



Martin O'Hanlon
Technical Curriculum Developer
Neo4j



GenAI Beyond Chat

... with RAG, Knowledge Graphs and Python

Welkom, ik ben Martin

Technical Curriculum Developer at Neo4j



@martinohanlon

GenAI Beyond Chat

GenerativeAI Beyond Chat

GenerativeAI Beyond Chat

Making LLMs make sense

Content Generation

Generate a personalised email to customers letting them know about an exciting new product called "GenAI.ai".

Of course. Here you go.

Attachment:  email.eml

Subject: 🚀 Introducing GenAI.ai: Your Next-Level AI Experience 🚀

Dear [Customer Name],

I hope this email finds you well. We're thrilled to announce the launch of something truly groundbreaking that we believe will revolutionize your digital experience – introducing GenAI.ai!

What is GenAI.ai?

GenAI.ai is an advanced AI-powered platform designed to enhance your productivity, streamline your workflows, and deliver powerful insights to take your business to the next level. We've combined cutting-edge technology with intuitive design to bring you the ultimate tool for success.

Here's what makes GenAI.ai so exciting:

- **Unmatched Automation:** Say goodbye to repetitive tasks. Automate and optimize your workflow like never before.
- **Intelligent Insights:** Harness the power of AI to uncover hidden patterns and opportunities in your data.
- **Seamless Integration:** Easily connect GenAI.ai with your existing systems to boost efficiency and collaboration.
- **User-Friendly Interface:** A sleek and intuitive design ensures you spend less time learning and more time achieving.

The problem

Why LLMs generate garbage

Prompt in > Response out (based on the training data)

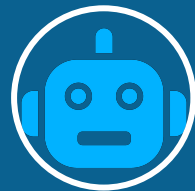
The background consists of several overlapping organic, blob-like shapes in various shades of blue, ranging from a light teal to a deep navy blue. The text is centered horizontally and partially overlaid by these shapes.

LLMs have a problem

<https://platform.openai.com/playground/chat?models=gpt-4>

The problem

What's the weather like?



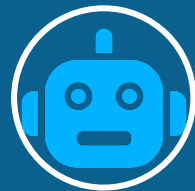
Sorry, as an AI, I don't have real-time information or access to live weather data. Please check a weather website or app for current conditions.



The problem

You are a surfer dude, having a conversation about the surf conditions on the beach.
Respond using surfer slang.

What's the weather like?



Dude, it's totally gnarly out here! The sun's gleaming, the water's blue, and the waves are firing! Totally glassy! So grab your stick, it's time to shred!



```
from langchain_openai import ChatOpenAI
from langchain_core.prompts import ChatPromptTemplate
from langchain.schema import StrOutputParser

chat_llm = ChatOpenAI(openai_api_key="sk-...")

prompt = ChatPromptTemplate.from_messages(
    [
        (
            "system",
            "You are a surfer dude, having a conversation about the surf conditions on the beach. Respond using surfer slang.",
        ),
        (
            "human",
            "{question}"
        ),
    ]
)

chat_chain = prompt | chat_llm | StrOutputParser()

response = chat_chain.invoke({"question": "What is the weather like?"})

print(response)
```



LLMs make things up

LLMs make things up



Hallucination

Hallucination (artificial intelligence)

Article [Talk](#)

From Wikipedia, the free encyclopedia

In the field of [artificial intelligence](#) (AI), a **hallucination** or **artificial hallucination** (also called **confabulation**^[1] or **delusion**^[2]) is a response generated by an AI which contains false or misleading information presented as fact.^{[3][4][5]} For example, a hallucinating [chatbot](#) might, when asked to generate a [financial report](#) for a company, falsely state that the company's revenue was \$13.6 billion (or some other number apparently "plucked from thin air").^[6] Such phenomena are termed "hallucinations", in loose analogy with the phenomenon of [hallucination in human psychology](#). However, one key difference is that human hallucination is usually associated with false *percepts*, but an AI hallucination is associated with the category of unjustified responses or beliefs.^[5] Some researchers believe the specific term "AI hallucination" unreasonably anthropomorphizes computers.^[1]

AI hallucination gained prominence around 2022 alongside the rollout of certain [large language models](#) (LLMs) such as [ChatGPT](#).^[7] Users complained that such bots often seemed to pointlessly embed plausible-sounding random falsehoods within their generated content.^[8] By 2023, analysts considered frequent hallucination to be a major problem in LLM technology, with some estimating chatbots hallucinate as much as 27% of the time.^{[9][10]}

LLMs are...

LLMs are designed to
generate human-like text

RAG

Use openai playground to have a conversation about surf conditions.

Add system messages - this is RAG

Show the Python

- But how do you know what to include in the context?

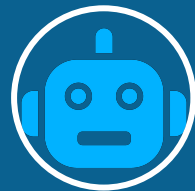
We are going to look at how we can use vectors and graphs to find relevant information to give context to the LLM.

A solution

You are a surfer dude, having a conversation about the surf conditions on the beach. Respond using surfer slang.

Current weather -calm, 3 feet waves, cloudy

What's the weather like?



**Man, it's kind of mellow out here.
We've got some ankle busters rolling
in, nothing too heavy, and the sky's
looking a bit grey. Might be a chill day
to just kick back and enjoy the vibe,
bro.**



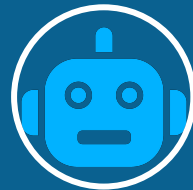
A solution

You are a surfer dude, having a conversation about the surf conditions on the beach. Respond using surfer slang.

Surf report:

```
{
  "surf": [
    {"beach": "Fistral", "conditions": "6ft waves and offshore winds"},
    {"beach": "Polzeath", "conditions": "Flat and calm"},
    {"beach": "Watergate Bay", "conditions": "3ft waves and onshore winds"}
  ]
}
```

What's happening at Fistral beach?



Dude, Fistral beach is totally tubular right now. We're talkin' 6ft waves and offshore winds. Perfect for some righteous carves!



```
prompt = ChatPromptTemplate.from_messages([
    (
        "system",
        "You are a surfer dude, having a conversation about the surf conditions on the beach. Respond using surfer slang.",
    ),
    ( "system", "{context}" ),
    ( "human", "{question}" ),
])

chat_chain = prompt | chat_llm | StrOutputParser()

current_weather = """
{
    "surf": [
        {"beach": "Fistral", "conditions": "6ft waves and offshore winds"},
        {"beach": "Polzeath", "conditions": "Flat and calm"},
        {"beach": "Watergate Bay", "conditions": "3ft waves and onshore winds"}
    ]
}"""

response = chat_chain.invoke(
    {
        "context": current_weather,
        "question": "What's it like at Watergate Bay"
    }
)
print(response)
```

Retrieval Augmented Generation

R

A

G

Retrieval

Augmented

Generation

Retrieval Augmented Generation

G A R

Retrieval Augmented Generation

G A R

Instruct an LLM
to generate text

Retrieval Augmented Generation

G A R

Instruct an LLM
to generate text

that has been
augmented or
improved

Retrieval Augmented Generation

G

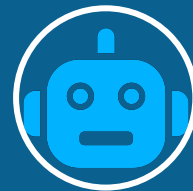
Instruct an LLM
to generate text

A

that has been
augmented or
improved

R

by the retrieval
of additional
information



What data should you include?



Graphs and vectors



What are **Graphs**?

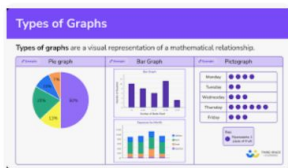




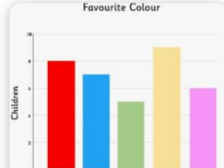
graphs

All Images Videos News Maps More

Tools



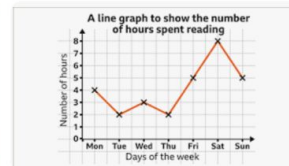
Third Space Learning
Types of Graphs - Math Steps, Examples ...



