

GETTING STARTED

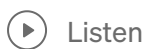
Community Detection Algorithms



Thamindu Dilshan Jayawickrama · Follow

Published in Towards Data Science

7 min read · Jan 29, 2021



Listen



Share

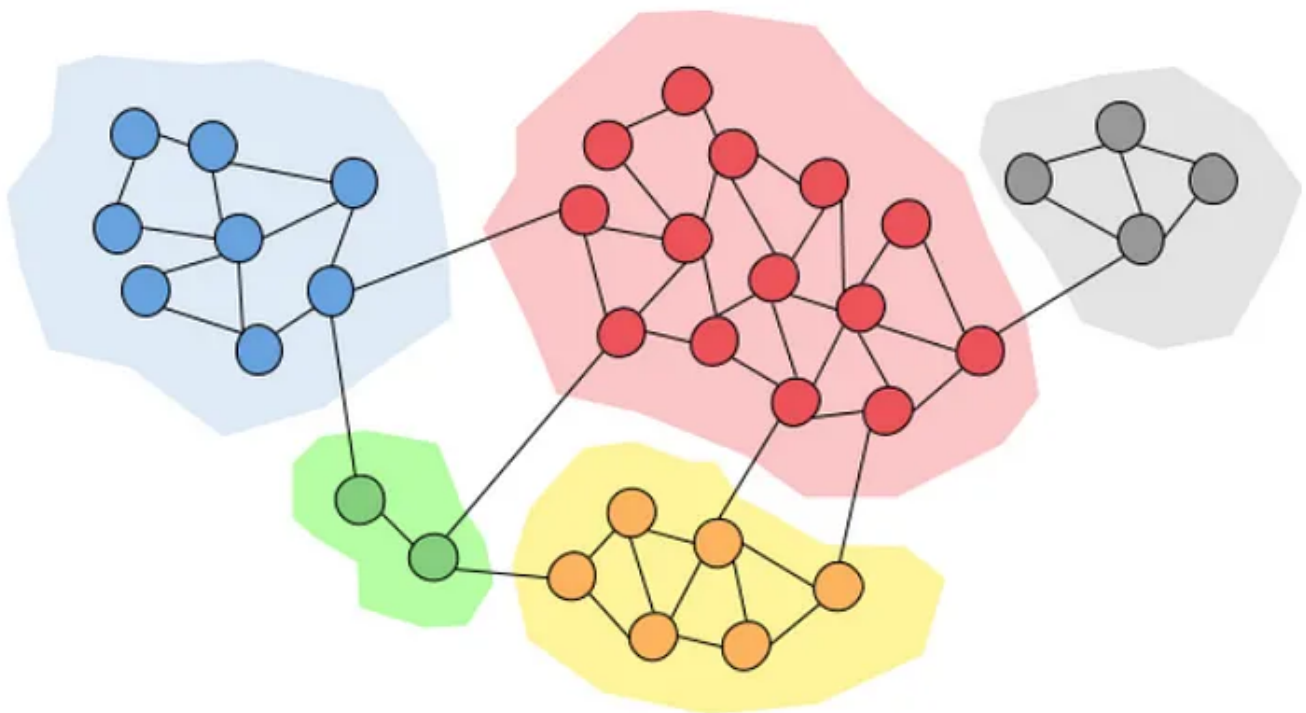


Image by Author

Many of you are familiar with networks, right? You might be using social media sites such as Facebook, Instagram, Twitter, etc. They are social networks. You might be dealing with stock exchanges. Either you might be buying new stocks, selling what you already have, etc. They are networks. Not only in the technological field but also in our day to day social life, we deal with many networks. **Communities are a property of many networks in which a particular network may have multiple communities such that nodes inside a community are densely connected.** Nodes in multiple communities can overlap. Think of your Facebook or Instagram account

and consider who you interact with daily. You might be heavily interacting with your friends, colleagues, family members and a few other important people in your life. They form a very dense community inside your social network.

M. Girvan and M. E. J. Newman are two popular researchers in the domain of community detection. In one of their research, they have highlighted the **community structure-property** using social networks and biological networks. According to them, **network nodes are tightly connected in knit groups within communities and loosely connected between communities.**

Why Community Detection?

