

HybridRAG: A Fusion of Graph and Vector Retrieval - Mitesh Patel, NVIDIA



AI Engineer
247K subscribers

Subscribe

235



Share

Ask

Download



8,818 views Jul 22, 2025 [AIEWF 2025 Complete Playlist](#)

Interpreting complex information from unstructured text data poses significant challenges to Large Language Models (LLM), with difficulties often arising from specialized terminology and the multifaceted relationships between entities in document architectures. Conventional Retrieval Augmented Generation (RAG) methods face limitations in capturing these nuanced interactions, leading to suboptimal performance. In our talk, we introduce a novel approach integrating Knowledge Graph-based RAG (GraphRAG) with VectorRAG, designed to refine question-answering (Q&A) systems for more effective information extraction from complex texts. Our approach employs a dual retrieval strategy that harnesses both knowledge graphs and vector databases, enabling the generation of precise and contextually appropriate answers, thereby setting a new standard for LLMs in processing sophisticated data.

About Mitesh Patel

Mitesh Patel is a developer advocate manager at NVIDIA. His team is responsible for creating workflows to showcase how developers can harness GPU acceleration in their workflows using tools and frameworks popular in the developer community. Before NVIDIA, he was a senior research scientist at Fuji Xerox Palo Alto Laboratory Inc. (a research subsidiary of Fuji Xerox), where he worked on developing indoor localization technologies for applications such as asset tracking in hospitals and delivery cart tracking in manufacturing facilities. Mitesh received his Ph.D. in Robotics from the Center of Autonomous Systems (CAS) at the University of Technology Sydney, Australia in 2014.

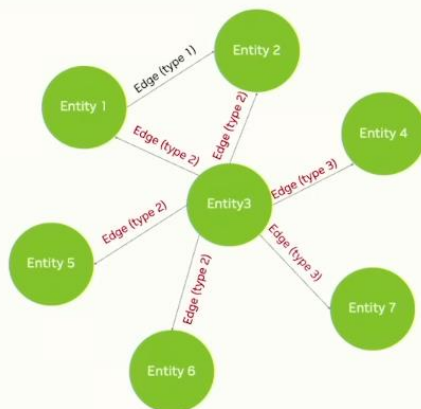


HybridRAG: A fusion of Graph and Vector Retrieval to Enhance Data Interpretation

Mitesh Patel, Sr. Manager – Developer Advocate

What is Knowledge Graph ?

A refresher

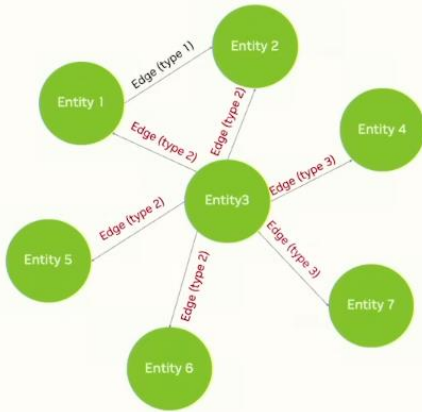


- Network that represent relationships between different entities
- Entities can be objects, places, people, concepts or events
- The edges represents the relationship between entities
- "Triplets" : [Entity 1 – Relationship – Entity 2]

```
[['Exxon Mobil', 'COMP', 'Cut', 'Spending on oil and gas exploration', 'ACTIVITY']]
```