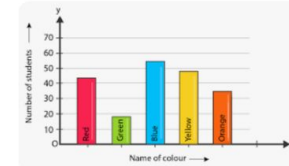
Twinkl
What is a Graph? - Maths - Twinkl



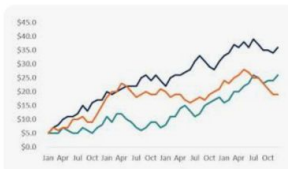
Cambridge Mathematics
Cambridge Mathematics



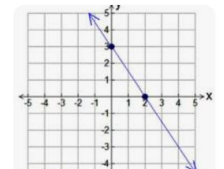
BBC
Line graphs - KS3 Maths - BBC Bitesize



BYJU'S
of Graphs in Mathematics and Statistics ...



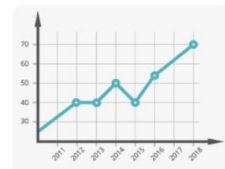
Corporate Finance Institute
Types of Graphs



MathBitsNotebook
Interpreting Graphs - MathBit...



Indeed
13 Types of Graphs and Charts (Plus ...



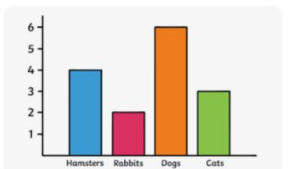
BYJU'S
of Graphs in Mathematics and S...



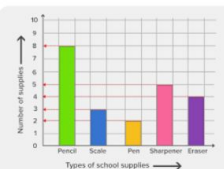
Carolina Knowledge ...
Graphs and Charts



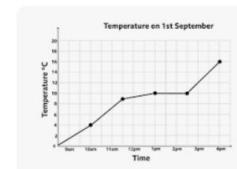
storytelling with data
verbalizing and explaining charts ...



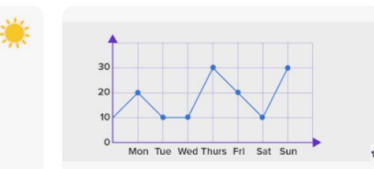
Twinkl
What is a graph? - Twinkl



SplashLearn
What is a Graph in Math? Definitio...



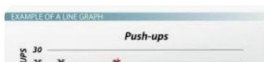
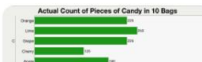
BBC
Introducing line graphs - Maths ...



SplashLearn
What is a Graph in Math? Definition ...



ChartExpo
Charts and Graphs for Data Visualization



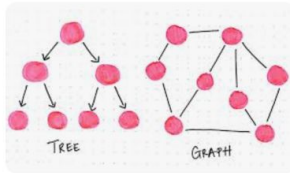


graph theory

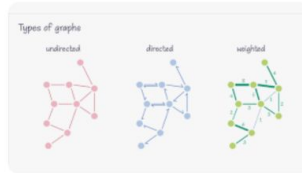
All Images Books Videos News More

Tools

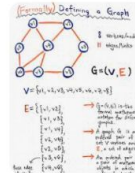
Computer science Discrete mathematics Path Brain Tree Simple Social network Connectivity



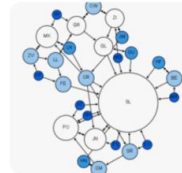
Medium
A Gentle Introduction To Graph Theory ...



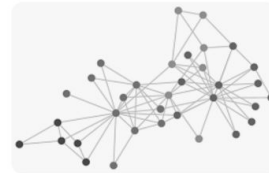
Science in the News - Harvard University
Graph Theory 101 - Science in the News



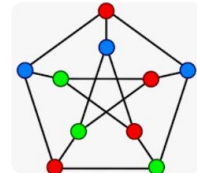
Medium
A Gentle Introduction To Graph Theory ...



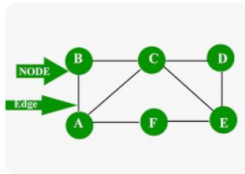
Wikipedia
Graph theory - Wikipedia



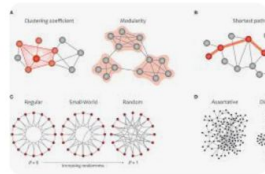
Adatis
Business Insights with Graph Theory ...



Brilliant
Graph Theory | Brilliant Mat...



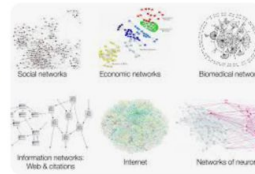
GeeksforGeeks
Mathematics | Graph Theory Basics ...



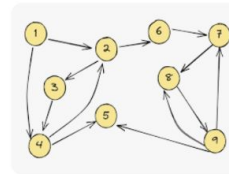
UCF College of Sciences
Dr. Richard contributes to new ...



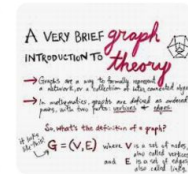
Science in the News - Harvard U...
Graph Theory 101 - Science i...



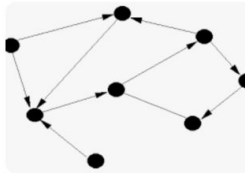
Towards Data Science
A Gentle Introduction to Graph theory ...



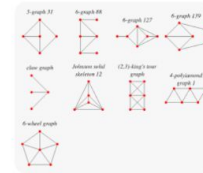
Medium - In Plain English
Graph Theory. 1. Introduction to G...



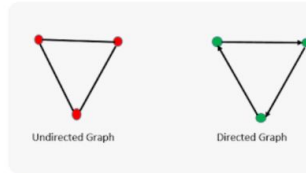
Medium
A Gentle Introduction To ...



people.brunel.ac.uk
Graph theory



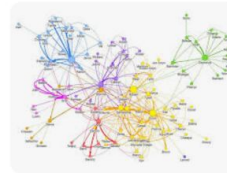
Wolfram MathWorld
Line Graph - from Wolfram ...



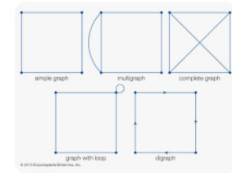
BYJU'S
Graph Theory-Discrete Mathematics ...



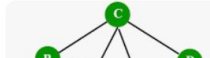
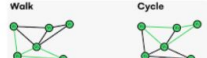
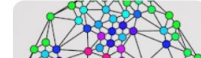
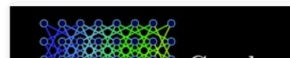
KDnuggets
What is Graph Theory, ...