When analyzing different networks, it may be important to discover communities inside them. Community detection techniques are useful for social media algorithms to discover people with common interests and keep them tightly connected. Community detection can be used in machine learning to detect groups with similar properties and extract groups for various reasons. For example, this technique can be used to discover manipulative groups inside a social network or a stock market.

Community Detection vs Clustering

One can argue that community detection is similar to clustering. Clustering is a machine learning technique in which similar data points are grouped into the same cluster based on their attributes. Even though clustering can be applied to networks, it is a broader field in unsupervised machine learning which deals with multiple attribute types. On the other hand, community detection is **specially tailored for network analysis** which depends on a **single attribute type called edges**. Also, clustering algorithms have a tendency to separate single peripheral nodes from the communities it should belong to. However, both clustering and community detection techniques can be applied to many network analysis problems and may raise different pros and cons depending on the domain.

Community Detection Techniques

Community detection methods can be broadly categorized into two types; **Agglomerative Methods** and **Divisive Methods**. In Agglomerative methods, edges are added one by one to a graph which only contains nodes. Edges are added from the stronger edge to the weaker edge. Divisive methods follow the opposite of agglomerative methods. In there, edges are removed one by one from a complete graph.

There can be any number of communities in a given network and they can be of varying sizes. These characteristics make the detection procedure of communities very hard. However, there are many different techniques proposed in the domain of community detection. Four popular community detection algorithms are explained below. All of these listed algorithms can be found in the **python cdlib library**.

1. Louvain Community Detection

Louvain community detection algorithm was originally proposed in 2008 as a fast community unfolding method for large networks. This approach is **based on modularity**, which tries to maximize the difference between the actual number of edges in a community and the expected number of edges in the community. However **optimizing modularity in a network is NP-hard, therefore have to use heuristics**. Louvain algorithm is divided into **iteratively repeating two phases**;

1. Local moving of nodes

2. Aggregation of the network

Open in app ↗

Sign up

Sign in



Search



node, it considered the neighbours and evaluate the gain of modularity by removing the particular node from the current community and placing in the neighbour's community. The node will be placed in the neighbour's community if the gain is positive and maximized. The node will remain in the same community if there is no positive gain. This process is applied repeatedly and for all nodes until no further improvement is there. The first phase of the Louvain algorithm **stops when a local maxima of modularity is obtained**. In the second phase, the algorithm builds a new network considering communities found in the first phase as nodes. Once the second phase is completed, the algorithm will reapply the first phase to the resulting network. These steps are repeated until there are no changes in the network and maximum modularity is obtained.

Louvain community detection algorithm **uncovers communities of communities** during the process. It is very popular because of the ease of implementation and also the speed of the algorithm. However, one major **limitation** of the algorithm is the **use of storage of the network in the main memory**.

The use of the Louvain community detection algorithm using the python cdlib library is given below.

```
from cdlib import algorithms
import networkx as nx
G = nx.karate_club_graph()
coms = algorithms.louvain(G, weight='weight', resolution=1., randomize=False)
```

2. Surprise Community Detection

Due to limitations of the modularity, a **measure based on classical probabilities** known as Surprise has been introduced to evaluate the quality of a partition of a network into communities. The algorithm is almost similar to the Louvain community detection algorithm except that it **uses surprises instead of modularity**. Nodes are moved from one community to another such that surprises are greedily improved. This approach considers the probability that a link lies within a community. The use of surprises **works well in the limit of many small communities** and the use of modularity works well in the limit of a few large communities.

The use of the Surprise community detection algorithm using the python cdlib library is given below.

```
from cdlib import algorithms
import networkx as nx
G = nx.karate_club_graph()
coms = algorithms.surprise_communities(G)
```

3. Leiden Community Detection

In later research (2019), V.A. Traag et al. showed that **Louvain community detection has a tendency to discover communities that are internally disconnected** (badly connected communities). In the Louvain algorithm, moving a node which has acted as a bridge between two components in a community to a new community may disconnect the old community. This won't be a problem if the old community is being further split. But according to Traag et al., this won't be the case. Other nodes in the old community allow it to remain as a single community due to their strong connections. Also according to them, **Louvain has a tendency to discover very weakly connected communities**. Therefore, they have proposed the much faster Leiden algorithm which guarantees that communities are well connected.