What is Unique about Knowledge Graph

Where do they Excel



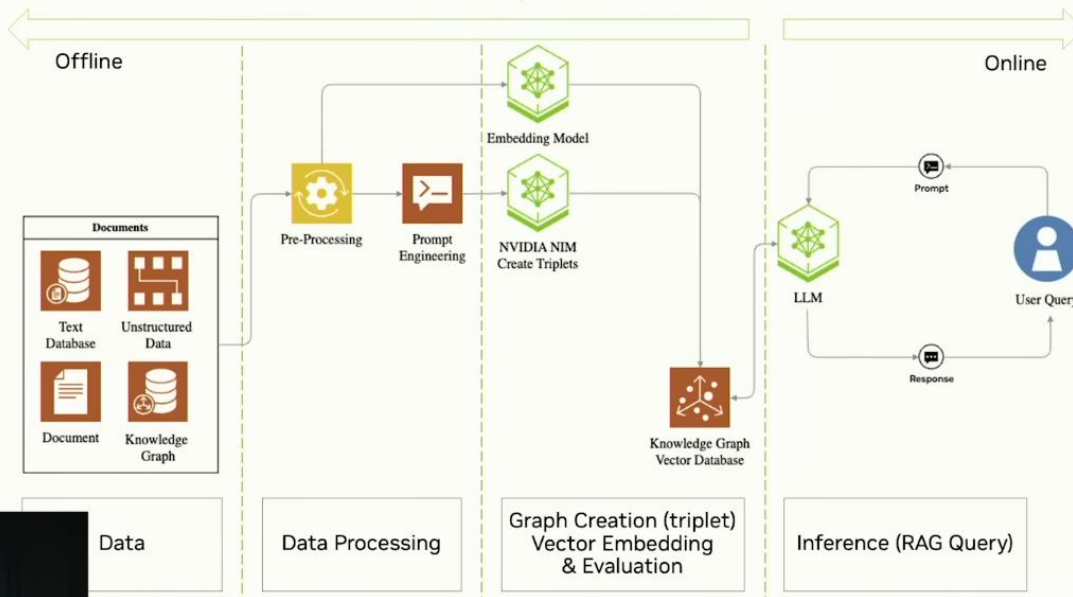
- Capture information about entities for a given domain or task
- Entity connection gives a comprehensive view of the knowledge base
- Knowledge Graph organize data from multiple sources

```
[['Exxon Mobil', 'COMP', 'Cut', 'Spending on oil and gas exploration', 'ACTIVITY']]
```

NVIDIA

How to Create a GraphRAG (Hybrid) System

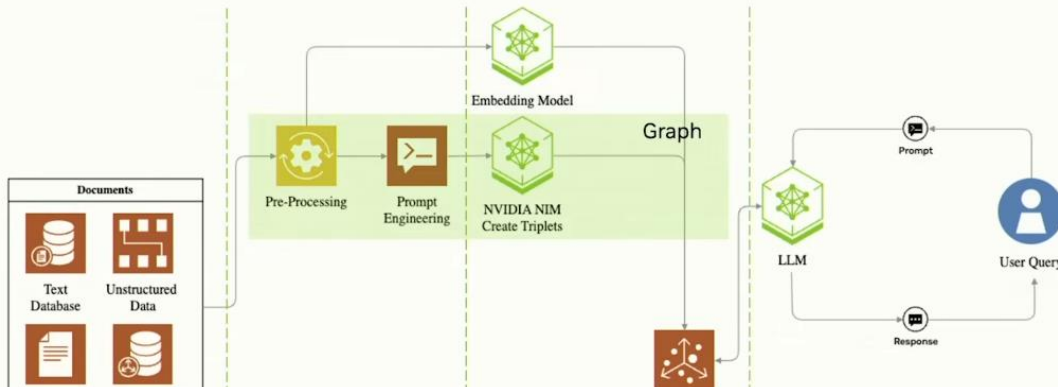
Components



NVIDIA

How to Create a GraphRAG (Hybrid) System

Components



Creating the Triplets

Text

LONDON (Reuters) - Despite the strongest start for oil prices in four years, the world's top oil companies are hesitating to accelerate the search for new resources as a determination to retain capital discipline trumps the hope of making bonanza discoveries.

Exxon Mobil, Royal Dutch Shell, Total and their peers are set to cut spending on oil and gas exploration for a fifth year in a row in 2018, according to consultancy Wood Mackenzie (WoodMac), despite a growing urgency to replenish reserves after years of reining back investment.

(For graphic 'Global spending on oil and gas exploration' click reut.rs/2CjAONv)

Global investment in exploration, vital to increase output and offset the natural decline of existing fields, will reach \$37 billion in 2018, down 7 percent from a year earlier and over 60 percent below the 2014 peak, according to WoodMac.