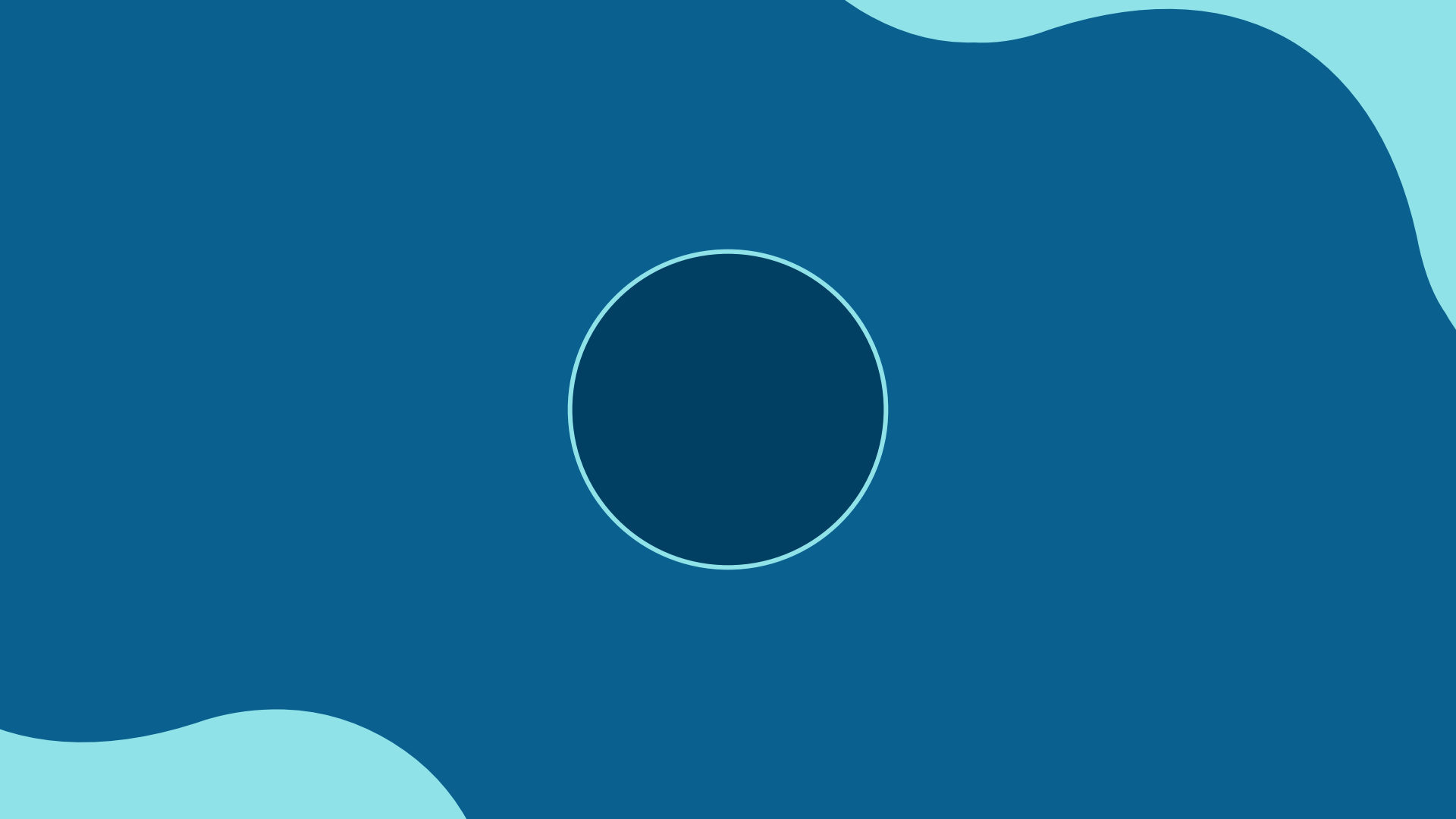


ResearchGate
Graph theory analysis in social ne...



Britannica
Graph theory | Problems & Appli...





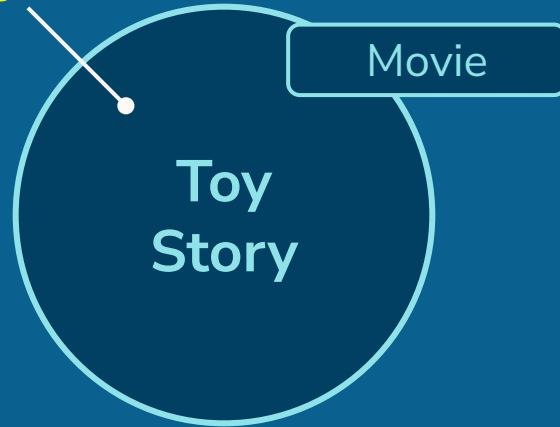
The image features a solid dark blue background with light blue wavy borders at the top and bottom. In the center is a dark blue circle with a thin white outline. Inside this circle, the words "Toy Story" are written in a white, sans-serif font, with "Toy" on the top line and "Story" on the bottom line.