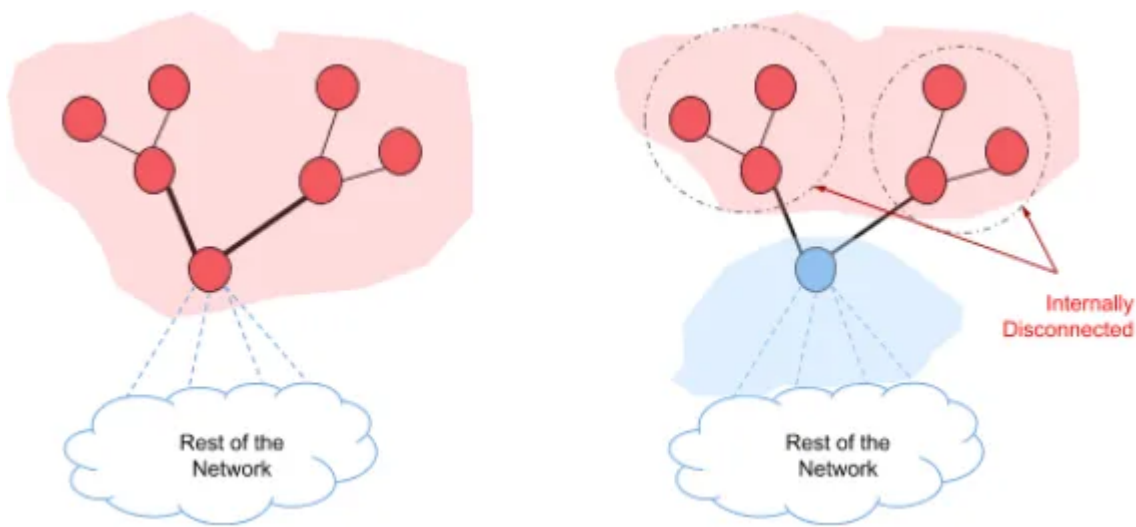


Image adapted from [4]

Additionally to the phases used in Louvain algorithm, **Leiden uses one more phase** which tries to refine the discovered partitions. The three-phases in Leiden algorithm are,

1. Local moving of nodes
2. Refinement of the partitions
3. Aggregation of the network based on refined partitions

In the refinement phase, the algorithm tries to identify refined partitions from the partitions proposed by the first phase. **Communities proposed by the first phase may further split into multiple partitions in the second phase.** The refinement phase does not follow a greedy approach and may merge a node with a randomly chosen community which increases the quality function. This **randomness allows discovering the partition space more broadly.** Also in the first phase, Leiden follows a different approach to the Louvain. **Instead of visiting all the nodes in the network after the first visit to all nodes has been completed, Leiden only visits those nodes whose neighbourhood has changed.**

The use of the Leiden community detection algorithm using the python cdlib library is given below.

```
from cdlib import algorithms
import networkx as nx
G = nx.karate_club_graph()
coms = algorithms.leiden(G)
```

4. Walktrap Community Detection

Walktrap is another approach for community detection **based on random walks** in which distance between vertices are measured through random walks in the network. Walktrap is an efficient algorithm which runs in $O(mn^2)$ time complexity and $O(n^2)$ space complexity in the worst case. But in most real-world scenarios, walktrap runs in $O((n^2) \log n)$ time complexity and $O(n^2)$ space complexity. The basic intuition of the algorithm is that **random walks on a graph/ network tend to get trapped into densely connected parts corresponding to communities**. Walktrap uses the result of random walks to merge separate communities in a bottom-up manner. Quality of the partitions can be evaluated using any available quality criterion. It can be either modularity as in Louvain community detection or any other measure.

