Neo4j-enhanced Machine Learning

Michael Schwarz
schwarz.michael@posteo.net

Abstract

With the growth of data, the importance of the relationships in the data are getting more important. Today, businesses recognize the predictive power of relationships, the ability to use network structures to improve their machine learning (=ML) models and their own need to innovate. This results in common use cases such as fraud prevention and targeted recommendations. Graph databases such as Neo4j are characterised by the management of complex data relationships and are increasingly being integrated with AI and ML.

In my research I investigate graph-enhanced ML, which is also referred to as "Graph Data Science" (=GDS). Building on this understanding, I would like to highlight ML pipelines with Neo4j. For research and writing I use ChatGPT as a tool beside many others. To prove proper use of ChatGPT, I disclose my complete chat history, which can be found in the linked Zenodo repository (Schwarz, 2023).

According to the results, Neo4j GDS technology enhances machine learning by leveraging knowledge graphs, graph algorithms, and graph native ML techniques to transform complex relationships and structures in data into predictive models and insights. Therefore, Neo4j offers a comprehensive GDS platform that integrates transactional, analytical and visualization capabilities for both developers and business users. Especially Neo4j's ML pipeline, particularly for link prediction, provides a comprehensive workflow for training models to predict new links in a graph, integrating feature generation and model training with many configuration options. For example, eBay enhanced its user experience by developing a chat bot called ShopBot using Neo4j's graph technology and AI, to create a more contextually aware search and recommendation system that adapts to user interactions and preferences.

1 The Relation between Neo4j and Machine Learning

Graph databases focus on relationships – how data is related. Twitter, for example, makes use of a graph database connecting 313 million monthly active users. The Neo4j database is a "native" graph storage and graph processing engine which is heavily optimized on relationships. A non-native graph storage uses a relational, columnar or some other general-purpose data store rather than being specifically engineered for the uniqueness of graph data. As the depth of relationships increase, high performance is only possible with a fully native graph database.(Sasaki et al., 2018)

Neo4j is the leader in graph technology and helps enterprise companies like eBay, NASA and Volvo to reveal and predict how people, processes and systems are interrelated. Neo4j tackles connected data challenges like fraud detection, recommendation engines, entity resolution, supply chain optimization, logistic route optimization, retail suggestion engines and social network monitoring.(Sasaki et al., 2018)

Between 2010 and 2021 the number of AI research paper that feature graph technology has increased over 700 percent; see figure 1. This burst is due to the increasing connectedness of data, breakthroughs in scaling graph technology to enterprise-sized problems, excellent results when integrated with machine learning (=ML) and artificial intelligence (=AI) solutions.(Frame, 2021)

Graph data science (=GDS) is a science-driven approach to gain knowledge from the relationships and structures in data, typically to power predictions. GDS can typically be broken down into three areas (Frame, 2021):

1. Graph statistics
2. Graph analytics
3. Graph-enhanced ML and AI

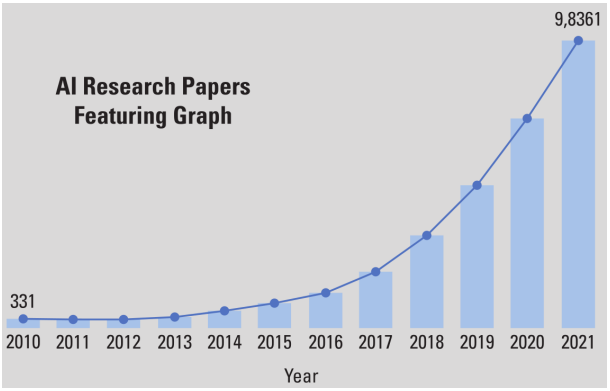

Figure 1: The rise of graph data science.

Graph statistics and analytics are often used in conjunction to answer certain types of questions about complex systems, and the subsequent insights are then applied to train ML and AI models. At the most abstract level, these questions fall into a few broad areas: movement, influence, groups and interactions, and patterns as shown in figure 2:

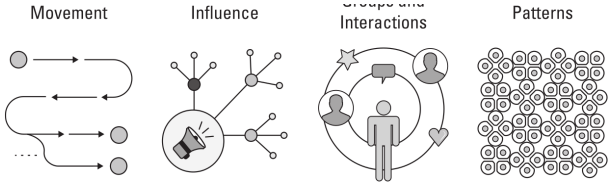

Figure 2: Graph data science questions fall into four different areas.