Toy
Story

Nodes represent *things*

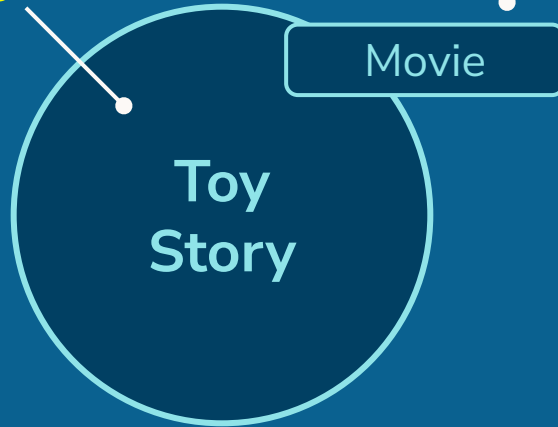


Nodes represent *things*

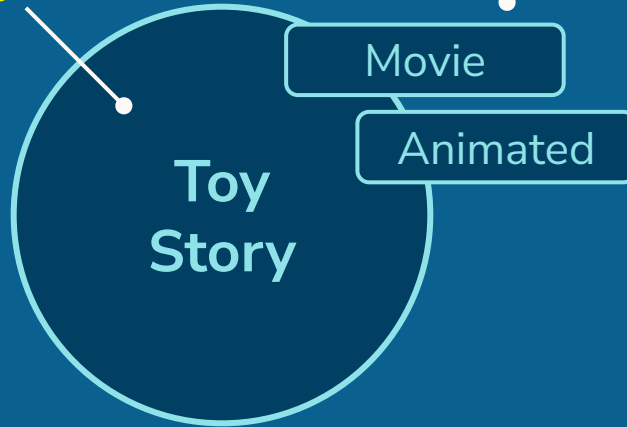


Nodes represent *things*

Nodes can be identified by
one or more *labels*



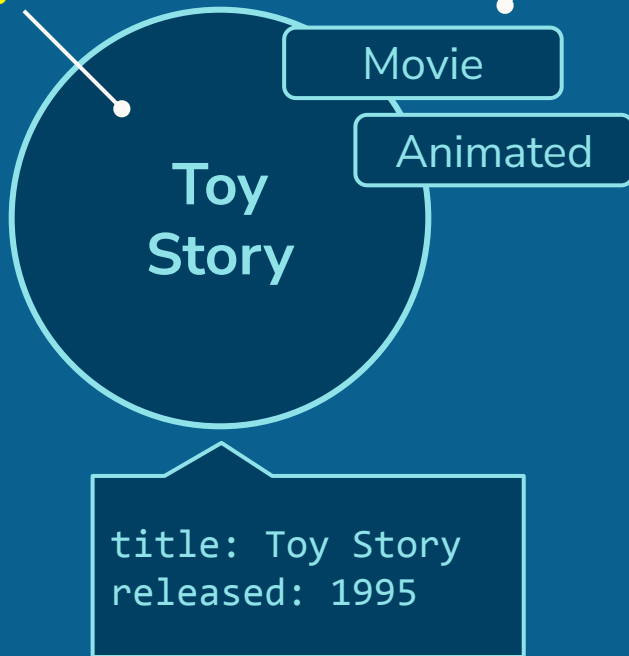
Nodes represent *things*



Nodes can be identified by one or more *labels*

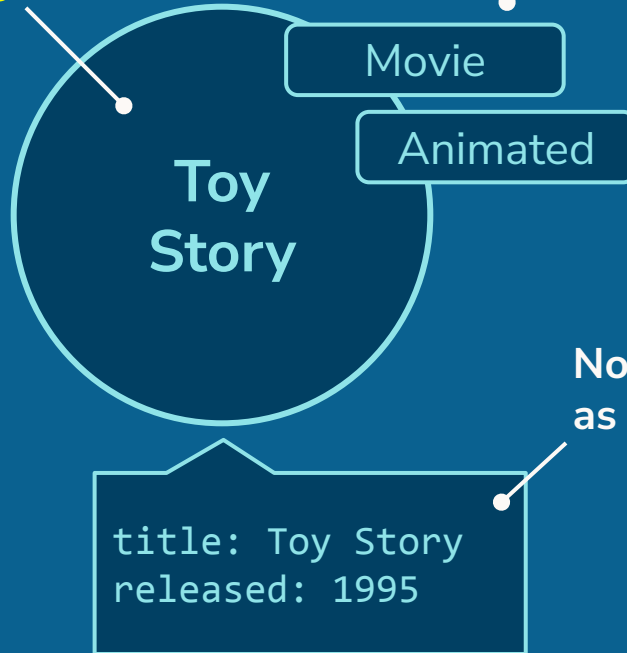
Nodes represent *things*

Nodes can be identified by one or more *labels*



Nodes represent *things*

Nodes can be identified by one or more *labels*



Nodes can hold *properties* as key/value pairs

**Toy
Story**

Movie

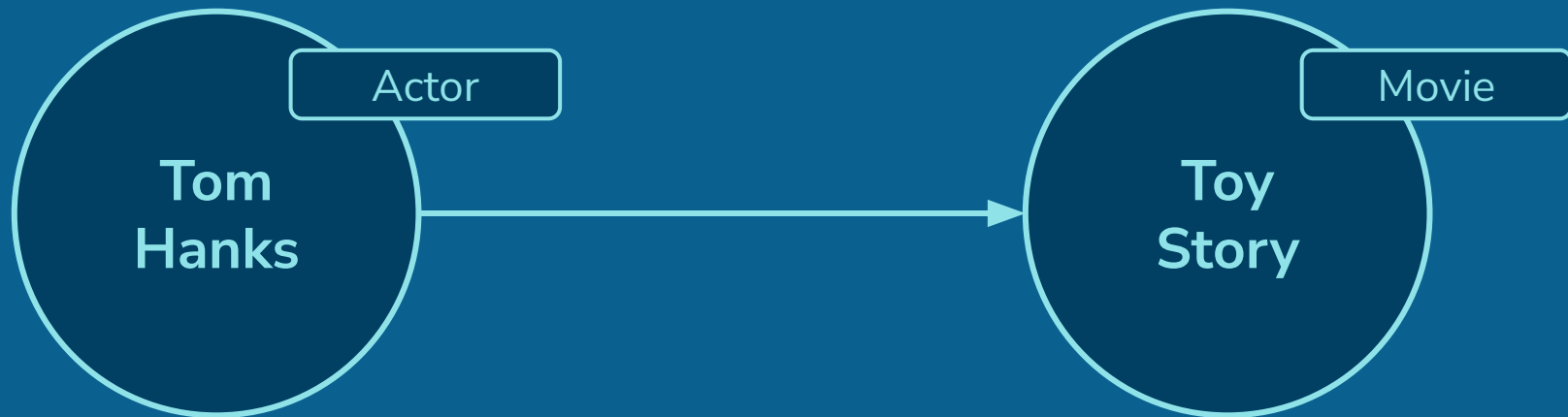
Actor

The image features a dark blue background with light blue wavy borders at the top and bottom. On the left, a dark blue circle contains the text 'Tom Hanks'. To its right, a light blue rounded rectangle contains the text 'Actor'. On the right, another dark blue circle contains the text 'Toy Story'. To its right, a light blue rounded rectangle contains the text 'Movie'.