The use of the Walktrap community detection algorithm using the python cdlib library is given below.

```
from cdlib import algorithms
import networkx as nx
G = nx.karate_club_graph()
coms = algorithms.walktrap(G)
```

Conclusion

Community detection is very applicable in understanding and evaluating the structure of large and complex networks. This approach uses the properties of edges in graphs or networks and hence more suitable for network analysis rather than a clustering approach. The clustering algorithms have a tendency to separate single peripheral nodes from the communities it should belong to. Many different algorithms have proposed and implemented for network community detection. Each of these has various pros and cons depending on the nature of the network as well as the applying problem domain.

References

- [1] Girvan, Michelle & Newman, Mark. (2001). "Community structure in social and biological networks," *proc natl acad sci*. 99. 7821–7826.
- [2] Blondel, V., Guillaume, J., Lambiotte, R. and Lefebvre, E., 2008. Fast unfolding of communities in large networks. *IOPscience*.
- [3] Traag, V., Aldecoa, R. and Delvenne, J., 2015. Detecting communities using asymptotical surprise. *PHYSICAL REVIEW*.

[4] V. A. Traag, L. Waltman, and N. J. van Eck, "From Louvain to Leiden: guaranteeing well-connected communities," Sci. Rep., vol. 9, no. 1, pp. 1–12, 2019, doi: 10.1038/s41598-019-41695-z.

[5] Pons, P. and Latapy, M., n.d. Computing communities in large networks using random walks.

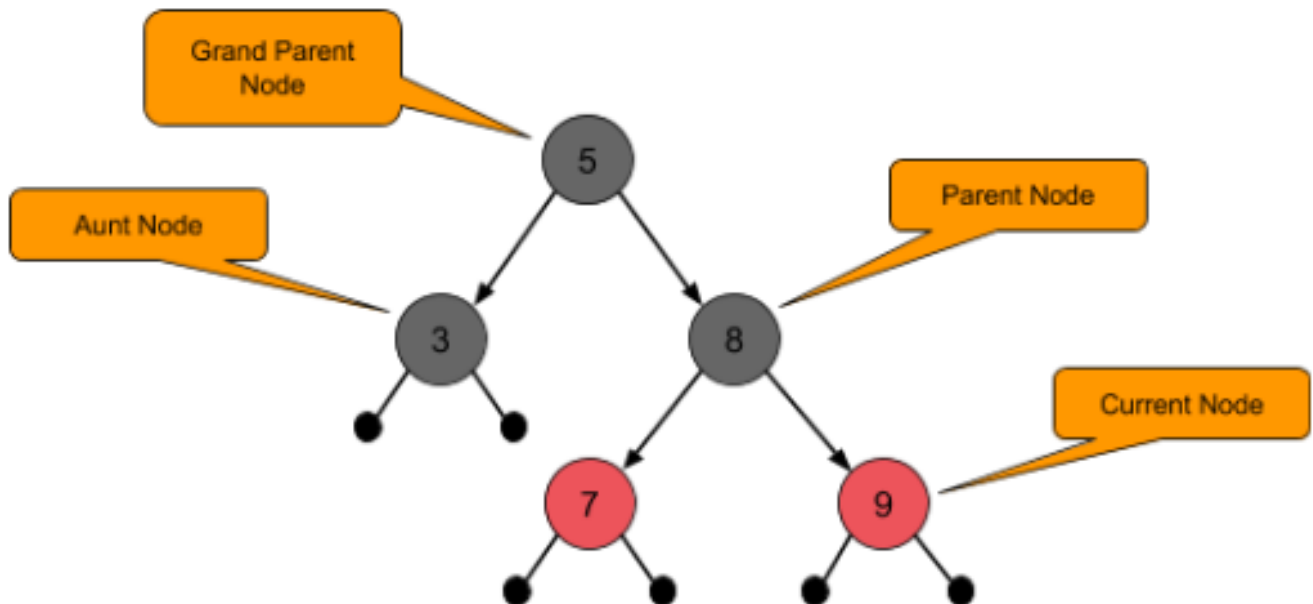
[Community Detection](#)[Louvain](#)[Leiden](#)[Getting Started](#)[Follow](#)

Written by Thamindu Dilshan Jayawickrama

112 Followers · Writer for Towards Data Science

Senior Software Engineer at WSO2 LLC. | B. Sc in Engineering (Hons), Computer Science and Engineering, University of Moratuwa

More from Thamindu Dilshan Jayawickrama and Towards Data Science



Thamindu Dilshan Jayawickrama in The Startup

Red Black Tree Rotations and Color Flips

A typical search tree like Binary Search Tree (BST) could run into a height of $O(n)$ which could result in worst-case time complexity of...

