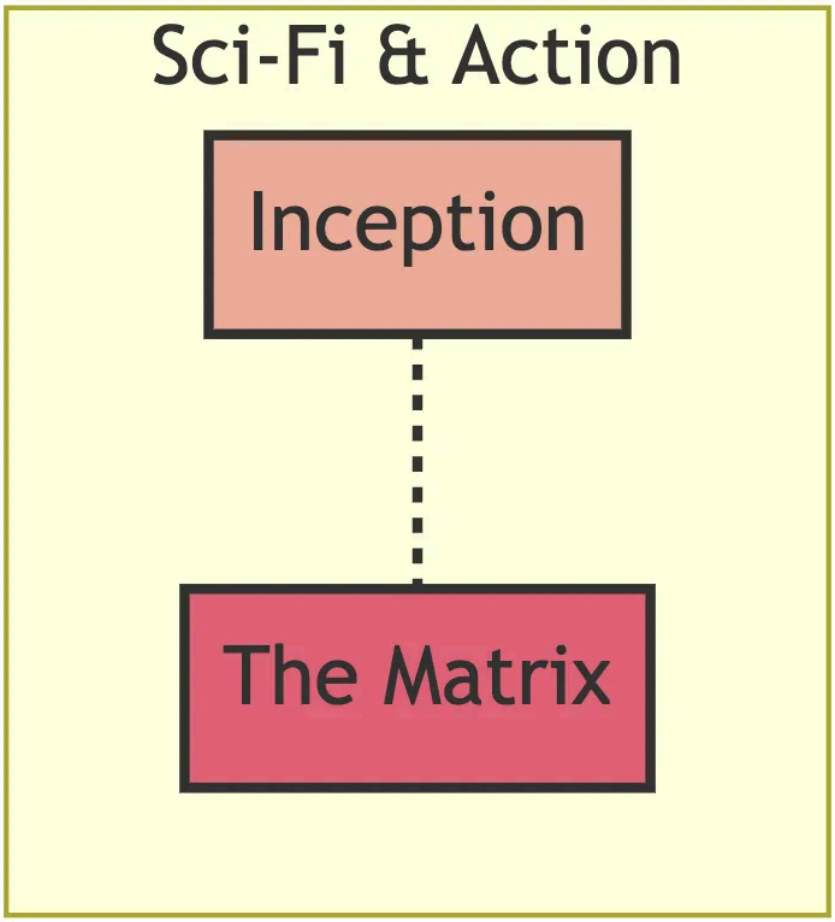
Step 5: Repeat Steps 1, 2, and 3 at a higher level

Look for connections between these community nodes. In this case, there aren't enough communities to merge further, so we shop here for the C0 level.

Step 6: Refine lower levels

Go back to the Sci-Fi and Action community and look for subcommunities.

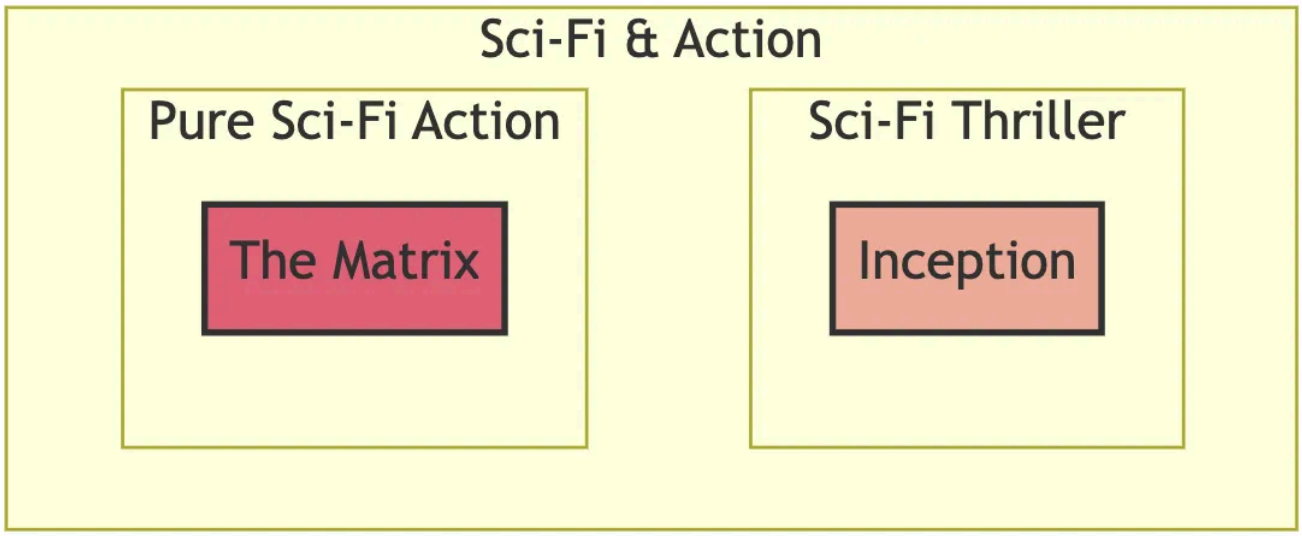
- Split Inception and The Matrix based on their more specific characteristics.