Triplets

```
[["Exxon Mobil", "COMP", "Cut", "Spending on oil and gas exploration", "ACTIVITY"],
["Royal Dutch Shell", "COMP", "Cut", "Spending on oil and gas exploration", "ACTIVITY"],
["Total", "COMP", "Cut", "Spending on oil and gas exploration", "ACTIVITY"],
["World's top oil companies", "ORG", "Hesitate", "Accelerate the search for new resources", "ACTIVITY"],
["Consultancy Wood Mackenzie", "ORG", "Estimate", "Global investment in exploration", "ECON_INDICATOR"],
["Global investment in exploration", "ECON_INDICATOR", "Reach", "$37 billion", "VALUE"],
["Global investment in exploration", "ECON_INDICATOR", "Decrease", "7 percent", "PERCENTAGE"]]
```

Triplets expose the relationship/information between 2 entities, this information is helpful. We can use LLMs to extract this information for us to save it in our KB in triplet format.

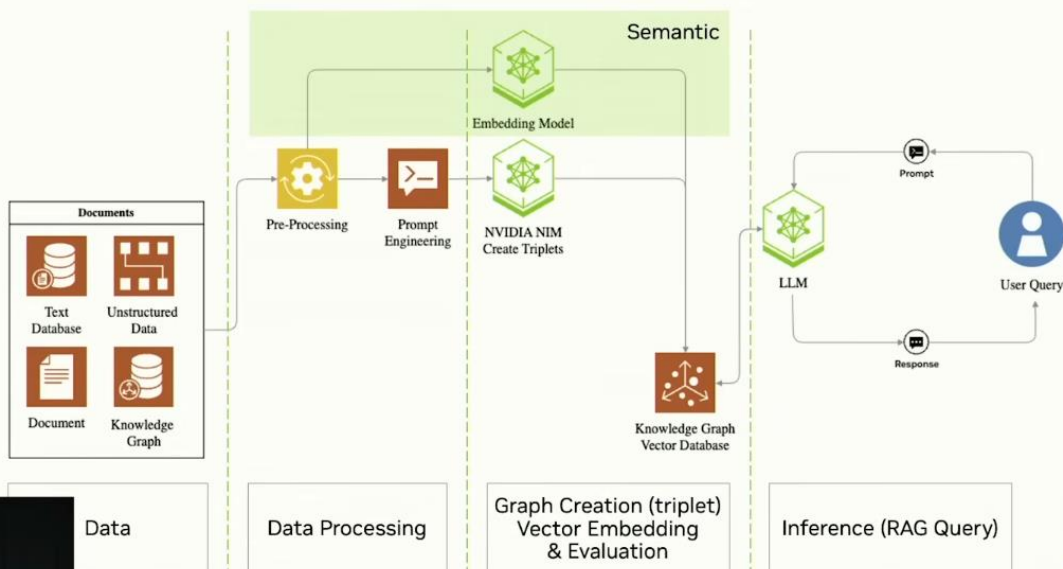
Creating the Triplets

Prompts is all that Matters

```
def extract_triples(text, llm):
    prompt = ChatPromptTemplate.from_messages(
        [
            ("system", """Note that the entities should not be generic, numerical, or temporal (like dates or percentage)
            - ORG: Organizations other than government or regulatory bodies
            - ORG/GOV: Government bodies (e.g., "United States Government")
            - ORG/REG: Regulatory bodies (e.g., "Food and Drug Administration")
            - PERSON: Individuals (e.g., "Marie Curie")
            - GPE: Geopolitical entities such as countries, cities, etc. (e.g., "Germany")
            - INSTITUTION: Academic or research institutions (e.g., "Harvard University")
            - PRODUCT: Products or services (e.g., "CRISPR technology")
            - EVENT: Specific and Material Events (e.g., "Nobel Prize", "COVID-19 pandemic")
            - FIELD: Academic fields or disciplines (e.g., "Quantum Physics")
            - METRIC: Research metrics or indicators (e.g., "Impact Factor"), numerical values like "10%" is not a METRIC;
            - TOOL: Research tools or methods (e.g., "Gene Sequencing", "Surveys")
            - CONCEPT: Abstract ideas or notions or themes (e.g., "Quantum Entanglement", "Climate Change")
            """)
        ])
    return llm.generate(prompt, text)
```

How to Create a GraphRAG (Hybrid) System

Components



Create Semantic Vector Database

- Document chunking size
- Chunking overlap size
- Embedding Model

The diagram illustrates the process of creating a semantic vector database. It starts with a document being split into chunks of a specific size, with an overlap between adjacent chunks. These chunks are then processed by an embedding model to generate semantic vectors. A detailed view of the Transformer architecture is provided, showing its internal components and the flow of information.