The areas in figure 2 answer the following questions:

1. How do things travel through a network?
2. What are the most influential points?
3. What are the groups and interactions?
4. What patterns are significant?

2 Neo4j Platform

Neo4j is a company that provides a graph data science software platform, which supports transactional, analytical processing as well as visualization. It also includes graph storage, data management and analytics tooling, integrations via API and the Cypher query language. The technology is available on-premise, self-hosted and fully manages in the cloud.[1] In the following, I explain the four main products of the platform, which are part of the neo4j ecosystem shown in figure 3:

[1] https://neo4j.com/docs/getting-started/

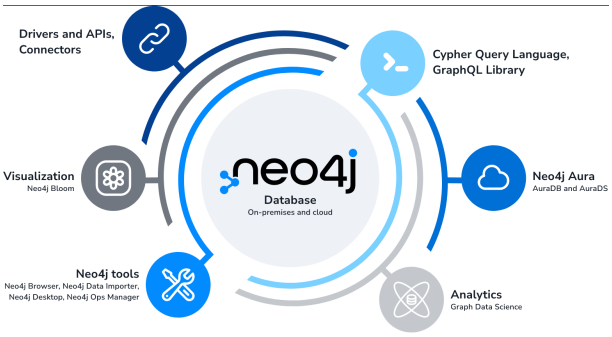

Figure 3: An overview of the Neo4j ecosystem

- *Neo4j Graph Database*[2] is the core product, a native graph database that is built to store and retrieve connected data. Accordingly, the graph query language Cypher can be used to interact with the database.

- *Neo4j Graph Data Science*[3] is a data analysis platform that unifies the ML and graph database into a single workspace. It helps to get insights from big data in order to analyse relationships, improve predictions and discover insights. GDS supports three different types of ML models:
  - *Node Classification* models are used to predict the classes of unlabeled nodes as a node properties based on other node properties, e.g. detect a fraud.
  - *Node Regression* models are used to predict the value of node property based on other node properties, e.g. determine the price of a house.
  - *Link Prediction* models are use to predict which relationships should exist between nodes, e.g. recommend a product for a user.

- *Neo4j Desktop/Browser*[4] is a user interface to support developers to query, visualize, administer and monitor their databases.

- *Neo4j Bloom*[5] is a graph visualization and exploration tool for business users that does not require any code or programming skills to view and analyse data.

[2] https://neo4j.com/product/neo4j-graph-database/
[3] https://neo4j.com/product/graph-data-science/
[4] https://neo4j.com/product/developer-tools/
[5] https://neo4j.com/product/bloom

In the following, I will take a closer look at graph-based ML via Neo4j.

3 Graph Data Science Technology

Figure 4 shows the major phases of a typical graph data science journey.

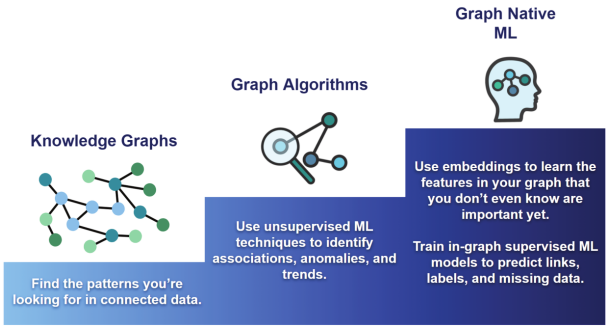

Figure 4: The graph data science journey.

3.1 Knowledge Graphs

Knowledge graphs (=KG) are the foundation of GDS. At a high level, KG are interlinked sets of data points and describe entities and their relationships. They are often implemented to bring divers information together to help to explore the connections in the data and/or add context to applications such as AI systems to help to find a better and faster decision. Graph queries are used to get this kind of information, when it is clear what to look for, like "How many relationships does person A have?". In this stage, queries are more likely to be straight forward and the results comprehensible, because they focus on small areas around a few nodes. Put simply, the Neo4j database itself can be regarded as a KG in simple scenarios.(Frame and Blumfeld, 2022)

3.2 Graph Algorithms

In most scenarios, users start using graph algorithms to understand their networks even better by looking at the entire graph to find clusters, identifying influential nodes, evaluating different pathways. The most common graph algorithms fall into five categories:(Frame and Blumfeld, 2022)