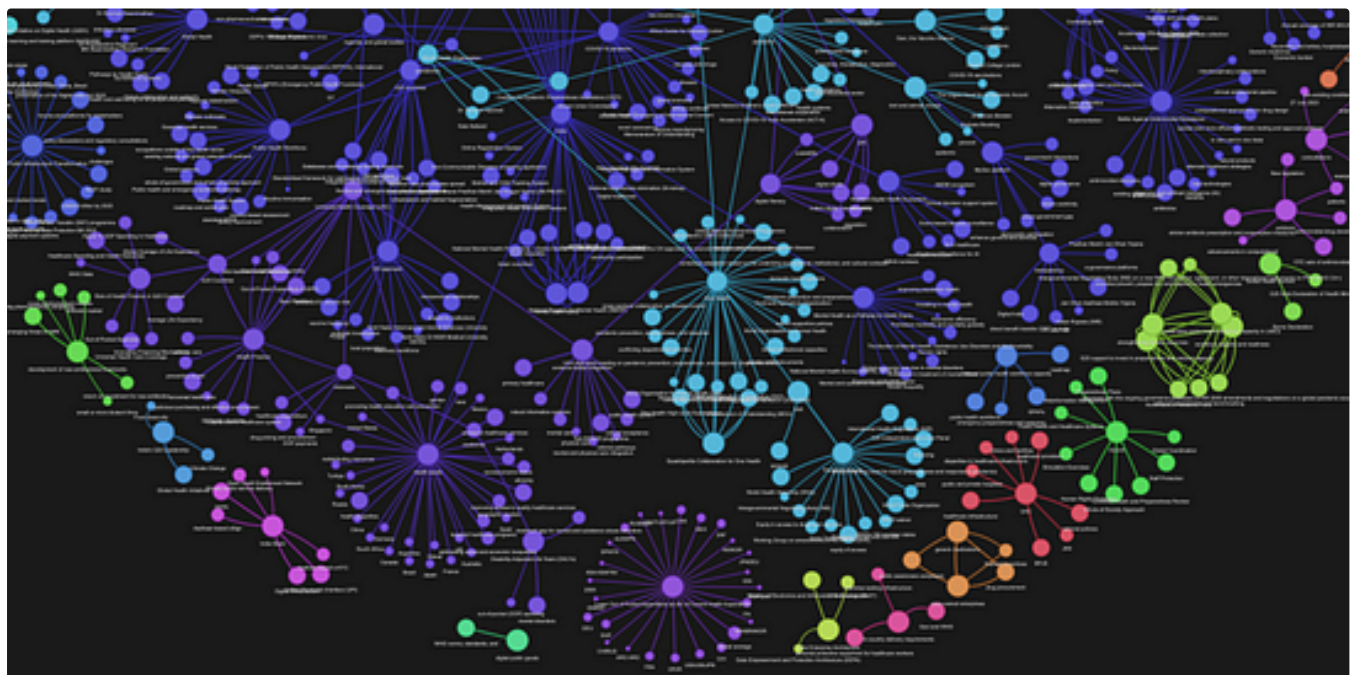
🌟 · 6 min read · Jul 11, 2020



452



1



Rahul Nayak in Towards Data Science

How to Convert Any Text Into a Graph of Concepts

<https://towardsdatascience.com/community-detection-algorithms-9bd8951e7dae>

8/14

A method to convert any text corpus into a Knowledge Graph using Mistral 7B.