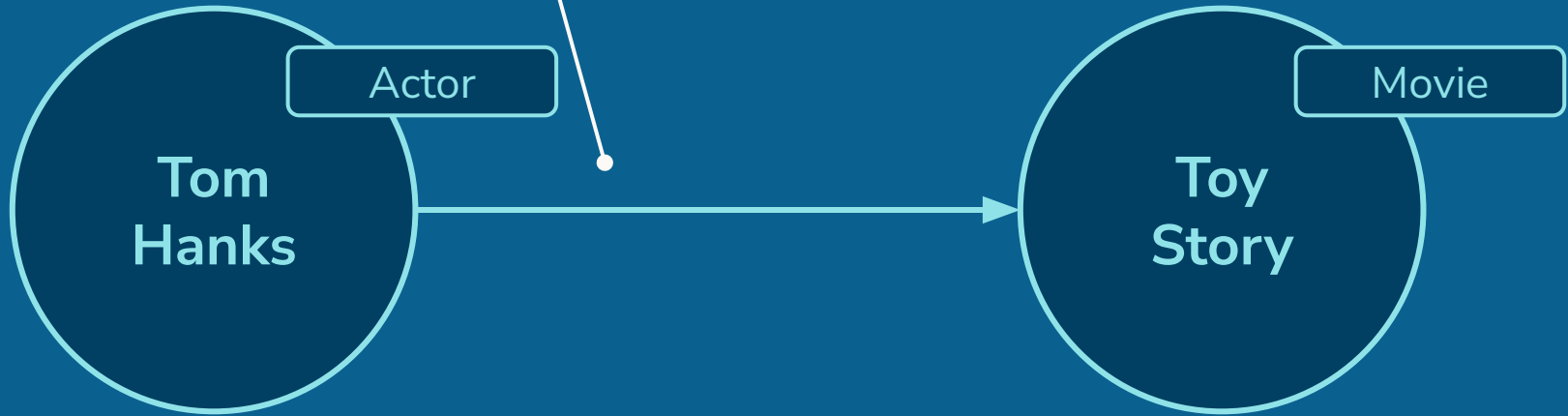
Tom
Hanks

Movie

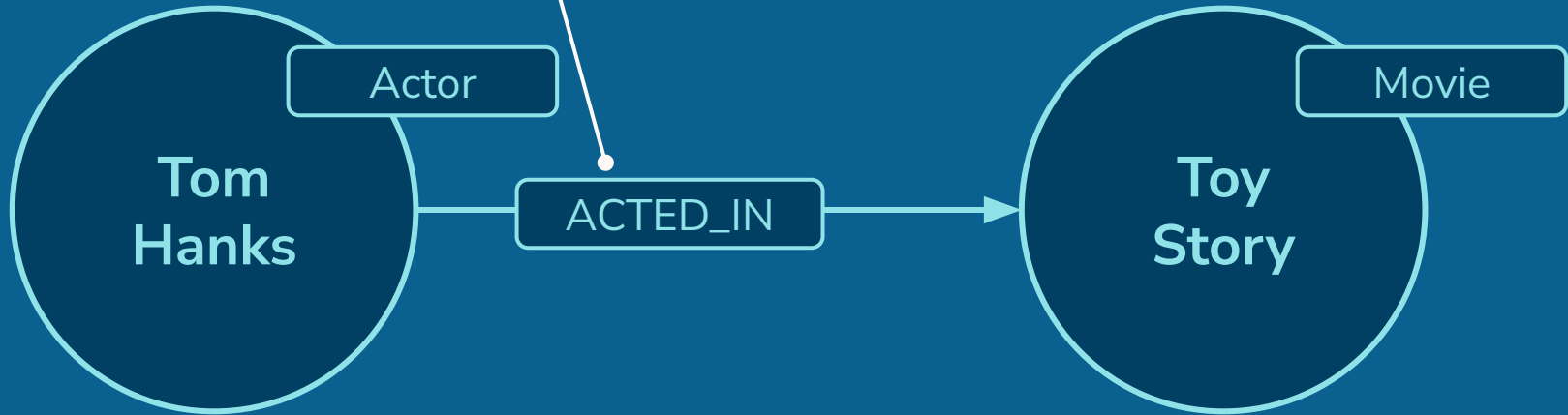
Toy
Story



Relationships connect *two nodes*

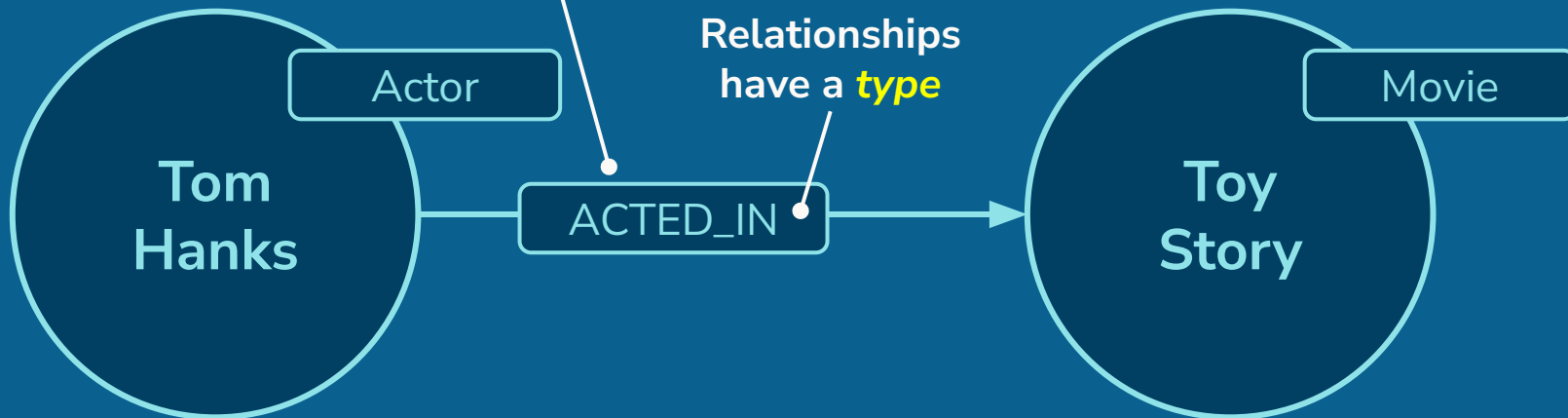


Relationships connect *two nodes*

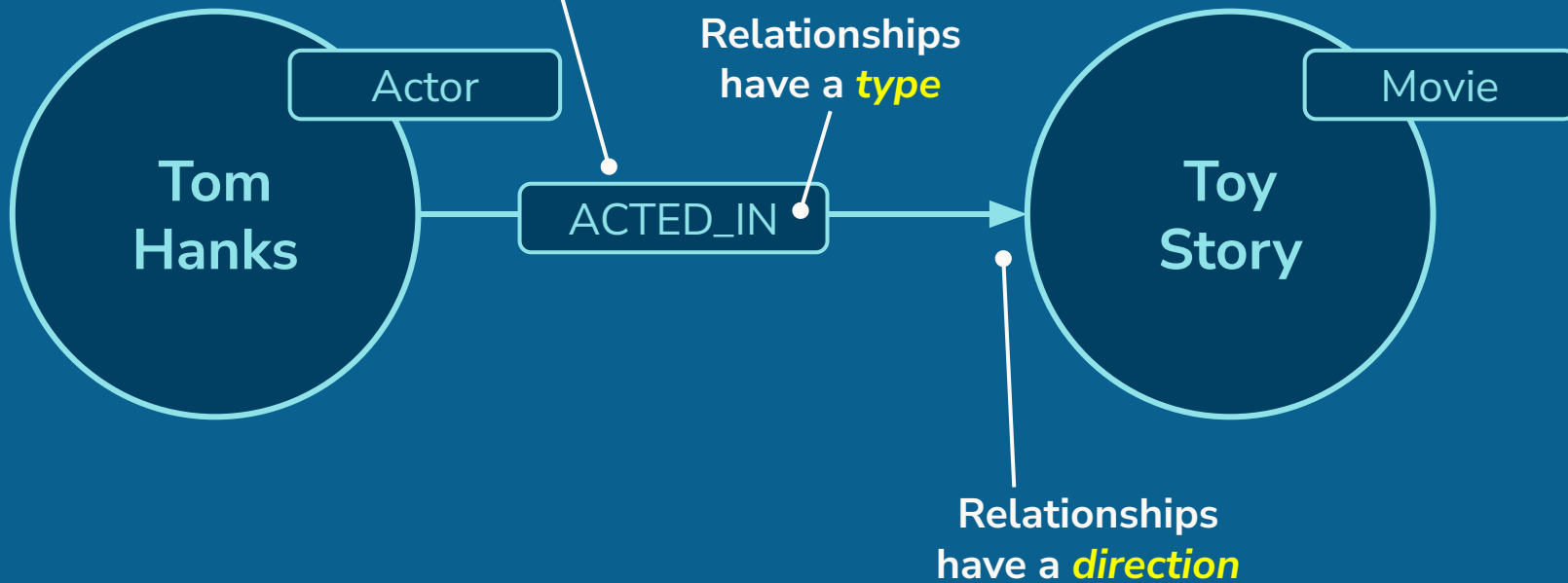


Relationships connect *two nodes*

Relationships
have a *type*

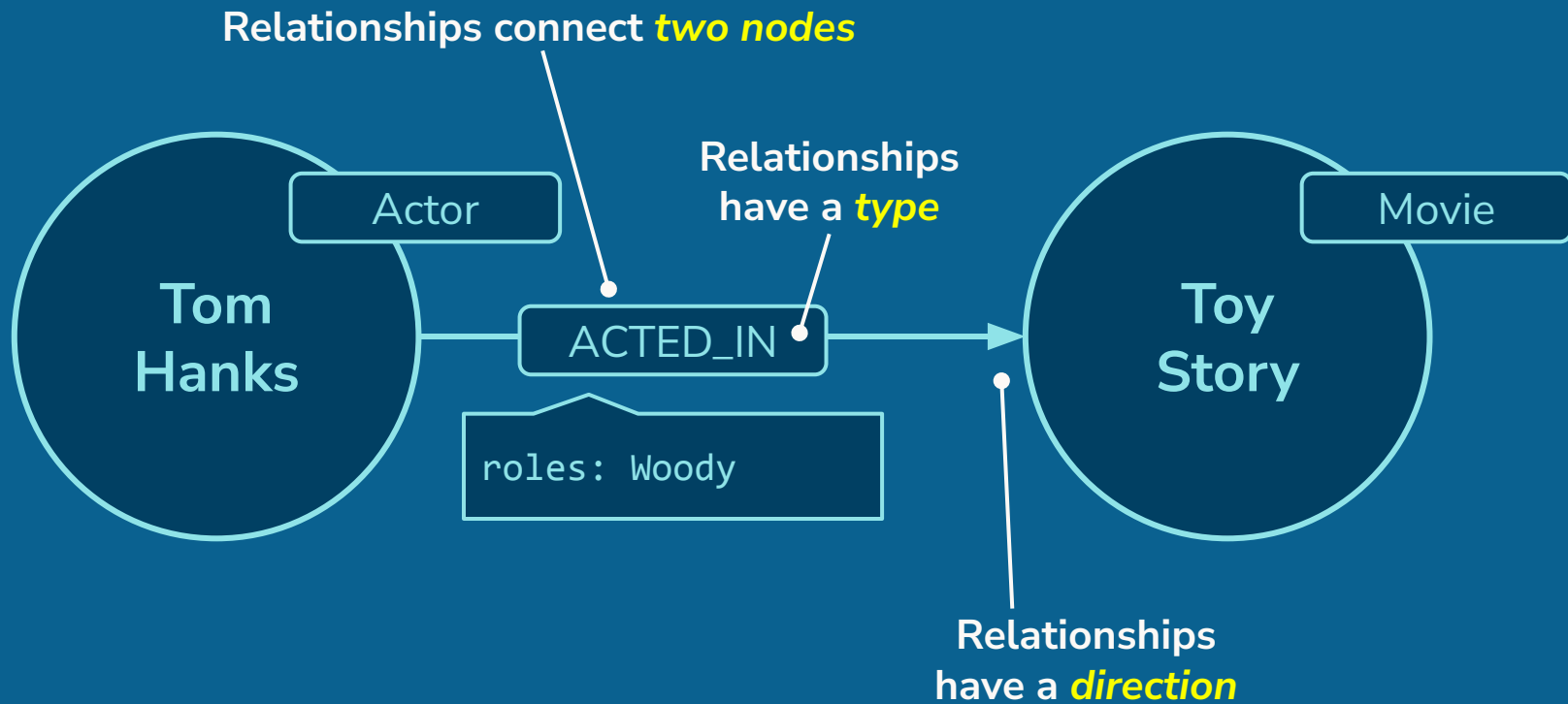


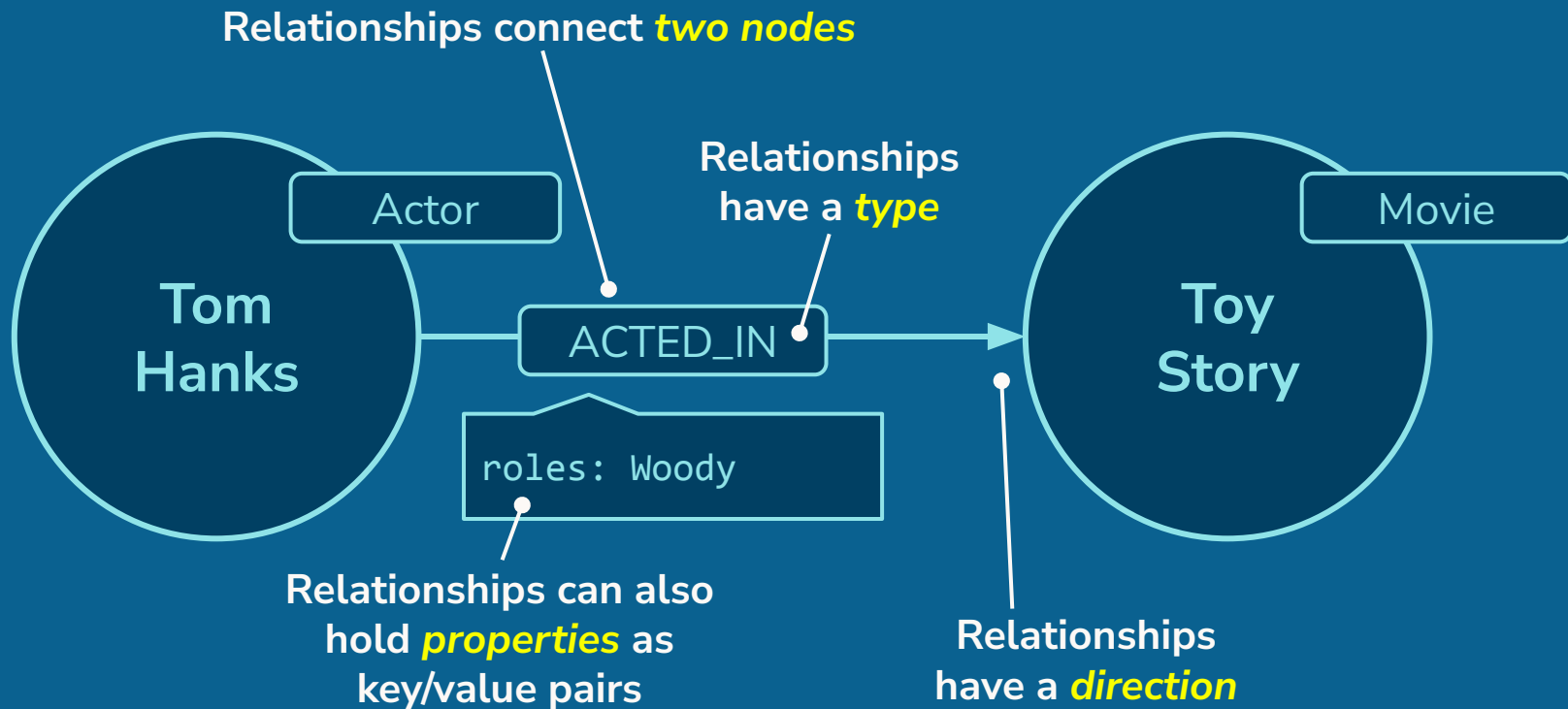
Relationships connect *two nodes*



Relationships have a *type*

Relationships have a *direction*







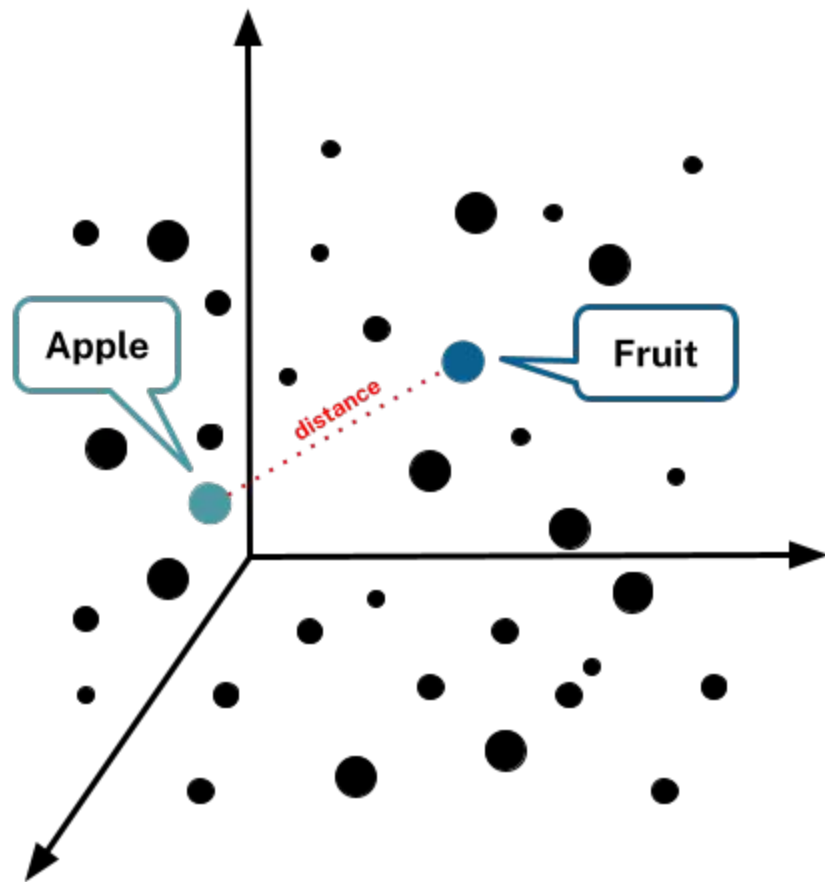
What are **Vectors**?





“apple”

0.0077788467, -0.02306925, -0.007360777, -0.027743412, -0.0045747845, 0.01289164, -0.021863015, -0.008587573, 0.01892967, -0.029854324, -0.0027962727, 0.020108491, -0.004530236, 0.009129008, -0.021451797, 0.002030382, 0.030813828, 9.744976e-05, 0.0019172973, -0.02568733, -0.020985752, -0.008066699, 0.02134214, -0.01222684, 0.0009980568, 0.005105939, 0.009999417, -0.000107408916, 0.015845545, -0.012980737, 0.020574536, -0.016160812, -0.018518453, 0.005263572, -0.019286057, -0.009293495, -0.012096621, -0.008854863, -0.005753605, -0.006157968, 0.010540851, 0.007724018, -0.0065554776, 0.00052944134, -0.023453051, 