1 Introduction

Recent advances in long short-term memory [13] and gated recurrent [7] neural networks in particular have firmly established as state of the art approaches in sequence modeling and transduction problems such as language modeling and machine translation [33, 2, 5]. Numerous efforts have since continued to push the boundaries of recurrent language models and encoder-decoder architectures [34, 24, 15].

Recurrent models typically factor computation along the symbol positions of the input and output sequences. Aligning the positions to steps in computation time by generating a sequence of hidden states h_t as a function of the previous hidden state h_{t-1} and the input for position t . This inherently sequential nature precludes parallelization within training examples, which becomes critical at longer sequence lengths, as memory constraints limit batching across examples. Recent work has achieved significant improvement in computational efficiency through factorization tricks [23] and conditional computation [32], while also improving model performance in case of the latter. The fundamental constraint of sequential computation, however, remains.

Attention mechanisms have become an integral part of compelling sequence modeling and transduction models in various tasks, allowing modeling of dependencies without regard to their distance in the input or output sequences [2, 19]. In all but a few cases [27], however, such attention mechanisms are used in conjunction with a recurrent network.

In this work we propose the Transformer, a model architecture eschewing recurrence and instead relying entirely on an attention mechanism to draw global dependencies between input and output. The Transformer allows for significantly more parallelization and can reach a new state of the art in translation quality after being trained for as little as twelve hours on eight P100 GPUs.

2 Background

The goal of reducing sequential computation also forms the foundation of the Extended Neural GPU [16], ByteNet [18] and ConVSeq [9], all of which use convolutional neural networks as basic building block, computing hidden representations in parallel for all input and output positions. In these models, the number of operations required to relate signals from two arbitrary input or output positions grows in the distance between positions, linearly for ConVSeq and logarithmically for ByteNet. This makes it more difficult to learn dependencies between distant positions [12]. In the Transformer this is reduced to a constant number of operations, albeit at the cost of reduced effective resolution due to averaging attention-weighted positions, an effect we counteract with Multi-Head Attention as described in section 3.2.

Self-attention, sometimes called intra-attention is an attention mechanism relating different positions of a single sequence in order to compute a representation of the sequence. Self-attention has been used successfully in a variety of tasks including reading comprehension, abstractive summarization, textual entailment and learning task-independent sentence representations [4, 27, 28, 22].

First-order memory networks are based on a recurrent attention mechanism instead of sequence-aligned recurrence and have been shown to perform well on simple-language question answering and language modeling tasks [34].

To the best of our knowledge, however, the Transformer is the first transduction model relying entirely on self-attention to compute representations of its input and output without using sequence-aligned RNNs or convolution. In the following sections, we will describe the Transformer, motivate self-attention and discuss its advantages over models such as [17, 18] and [9].

3 Model Architecture

Most competitive neural sequence transduction models have an encoder-decoder structure [5, 2, 35]. Here, the encoder maps an input sequence of symbol representations (x_1, \dots, x_n) to a sequence of continuous representations $z = (z_1, \dots, z_n)$. Given z , the decoder then generates an output sequence (y_1, \dots, y_m) of symbols one element at a time. At each step the model is autoregressive [16], consuming the previously generated symbols as additional input when generating the next.

Accelerate : Knowledge Graph Retrieval Strategies

Knowledge Graphs can Increase RAG Accuracy

- Search beyond a single hop/node (Multi-hops):
 - Helps in retrieve all relevant nodes
 - Provides far greater context between entities and relations
- Accelerate Search on GPUs through cuGraph
 - Accelerate multi-hop & multi-step search
 - BFS/DFS strategies accelerated

```
graph TD; A(( )) --- B(( )); A --- C(( )); B --- D(( )); B --- E(( )); B --- F(( )); C --- G(( )); C --- H(( )); C --- I(( ));
```

You can query your data using different strategies like depth or breadth approaches