Step 6 (by author)

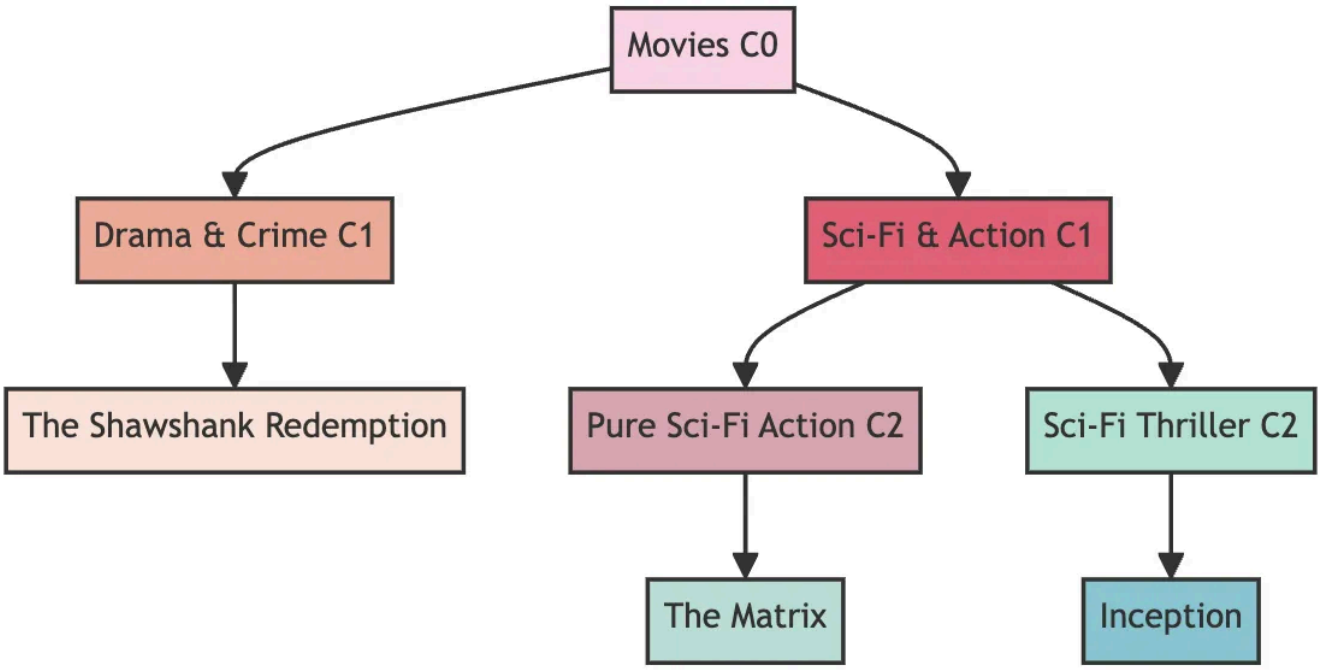
Step 7: Create the second level of hierarchy (C2)

- C2 Community 1: Pure Sci-Fi Action (The Matrix)
- C2 Community 2: Sci-Fi Thriller (Inception)



Step 7 (by author)

Finally, we have the following hierarchy.