0.011089141, -0.021671113, -0.00061425474, -0.012754567, 0.015489157, -0.0054520466, -0.0020355221, -0.015050527, -0.0052944133, -0.0028082666, 0.0027431573, -0.019450543, 0.0063807103, -0.010725899, 0.0049243183, 0.005266999, 0.01513277, -0.027921606, 0.0055754115, -0.009183837, 0.00380718, -0.013624975, -0.0084710615, 0.012905347, 0.015667351, 0.033363372, 0.013268588, 0.014036193, 0.0063464423, 0.004454846, 0.0014820931, -0.03396649, -0.0062779062, -0.00314238, 0.01818948, 0.0075389706, -0.02637269, 0.009574492, 0.024974553, 0.024823774, 0.009882905, -0.021657405, 0.010109074, -0.007970748, 0.0028887964, 0.011849891, 0.0054726074, 0.0078336755, 0.016448664, -0.026975807, 0.016599443, -0.012713445, 0.026345275, 0.004667308, -0.03736588, 0.0009834929, 0.006089432, -0.028730331, -0.011198798, -0.020396343, 0.0019738395, 0.012459862, -0.003738644, 0.015448036, -0.019902883, 0.0064389664, 0.00926608, 0.021945259, -0.051648803, -0.016448664, -0.01744929, 0.0091099103, 0.0021713076, 0.022705105, 0.025556525, 0.021021318, 0.025892028, 0.028107627,



Vector index

Find movies by plot using Cypher

Show how to query a vector index using Python

Show how to create embeddings using Python

```
neo4j$ MATCH (m:Movie {title: "Toy Story"}) RETURN m
```