Evaluate Knowledge Graph

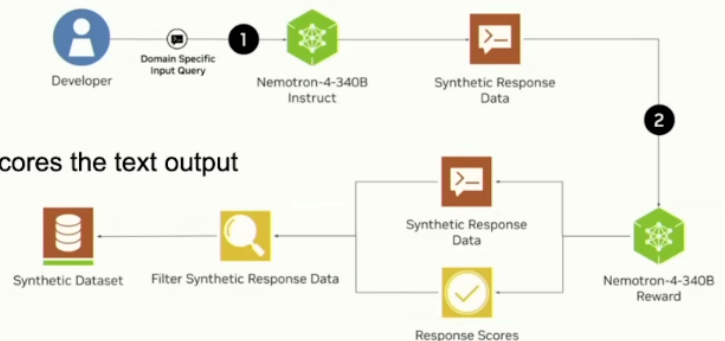
Is Your KG good enough

- RAGAS: evaluation framework for RAG systems

- Faithfulness
- Answer Relevancy
- Context Precision
- Context Recall

- Nemotron-340b-reward: A reward model that scores the text output

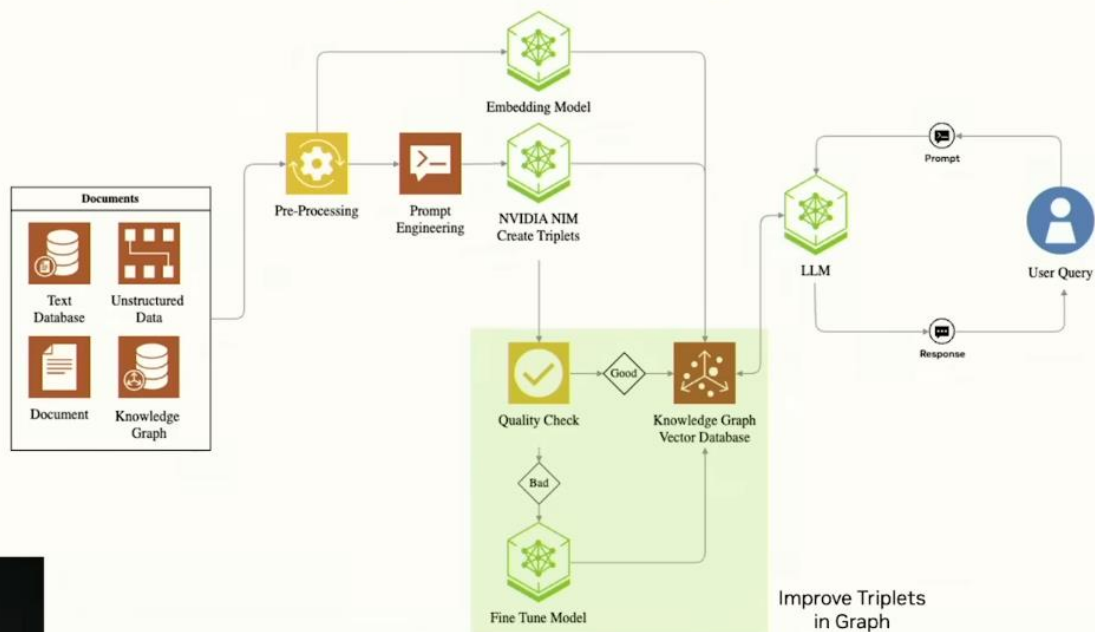
- Helpfulness
- Correctness
- Coherence
- Complexity
- Verbosity



RAGAS is a pip install library that allows you to bring your own model

The Last Mile

Improve Performance & Accuracy



The way you create your Knowledge Graph (KG) can help improve your system performance and results

The Last Mile

Improve Performance & Accuracy

Apostrophe

```
[['HP', 'COMP', 'Recall', 'Laptop Batteries', 'PRODUCT'],  
 ['Laptop Batteries', 'PRODUCT', 'Operate_In', 'All HP Laptops',  
 'GPE'], ['Affected Laptops', 'GPE', 'Has', 'Burn Hazard Risk',  
 'CONCEPT'], ['Eligible Batteries', 'PRODUCT', 'Replace',  
 'Free', 'FIN_INSTRUMENT'], ['Consumers', 'PERSON', 'Check',  
 'HP Website', 'ORG'], ['HP', 'COMP', 'Push', 'Battery Safety  
Mode Update', 'PRODUCT'], ['Affected Computers', 'GPE',  
 'Activate', 'Battery Safety Mode', 'PRODUCT']]
```