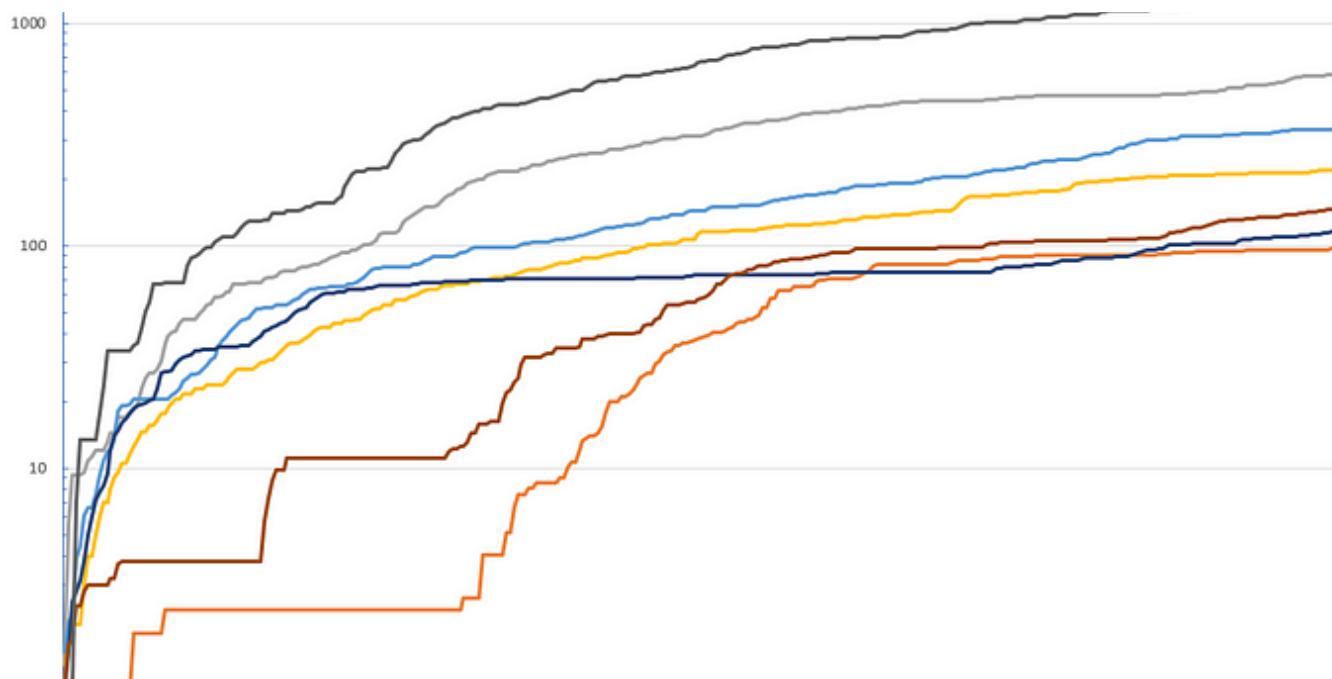
12 min read · Nov 10



3.8K



41



Pau Blasco i Roca in Towards Data Science

My Life Stats: I Tracked My Habits for a Year, and This Is What I Learned

I measured the time I spent on my daily activities (studying, doing sports, socializing, sleeping...) for 332 days in a row.

12 min read · Nov 21



4.2K



79





 Thamindu Dilshan Jayawickrama in Towards Data Science

Feature Engineering for Election Result Prediction (Python)

First, we should load the dataset into a notebook. Most ML engineers and experts prefer to use notebooks for machine learning and data...