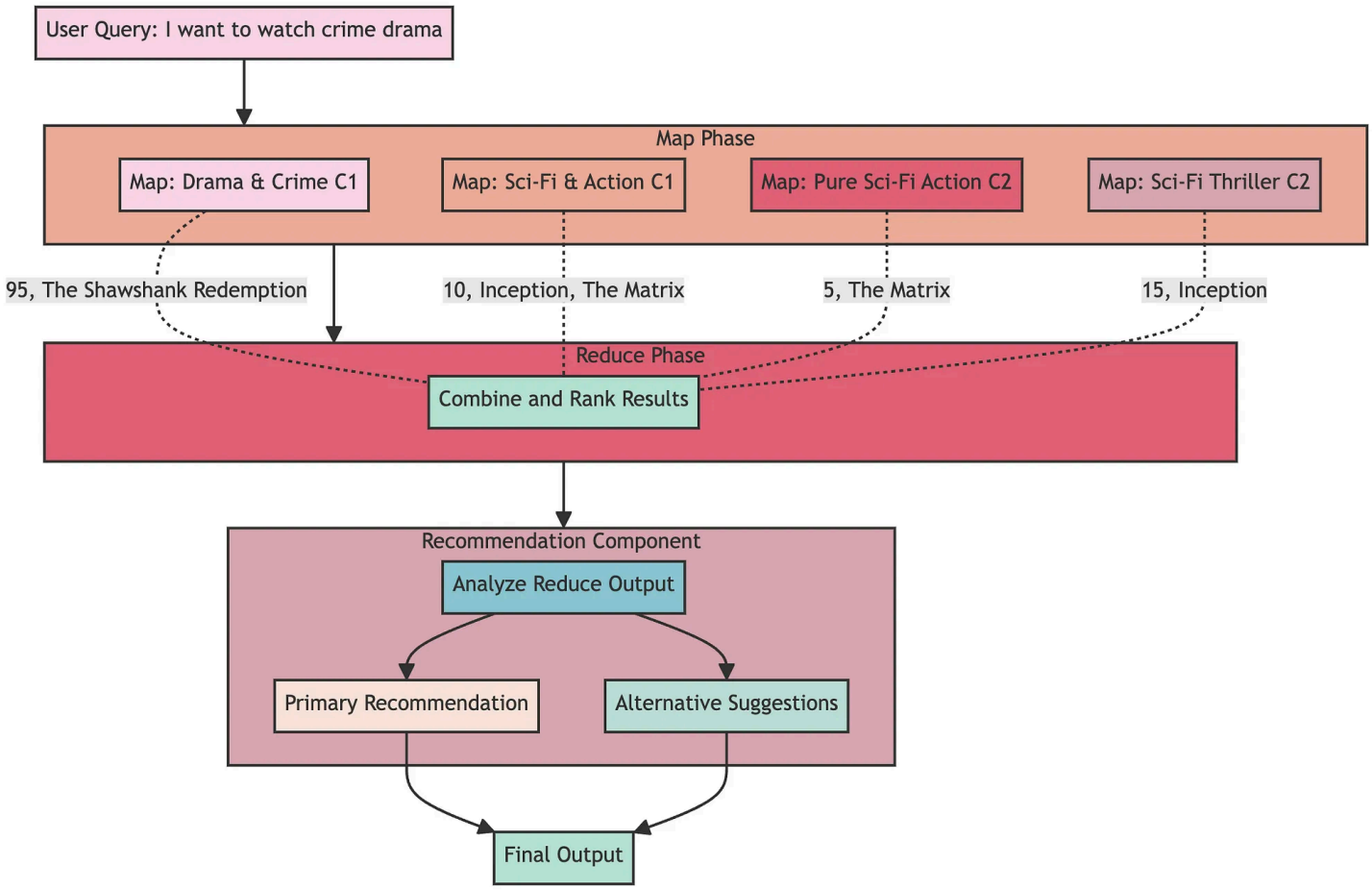
Final community hierarchy (by author)

Query

In the *query* part, we use a map-reduce approach to find relevant communities using a map operation. The map outputs are then provided to the reduce (reducer) to answer the user query.