Graph



Table



Text



Code



Node properties



Movie

movieId	1	
plot	A cowboy doll is profoundly threatened and jealous when a new spaceman figure supplants him as top toy in a boy's room.	
plotEmbedding	<p>$[-0.026989128440618515, -0.024155009537935257, 0.006058253347873688, -0.024324016645550728, -0.022516941651701927, -0.0050864629447460175, -0.013442561961710...$</p> Show all	
poster	https://image.tmdb.org/t/p/w440_and_h660_face/uXDfjjbdP4ijW5hWSBrPrIKpxab.jpg	
released	1995-11-22	
revenue	373554033	
runtime	84	




```

1 MATCH (m:Movie {title: 'Toy Story'})
2
3 CALL db.index.vector.queryNodes('moviePlots', 6, m.plotEmbedding)
4 YIELD node, score
5
6 RETURN node.title AS title, score, node.plot AS plot

```



Table



Text



Code

	title	score	plot
1	"Toy Story"	1.0	"A cowboy doll is profoundly threatened and jealous when a new spaceman figure supplants him as top
2	"Little Rascals, The"	0.9214558601379395	"Alfalfa is wooing Darla and his He-Man-Woman-Hating friends attempt to sabotage the relationship."
3	"NeverEnding Story III, The"	0.920598566532135	"A young boy must restore order when a group of bullies steal the magical book that acts as a portal bet
4	"Drop Dead Fred"	0.9200617074966431	"A young woman finds her already unstable life rocked by the presence of a rambunctious imaginary frie
5	"E.T. the Extra-Terrestrial"	0.9191048741340637	"A troubled child summons the courage to help a friendly alien escape Earth and return to his home-wor
6	"Gumby: The Movie"	0.9181168079376221	"In this offshoot of the 1950s claymation cartoon series, the crazy Blockheads threaten to ruin Gumby's

Started streaming 6 records after 8 ms and completed after 11 ms.

```

1 WITH genai.vector.encode(
2   "A mysterious spaceship lands Earth",
3   "OpenAI",
4   { token: $token }) AS myMoviePlot
5 CALL db.index.vector.queryNodes('moviePlots', 6, myMoviePlot)
6 YIELD node, score
7 RETURN node.title, node.plot, score

```

Table

Text

Code

	node.title	node.plot
1	"2001: A Space Odyssey"	"Humanity finds a mysterious, obviously artificial object buried beneath the Lunar surface and, with the intelligent computer H.A.L. 9
2	"Alien"	"The commercial vessel Nostromo receives a distress call from an unexplored planet. After searching for survivors, the crew heads I
3	"Arrival, The"	"Zane, an astronomer, discovers intelligent alien life. But the aliens are keeping a deadly secret, and will do anything to stop Zane fr
4	"Coneheads"	"Aliens with conical crania crash land on Earth."
5	"Aliens"	"The planet from Alien (1979) has been colonized, but contact is lost. This time, the rescue team has impressive firepower, but will it
6	"E.T. the Extra-Terrestrial"	"A troubled child summons the courage to help a friendly alien escape Earth and return to his home-world."

Started streaming 6 records after 10 ms and completed after 849 ms.

Embedding non-text data

(extension)

It doesn't just have to be text, you can embed other data such as images

```

1 MATCH (m:Movie{title: 'Babe'})
2
3 CALL db.index.vector.queryNodes('moviePosters', 6, m.posterEmbedding)
4 YIELD node, score
5
6 RETURN node.title, score, node.poster

```



Table

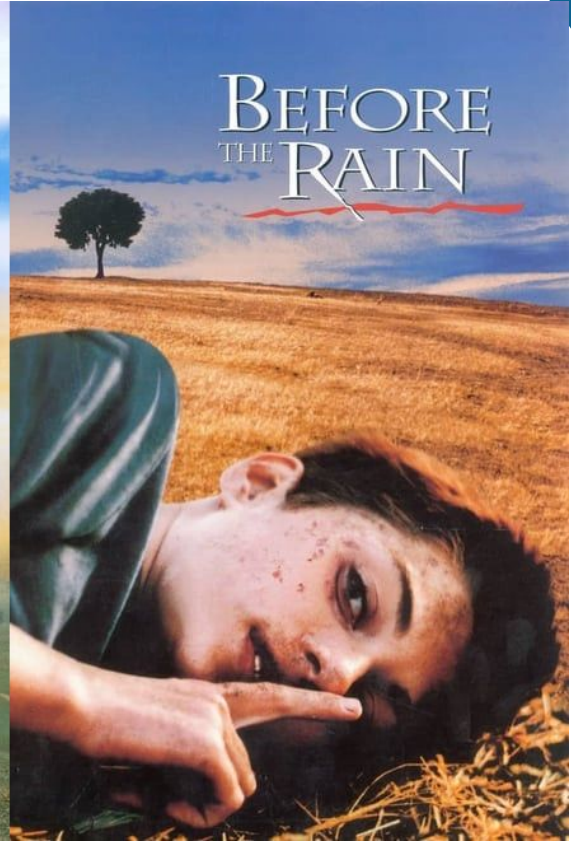
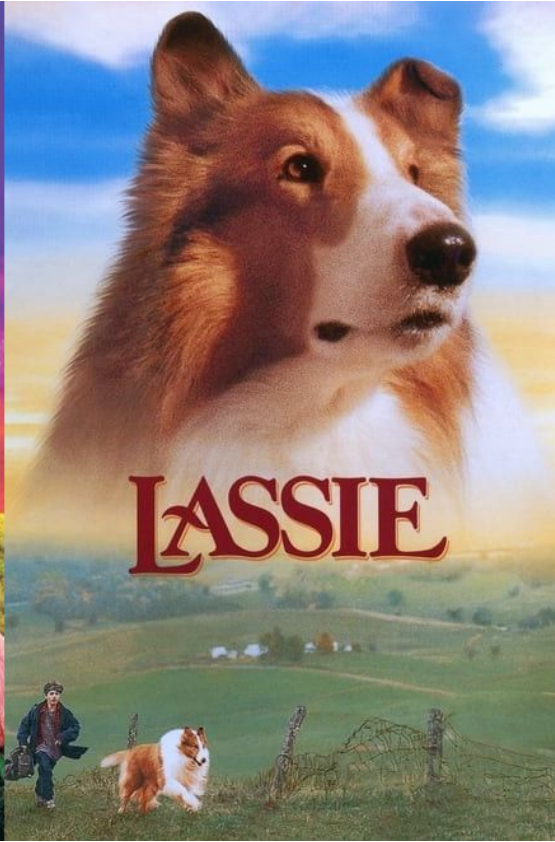
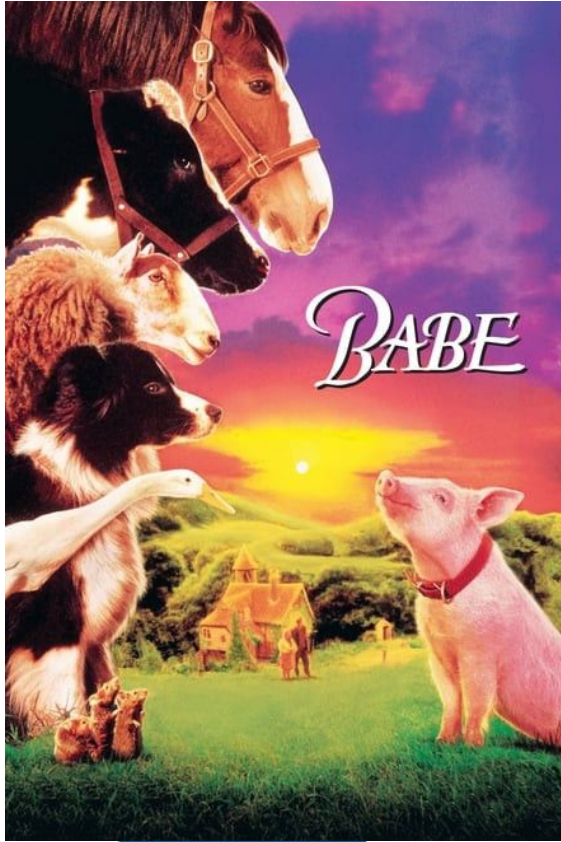