🌟 • 13 min read • Apr 6, 2020

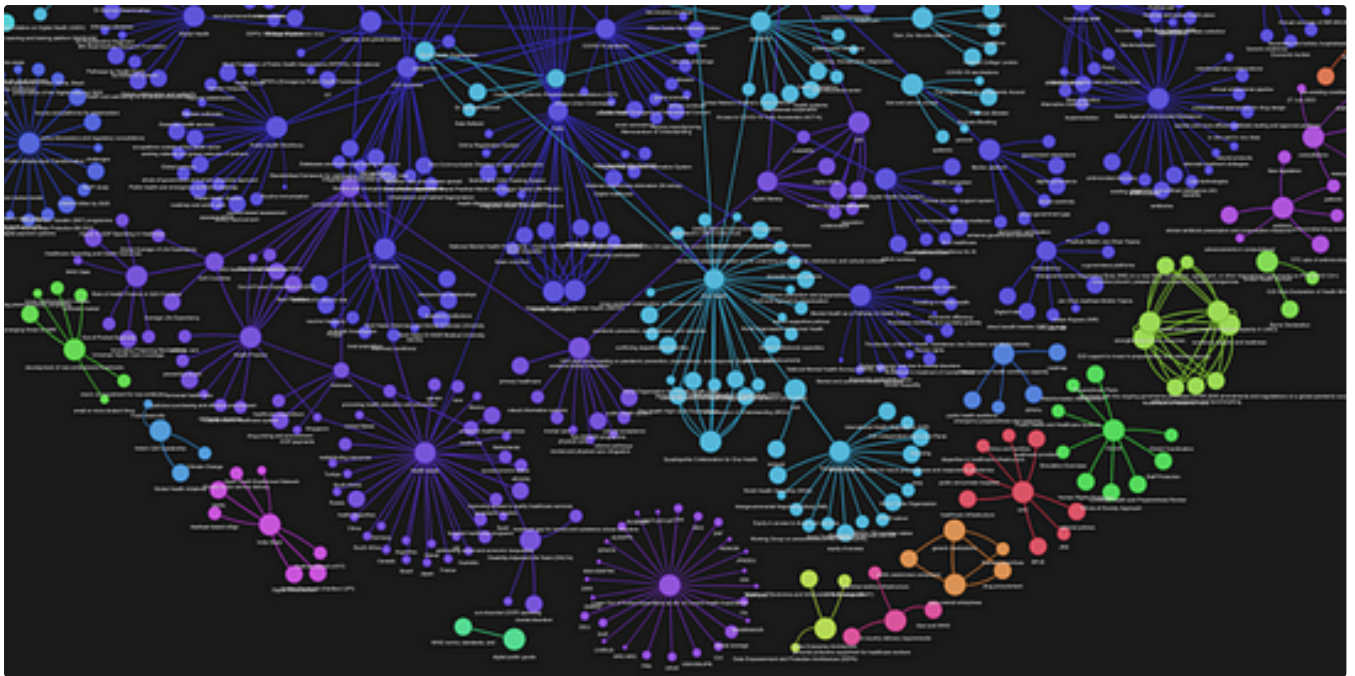
 133



See all from Thamindu Dilshan Jayawickrama

See all from Towards Data Science

Recommended from Medium



Rahul Nayak in Towards Data Science

How to Convert Any Text Into a Graph of Concepts

A method to convert any text corpus into a Knowledge Graph using Mistral 7B.

12 min read · Nov 10



3.8K



41




Vishal Sharma in Nerd For Tech

Comparative Analysis of Degree Centrality and Betweenness Centrality in Large Graphs


Centrality measures play a crucial role in network analysis, helping us identify the most important nodes within a network. Two commonly...


6 min read · Oct 8




Lists

- 

Natural Language Processing
956 stories · 458 saves
- 


New_Reading_List
174 stories · 221 saves

 Harry Powell

ML Graph Feature Extraction

One of the quickest and easiest ways to boost the performance of machine learning models.

6 min read · Aug 30

 3 



 Francesco Franco


Affinity Propagation with Python and Scikit-learn

Say you've got a dataset where there exist relationships between individual samples, and your goal is to identify groups of related samples...

9 min read · Nov 2



21

 Patrisha Estrada

Connecting the Dots: Understanding Social Media Dynamics through Network Science


Unravel social media dynamics with network science. Let's explore the connections that shape our digital discourse.

11 min read · Aug 4

 8





 Lei Wang in graphscope

Processing 100-billion edges in one second: Empowering Graphalytics with GPU Acceleration

Graph algorithms serve as essential building blocks for a wide range of applications, such as social network analytics, routing...

8 min read · Aug 8

 5

 1



See more recommendations