Let's look at the query process with an example query — *I want to watch a crime drama.*

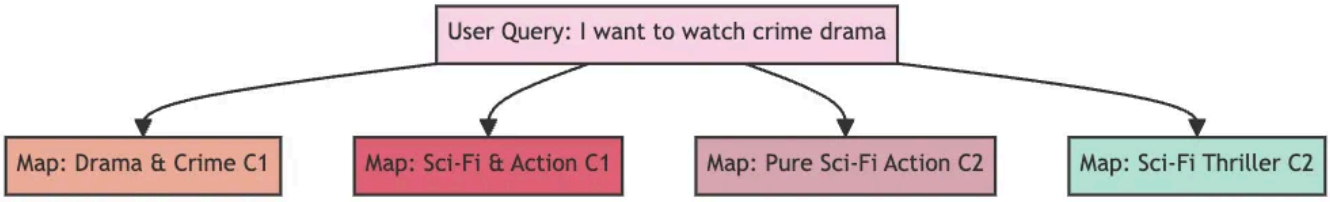
The following is how the entire process will look like.



Query Step (by author)

Map Phase

We first have the map phase. Here, every community report is passed to the mapper which will output how relevant the community is for the given query along with the movies.



Query Map Phase (by author)

The output of the map phase through every community will look like the following.

- Drama and Crime C1:

```
{
    "community": "Drama & Crime C1",
    "relevance_score": 95,
    "movies": ["The Shawshank Redemption"],
    "reason": "Directly matches the crime drama genre request"
}
```

- Sci-Fi and Action C1

```
{
    "community": "Sci-Fi & Action C1",
    "relevance_score": 10,
    "movies": ["Inception", "The Matrix"],
    "reason": "Does not match the crime drama genre request"
}
```

- Pure Sci-Fi Action C2

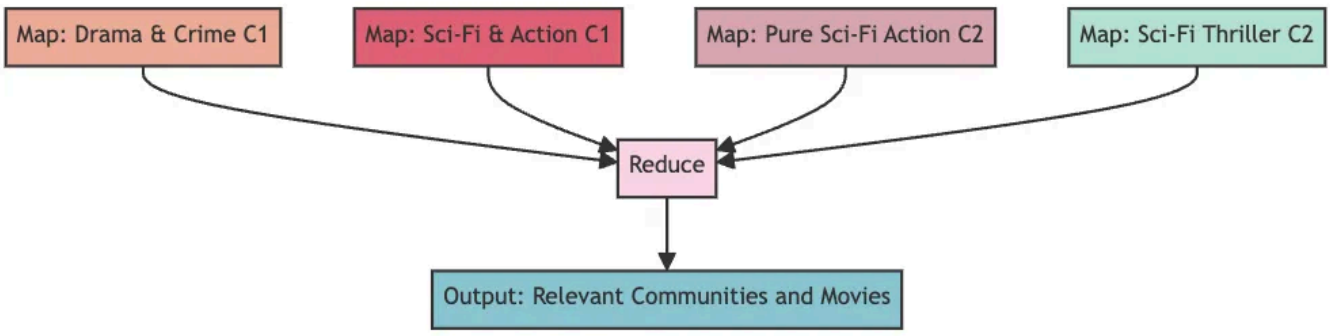
```
{
    "community": "Pure Sci-Fi Action C2",
    "relevance_score": 5,
    "movies": ["The Matrix"],
    "reason": "Does not match the crime drama genre request"
}
```

- Sci-Fi Thriller C2

```
{
    "community": "Sci-Fi Thriller C2",
    "relevance_score": 15,
    "movies": ["Inception"],
    "reason": "Has some thriller elements but not crime drama"
}
```

Reduce Phase

The outputs from the map phase are passed to the reducer along with the user query to get a list of relevant suggestions along with other recommendations.