- Pathfinding and search
- Centrality (importance)
- Community Detection
- Similarity
- Heuristic Link Prediction

Graph algorithms can be used for graph analysis (=GA) or graph feature engineering (=GFE) when included in a prediction model. GA is asking questions about graph topology that can be directly answered with an unsupervised algorithm; for example "What is the shortest route between two nodes" or "How is the data clustered". However, more advanced use cases leverage graph algorithms in predictive models.
GFE is the process of finding, combining and extracting predictive elements from raw graph data to be used in ML tasks. For example, calculating a PageRank (Centrality) of each node in the graph or labelling nodes based on the communities they belong to (Community Detection). These scores and labels can then be extracted to a list or table of numbers and identifiers, so called feature vectors, for training ML models. Therefore, the data is usually split into test and training datasets. Finally, the graph features and resulting ML metrics are often written back to the graph database for persistence and future use. Figure 5 shows how the use of graph features to enhance ML fits into a lager workflow.

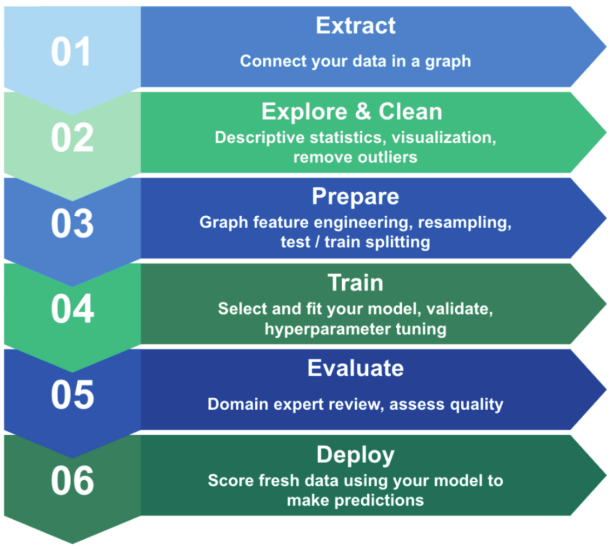

Figure 5: Graph feature engineering is part of a larger ML workflow.

3.3 Graph Native ML

Graph native ML (=GNML) is the most advanced type of GDA. In this case, the graph is used as the input for a ML model to make predictions about how the graph will evolve in the future. Unlike GFE, where the features are predefined,

GNML uses the entire graph as input. GNML represents a new approach to ML that may drastically improve results with less data and make predictions more explainable. Most commonly, the graph needs to be converted into a structure that is compatible with ML techniques, so called graph embeddings.(Frame and Blumfeld, 2022)

3.3.1 Embeddings

Embeddings are the technology used to translate connected data into a predictive signal. Therefore empeddings are simplified graphs or subsets of graphs which can be used as a feature vector or set of vectors that are in a lower dimensional form, such as a list of numbers. Figure 6 shows a visualization of a graph in two dimensions via techniques like PCA[6] and t-SNE[7] plots. There are three types of embeddings(Frame, 2021):

- *Node embeddings*, which describe connectivity of each node
- *Path embeddings*, which encompass the traversals across a graph
- *Graph embeddings*, which encode an entire graph into a single vector

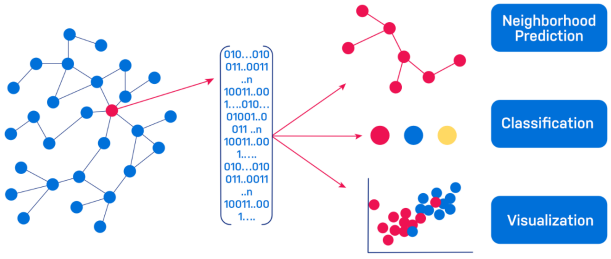

Figure 6: Graph embeddings create numerical representations of specific graph data.

Node and path embeddings are often used for more advanced feature engineering that incorporates more complex information. Instead of guessing which algorithm is going to tell something useful, graph embeddings highlight predictive patterns.