Add to prompt

Do not output apostrophe removes apostrophes for example
convert Apple's to Apple , Korea's to Korea etc

clean the output

```
.replace("'s", "s")
```

Really long output

```
[['I', 'PERSON', 'Visit', 'Puerto Rico', 'GPE'],  
 ['Hurricane Maria', 'EVENT', 'Relate_To', 'Puerto Rico',  
 'GPE'], ['I', 'PERSON', 'Feel', 'Guilty', 'CONCEPT'],  
 ['Everybody', 'PERSON', 'Exhausted', 'CONCEPT'],  
 ....  
 ....  
 ['U.S. Army Corps of Engineers', 'ORG', 'Classified',  
 'Total_Impact_On', 'Puerto Rico', 'GPE'], ['FEMA', 'ORG',  
 'Classified', 'Impacts', 'ECON_INDICATOR', 'Puerto Rico',  
 '..... // Really long output :- Incomplete output
```

Implement Correction

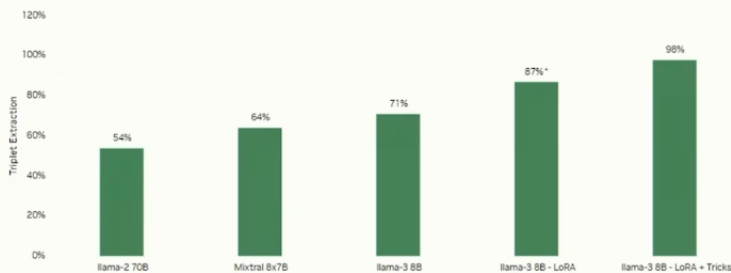
In case of missing entries at the end of list
truncate it for example

```
INPUT : [['Polaris Industries Inc.', 'COMP',  
 'Announce', 'fourth quarter and full-year 2017  
financial results', 'ECON_INDICATOR'], ['Polaris  
Industries Inc.', 'COMP', 'Hold', 'webcast and  
conference call', 'EVENT'], ['webcast and conference  
call', 'EVENT', 'Hosted_By', 'Scott Wine',  
 'PERSON']]
```

```
OUTPUT : [['Polaris Industries Inc.', 'COMP',  
 'Announce', 'fourth quarter and full-year 2017  
financial results', 'ECON_INDICATOR'], ['Polaris  
Industries Inc.', 'COMP', 'Hold', 'webcast and  
conference call', 'EVENT']]
```

The Last Mile

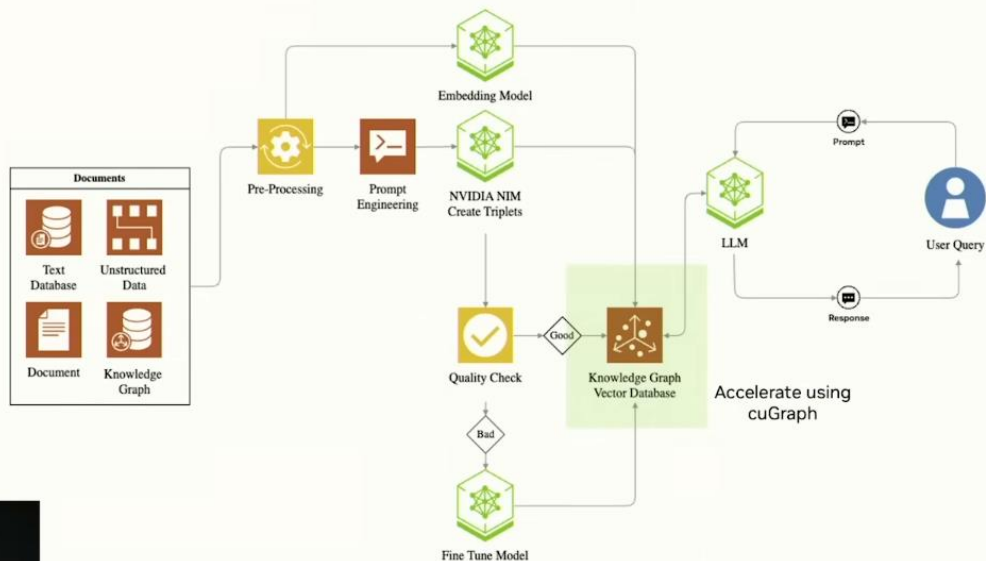
Improve Performance & Accuracy



Improvement in Accuracy over 100 document

The Last Mile

Improve Performance & Accuracy



If your graph gets really big with lots of nodes, you might start having latency and network issues. Tweak your logic and measure the network performance