Query Reduce Phase (by author)

The following is how the output of the reduce phase will look like.

```
{
    "relevant_communities": [
        {
            "community": "Drama & Crime C1",
            "relevance_score": 95,
            "movies": ["The Shawshank Redemption"]
        }
    ],
    "other_suggestions": [
        {
            "community": "Sci-Fi Thriller C2",
            "relevance_score": 15,
            "movies": ["Inception"]
        }
    ]
}
```

Moreover, we can communicate this output via an LLM by providing the user query and the relevant communities with movie details along with suggestions. We can prompt the LLM to personalize the output based on the user query and the relevant movies and extra suggestions.

Conclusion

In this blog, we got an introduction to GrapRAG and the key components — extract, embed, and query. Along with that we also learnt about hierarchical community detection using the Leiden algorithm. In the upcoming blogs, we'll build upon this knowledge to develop a GraphRAG module for a content-based movie recommendation system — GraphRAG4Recommendation.

See you in Part 2: GraphRAG vs semantic/keyword-based RAG.

Software Development | AI | Technology | Generative Ai Tools | Llm

Follow

Written by Vatsal Saglani

2K Followers · Writer for Towards AI

Data Science Lead - GenAI. A Software Engineer, Programmer & Deep Learning professional.
https://vatsalsaglani.vercel.app

More from Vatsal Saglani and Towards AI

Vatsal Saglani in Towards AI

Llama 3 + Groq is the AI Heaven

Llama 3 shines on Groq with blazing generation

Apr 21 1.4K 10

Milan Tamang in Towards AI

Build your own Large Language Model (LLM) From Scratch Using PyTorch

A Step-by-Step guide to build and train an LLM named MalayGPT. This model's task is to translate texts from English to Malay language.

Jun 5 1.1K 10

Gao Dalie (高達烈) in Towards AI

Langchain + Graph RAG + GPT-4o Python Project: Easy AI/Chat for your Website

This is Graph and I have a super quick tutorial showing how to create a fully local chatbot with Langchain, Graph RAG and GPT-4o to make a...

Jul 7 964 2

Vatsal Saglani in Towards AI

Llama 3 + Llama.cpp is the local AI Heaven

Build a fully local (nano) DiagramGPT using Llama 3 8B and learn about inline function calling.

May 13 381 1

See all from Vatsal Saglani

See all from Towards AI

Recommended from Medium

Jim Clyde Monge in Generative AI

Meet Claude Dev—An Open-Source Autonomous AI Programmer In VS Code

Claude Dev is an autonomous software engineer right in your IDE. Open source and available on VSCode marketplace now.

6d ago 410 6

Karthik Rajan in AI Advances

Microsoft's GraphRAG + AutoGen + Ollama + Chainlit = Fully Local & Free Multi-Agent RAG Superbot

This superbot app integrates GraphRAG with AutoGen agents, powered by local LLMs from Ollama, for free & offline embedding & inference.

Jul 15 339 1

Lists

Generative AI Recommended Reading
52 stories · 1227 saves

The New Chatbots: ChatGPT, Bard, and Beyond
12 stories · 427 saves

AI Regulation
6 stories · 516 saves

What is ChatGPT?
9 stories · 396 saves

Ali Waseem

RouteLLM: Achieves 90% GPT-4 Quality at 80% Lower Cost

Open-Source Solution for Efficient LLM Routing

Jul 9 167 2

Jan Kammerath

Inside The Outages: A Dangerous Null Pointer Exception Deployed On Friday

The world went into shock when cyber security firm “Crowdstrike”, a provider of endpoint protection software, released an update on Friday...

5d ago 3.6K 92

Andrew Zuo

GPT-3.5 Turbo FINALLY Has A Successor

And It's Crazy Good

6d ago 60 3

See more recommendations