Text



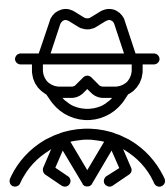
Code

	node.title	score	node.poster
1	"Babe"	1.0	"https://image.tmdb.org/t/p/w440_and_h660_face/bjDlp0b5TQRx4kF8i9TKeUANLtp.jpg"
2	"Lassie"	0.873661994934082	"https://image.tmdb.org/t/p/w440_and_h660_face/5Tz2rzFivQKmMBz5T8Cj7M7CdnI.jpg"
3	"Before the Rain (Pred dozhdot)"	0.8410471081733704	"https://image.tmdb.org/t/p/w440_and_h660_face/rSdjBP0UfJsaju4XcV64oTCQ2ZL.jpg"
4	"Balto"	0.8397032618522644	"https://image.tmdb.org/t/p/w440_and_h660_face/tpoaKJMRjOZI9cbZRR2aiKikyRC.jpg"
5	"Amazing Panda Adventure, The"	0.8371400833129883	"https://image.tmdb.org/t/p/w440_and_h660_face/tIK40vCE9i8g4VhyJ3LLr5y1ROJ.jpg"
6	"Bad Girls"	0.832358181476593	"https://image.tmdb.org/t/p/w440_and_h660_face/m2689gml09K6y92Ffb5lGVQlqRu.jpg"

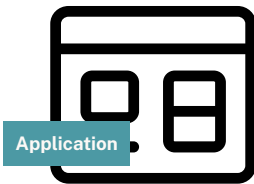




Vector + Graph = GraphRAG



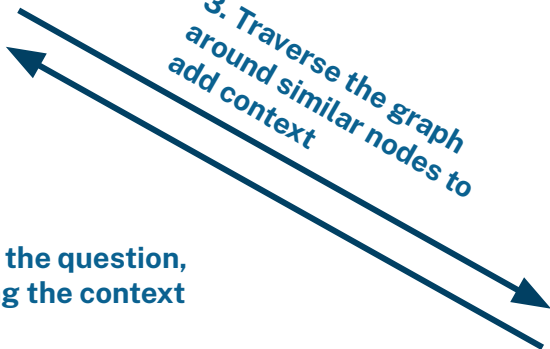
1. User asks a question



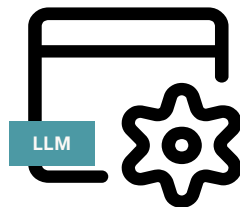
2. Use the question to identify similar nodes



3. Traverse the graph around similar nodes to add context



4. Ask the question, passing the context



```

6 YIELD node, score
7 RETURN
8 node.title + "-" + node.plot AS text,
9 score,
10 {
11   title: node.title,
12   directors: [ (person)-[:DIRECTED]→(node) | person.name ],
13   actors: [ (person)-[r:ACTED_IN]→(node) | [person.name, r.role] ],
14   genres: [ (node)-[:IN_GENRE]→(genre) | genre.name ],
15   tmdbId: node.tmdbId,
16   source: 'https://www.themoviedb.org/movie/' + node.tmdbId
17 } AS metadata

```

Table

Text

Code

score

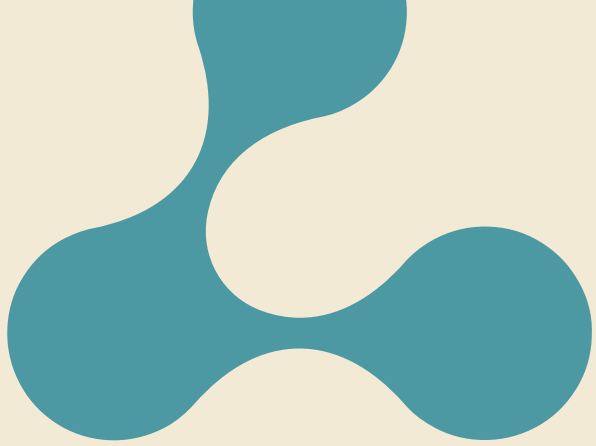
metadata

```

"https://www.themoviedb.org/movie/62",
"actors": [
  [
    "Keir Dullea",
    "Dr. David Bowman"
  ],
  [
    "Gary Lockwood",
    "Dr. Frank Poole"
  ],
  [
    "William Sylvester",
    "Dr. Heywood Floyd"
  ],
  [
    "Daniel Richter",

```


Agents, retrievers, and tools



Agents



```

llm = ChatOpenAI(
    openai_api_key=os.getenv('OPENAI_API_KEY'),
    model="gpt-4")

chat_prompt = ChatPromptTemplate.from_messages(
    [
        ("system", "You are a movie expert providing information about movies."),
        ("human", "{input}"),
    ]
)

movie_chat = chat_prompt | llm | StrOutputParser()

tools = [
    Tool.from_function(
        name="General Chat",
        description="For general movie chat not covered by other tools",
        func=movie_chat.invoke,
    ),
]

agent_prompt = hub.pull("hwchase17/react")
agent = create_react_agent(llm, tools, agent_prompt)
agent_executor = AgentExecutor(
    agent=agent,
    tools=tools,
    verbose=True
)

question = input("> ")
response = agent_executor.invoke({"input": question})
print(response["output"])

```



Retrievers



```

llm = ChatOpenAI(
    openai_api_key=os.getenv('OPENAI_API_KEY'),
    model="gpt-4"
)

embeddings = OpenAIEmbeddings(openai_api_key=os.getenv('OPENAI_API_KEY'))

graph = Neo4jGraph(
    url=os.getenv("NEO4J_URI"),
    username=os.getenv("NEO4J_USERNAME"),
    password=os.getenv("NEO4J_PASSWORD"),
)

neo4jvector = Neo4jVector.from_existing_index(
    embeddings,
    graph=graph,
    index_name="moviePlots",
    node_label="Movie",
    text_node_property="plot",
    embedding_node_property="plotEmbedding",
    retrieval_query=""
)

RETURN
node.title + " - " + node.plot AS text,
score,
{
    title: node.title,
    tmdbId: node.tmdbId,
    source: 'https://www.themoviedb.org/movie/' + node.tmdbId
} AS metadata
"""
)

retriever = neo4jvector.as_retriever()

instructions = (
    "Use the given context to answer the question."
    "If you don't know the answer, say you don't know."
    "Context: {context}"
)

```



Tools



```
from plot_retriever import get_movie_plot

tools = [
    Tool.from_function(
        name="General Chat",
        description="For general movie chat not covered by other tools",
        func=movie_chat.invoke,
    ),
    Tool.from_function(
        name="Movie Plot Search",
        description="For when you need to find information about movies based on a plot",
        func=get_movie_plot,
    )
]
```

```
    retrieval_query=""
RETURN
    node.title + " - " + node.plot AS text,
    score,
    {
        title: node.title,
        directors: [ (person)-[:DIRECTED]->(node) | person.name ],
        actors: [ (person)-[r:ACTED_IN]->(node) | [person.name, r.role] ],
        genres: [ (node)-[:IN_GENRE]->(genre) | genre.name ],
        tmdbId: node.tmdbId,
        source: 'https://www.themoviedb.org/movie/' + node.tmdbId
    } AS metadata
"""
```


The background is a solid blue color with several large, overlapping, wavy shapes in a lighter shade of blue, creating a layered, organic effect.

Thoughts, comments, questions?

The code!

github.com/martinothanolon/pydata-ams-24

LLM graph builder

Learn more

graphacademy.neo4j.com





Workshop

dev.neo4j.com/genai-workshop

Thank you ❤️



@martinohanlon

Unstructured data

Look at a typical challenge of dealing with unstructured data.

How can we make sense of this data.

Review GraphAcademy course data

Chunking

Build a graph

Build a graph of the course content

Load the documents

Chunk and create vectors

Use metadata to create the graph

Use graph in the chatbot

Updated version of the chatbot that uses the graph

Add topics to the graph

Modify the retriever to pull back graph elements data