The Last Mile

Improve Performance & Accuracy

The full end-to-end accelerated NetworkX experience is now enabled by setting the `NX_CUGRAPH_AUTOCONFIG` environment variable to True.

```
%env NX_CUGRAPH_AUTOCONFIG=True

import pandas as pd
import networkx as nx

url = "https://data.rapids.ai/cugraph/datasets/cit-Patents.csv"
df = pd.read_csv(url, sep=";", names=["src", "dst"], dtype="int32")
G = nx.from_pandas_edgelist(df, source="src", target="dst")

%time result = nx.betweenness_centrality(G, k=10)
```

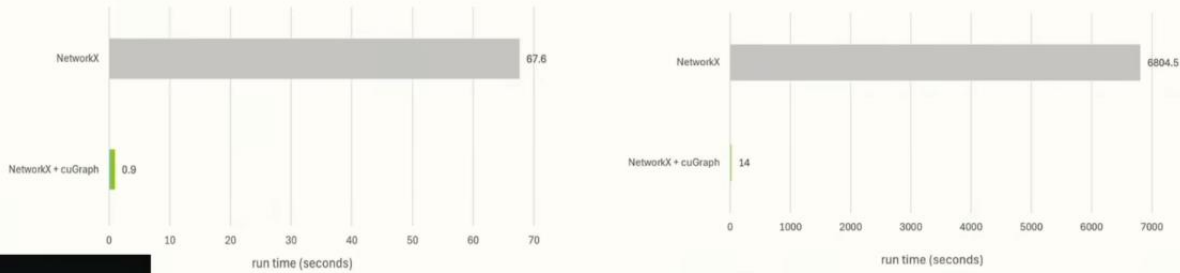


Figure 1. The betweenness centrality algorithm run on a citation graph of U.S. patents (4M nodes, 16M edges) is 70x faster than NetworkX on CPU.

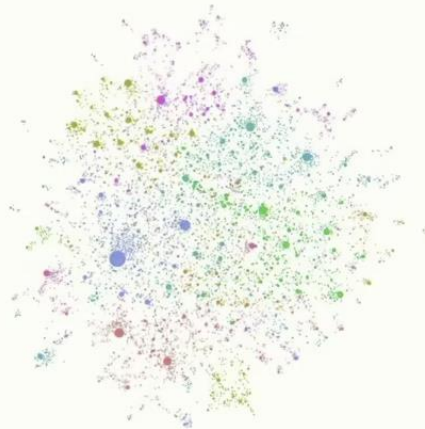
Figure 2. The betweenness centrality algorithm run on the Live Journal social network (5M nodes, 69M edges) is 485x faster than NetworkX on GPU for number of samples (k) set to 100. SW: NetworkX 3.4.1, cuGraph/cuGraph 24.10. GPU: NVIDIA A100 80GB; CPU: Intel Xeon w9-3495X (56 cores) 250GB



Should I Use Graph or Semantic or Hybrid

What to use ?

- Your data
 - Structured data
 - Semantic/Unstructured data
 - Can graph be created
- Application/Use case
 - Complex relationship
 - Semantic relationship
 - Latency



Developer Tools and Resources

Accelerate innovation and growth



Learn more: developer.nvidia.com

Individuals

Software

100s of APIs, models, SDKs, microservices, and early access to NVIDIA tech

Training

Hands-on self-paced courses, instructor-led workshops, and certifications

GPU Sandbox

Approval basis, multi-GPU and multi-node

Learning

Tutorials, self-paced courses, blogs, documentation, code samples

Community

Dedicated developer forums, meetups, hackathons

Ecosystem

GTC, NVIDIA Partner Network

Organizations

Startups

Cloud credits, engineering resources, technology discounts, exposure to VCs

Venture Capital

Deal flow and portfolio support for Venture Capital firms

Higher Education

Teaching kits, training, curriculum co-development, grants

ISVs and SIs

Engineering guidance, discounts, marketing opportunities

Research

Grant programs, collaboration opportunities

Enterprises

Tailored developer training, skills certification, technical support