In the following I want to develop a rough understanding behind embeddings. There are three commonly used embedding technics: Node2Vec (1), FastRP (2) and GraphSAGE (3) (Frame, 2021):

1. Probably the most well-known graph embedding algorithm, *Node2Vec* uses random walks to sample a graph and a neural network to learn the best representation of each node. Node2Vec can capture both topological similarity (nodes that are neighbors) and structural similarity (nodes with similar roles, like bridge nodes)

2. *Fast Random Projection* (=FastRP) uses linear algebra to generate embeddings. It is faster than Node2Vec and can encode the structure of a graph alone, or incorporate node properties into embeddings

3. *GraphSAGE* uses a graph convolutional neural network (=GCNN) to encode both the topology of the graph and the properties of nodes. It can generate high-quality embeddings for new data based on prior training.

4 Neo4j ML Pipeline

4.1 Link Prediction

Neo4j link prediction[8] (=LP) pipelines is still in beta and must also not be confused with the graph algorithm "Heuristic Link Prediction", LP is a specific ML pipeline. Via Neo4j GDS it is possible to train a ML model to predict new links. Figure 7 gives a quick overview of a common ML pipeline workflow for training such a model in the context of link prediction (Brunenberg, 2022).

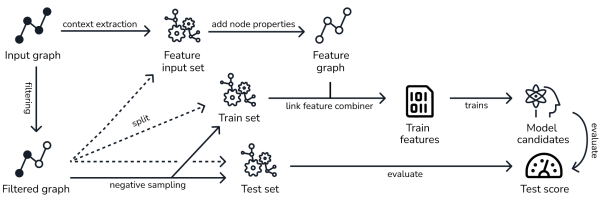

Figure 7: Pipeline for training the model.

1. *Create a new pipeline* in the Neo4j GDS workspace.

2. *Create an in-memory subgraph* from a general graph. A projection can be created by filtering node labels and relation types or even by custom cypher queries. An important note for working with link prediction pipelines is that they only work with undirected links.

[6] https://en.wikipedia.org/wiki/Principal_component_analysis

[7] https://en.wikipedia.org/wiki/T-distributed_stochastic_neighbor_embedding

[8] https://neo4j.com/docs/graph-data-science/current/machine-learning/linkprediction-pipelines/link-prediction

3. *Node features* are generated via executing GDS graph algorithms like node embeddings such as FastRP within the pipeline.

4. *Create link features*: This step is one of the main differences to conventional ML tasks, because of classifying a pair of nodes instead of a single data point. To create a link feature vector for each pair of nodes, both nodes are combined into a single vector as shown in figure 8.

5. *Data splitting*. The data gets split in training and testing data sets for training and testing the ML model. The main data set will remain the features, which are used for the link prediction afterwards. Also, these three data sets must be disjoint to each other to prevent biased results. Additionally, negative samples are needed for training and testing the model, which should be equally in size to the positive training and testing data set, like in figure 9.

6. *Model configuration & training*: Currently, there exist two classification approaches in the Neo4j GDS: Logistic Regression and Random Forests. Luckily, Neo4j supports to add multiple model candidates with various parameter spaces to the pipeline. During training, each model candidate is subject to a k-fold cross validation to find the best model configuration. Another important parameter for the training is the evaluation metric to use. The winning model will be stored in the model catalog for later usage.

7. *Prediction*: Finally, the trained model can be applied to a graph in the graph catalog to create a new relationship type containing the predicted links. The relationships also have a property which stores the predicted probability of the link, which can be seen as a relative measure of the models prediction confidence.

Neo4j offers a bunch of configuration options for fine tuning which I did not mention. I just gave a overview how basically on a high level graphs can be used in the Neo4j GDS workspaces to train ML models to predict new insights. Also Neo4j provides AuraDS, a fully managed version of Neo4j Graph Data Science.

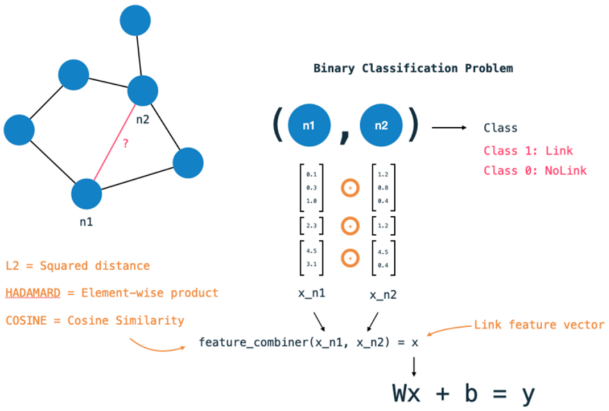

Figure 8: Link feature vector based on a pair of nodes.

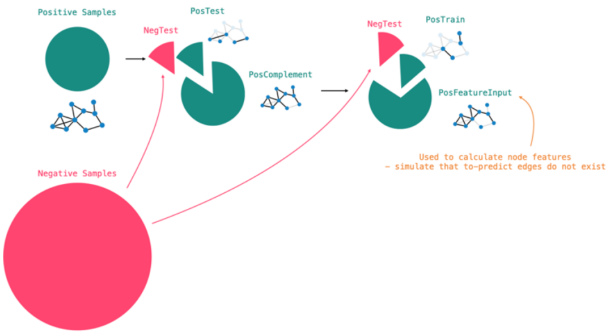

Figure 9: Split data for training, testing and calculating features

5 eBay Case Study

This case study[9][10] sheds light on the improvements to the user experience with the eBay platform.

5.1 The Challenge

The future of commerce is envisioned as a shift towards a distributed model, moving away from centralized platforms and meeting users on messaging and social platforms where they spend most of their time. This approach involves integrating a personal shopping assistant into these platforms, aiming to bring a vast selection of products to a wide audience. In line with this vision, eBay is addressing a specific challenge in online shopping: the need for more contextually aware search and recommendation systems.

5.2 The Solution

eBay developed a chat bot (=ShopBot) which can be integrated in popular messenger or social media platforms as a personal shopping assistant. One of

[9] https://neo4j.com/case-studies/ebay/
[10] https://medium.com/@rjpittman/ebay-shopbot-beta-under-the-hood-bedf69157a70

the fundamental technologies is Neo4j, which is responsible for holding and managing the probabilistic models that are essential for understanding conversations in shopping scenarios:

- The graph contains detailed information about eBay's extensive product catalogue and buyer interactions. This data is important to understand what customers are looking for and how they interact with the inventory.
- By coupling with AI and natural language processing technologies Neo4j helps the ShopBot interpret and respond to user queries more effectively.
- The ShopBot accumulates information and traverses through the graph to continuously check the inventory for the best match. This process allows for dynamic and real-time decision-making.
- The probabilistic models in Neo4j aid the ShopBot in learning from past interactions. This learning process is key to improving the accuracy and efficiency of the shopping assistant over time.

6 Conclusion

During my research I found out, that Neo4j is not only a graph database but rather a whole platform supporting users to manage complex data relationships and many other enterprise challenges, as well as enhance AI and machine learning applications by analysing and predicting connections in data.
Depending on the use case GDS technologies such as knowledge graphs, graph algorithms and graph native ML provide different approaches to gain insights from the data or even predict new connections or nodes in the graph in combination with a ML model. Beside prediction, feature engineering is a very important working step in ML. Graph databases make this process especially easier. This makes customer data accessible and can be used to train a model in a specific domain. Additionally, Neo4j has a very good performance when handling large amounts of data, especially, when the graph features are transformed into vectors resp. embeddings. In general, the GDS ML pipelines are a very good tool for setting up complex machine learning workflows directly in the Neo4j infrastructure. It reduces the overhead required for external tools and resources by enabling workflows within the interface of the framework. Moreover, the pipeline can also be automated to keep the input features and prediction model up to date.
Looking to the future, as businesses continue to accumulate vast amounts of data, the demand for real-time analytics will rise. Neo4j's ability to handle complex, interconnected data sets efficiently makes it a prime candidate for powering real-time, context-aware analytics in various industries. Furthermore, the convergence of graph databases and deep learning could give rise to more powerful graph neural networks. These networks could be able to leverage the structural information of graph data more effectively, leading to breakthroughs in pattern recognition and decision-making processes.

References

P. Brunenberg. 2022. Understanding neo4j gds link predictions.

A. Frame. 2021. Graph embeddings: Ai that learns from your data to solve your problems. *Neo4j*.

A. Frame and Z. Blumfeld. 2022. Graph data science for dummies, 2nd neo4j special edition. *John Wiley & Sons, Inc.*

B.M. Sasaki, J. Chao, and R. Howard. 2018. Graph databases for beginners. *Neo4j*.

M. Schwarz. 2023. Neo4j-enhanced machine learning, chatgpt log.

List of Figures