Graph Machine Learning for Everyone

Learning from relationships, not only from rows

Robin Brochier, Clement Wang

2026

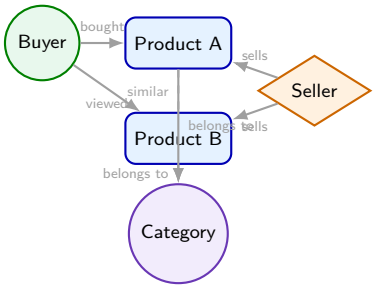
Most company data is connected

Traditional view: a table

Product	Price	Brand	Sales
A	49	X	120
B	89	Y	32
C	55	X	71
D	20	Z	12

Useful to store data, but relationships are hidden.

Graph view: entities and links

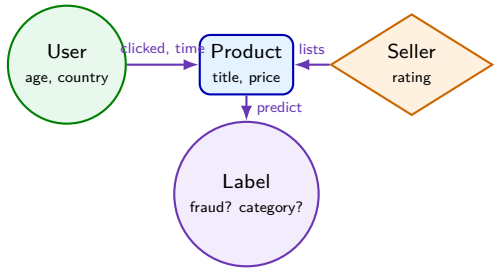


The graph makes relationships explicit.

What is a graph?

A graph contains:

- **Nodes**: things
- **Edges**: relationships
- **Features**: information about them
- **Labels**: what we want to predict



The same graph can support many business questions.

Graphs are everywhere

Graphs are everywhere

Web
pages link to pages

Social
people follow people

Commerce
buyers, sellers, products

Finance
accounts transfer money

Biology
proteins interact

Logistics
locations and routes

Knowledge
entities and facts

Graphs are everywhere

Web
pages link to pages

Social
people follow people

Commerce
buyers, sellers, products

Finance
accounts transfer money

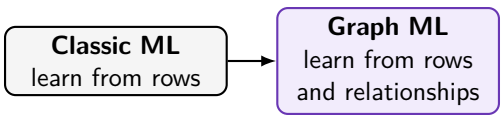
Biology
proteins interact

Logistics
locations and routes

Knowledge
entities and facts

If the important signal is in the connections, a graph representation may help.

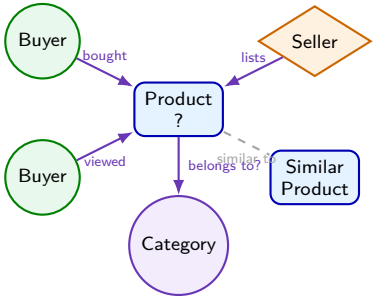
A useful mental model



Graph ML does not replace classic ML.
It adds the network context.

Graph ML turns connections into predictions

A node is understood partly by its neighborhood.

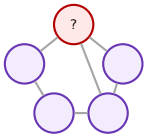


To understand a product, use its text and price, but also buyers, seller, category, similar products, orders, reviews, and behavior.

Four common graph prediction tasks

Four common graph prediction tasks

Node prediction

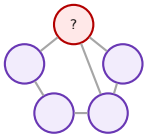


Is this seller risky?

What category is this product?

Four common graph prediction tasks

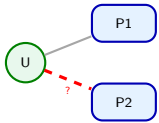
Node prediction



Is this seller risky?

What category is this product?

Edge prediction

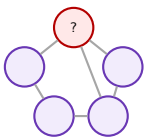


Will this user buy this product?

Are these items duplicates?

Four common graph prediction tasks

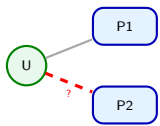
Node prediction



Is this seller risky?

What category is this product?

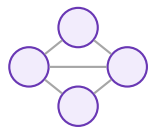
Edge prediction



Will this user buy this product?

Are these items duplicates?

Graph prediction



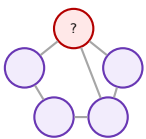
?

Is this transaction network suspicious?

Is this molecule toxic?

Four common graph prediction tasks

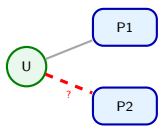
Node prediction



Is this seller risky?

What category is this product?

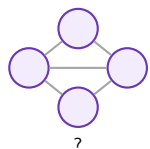
Edge prediction



Will this user buy this product?

Are these items duplicates?

Graph prediction

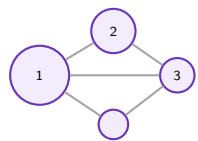


?

Is this transaction network suspicious?

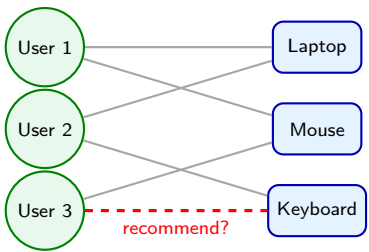
Is this molecule toxic?

Ranking



Which pages, products, or sellers matter most?

Example: recommendations as link prediction



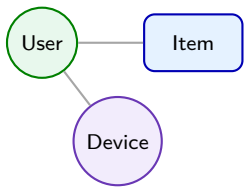
A recommendation can be seen as:

Which missing user-product link is likely to appear next?

This makes recommendations a natural graph ML problem.

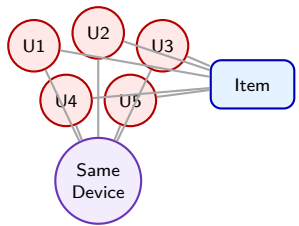
Example: fraud as suspicious structure

Individual view



Each account may look normal.

Network view



The group pattern is suspicious.

When is Graph ML useful?

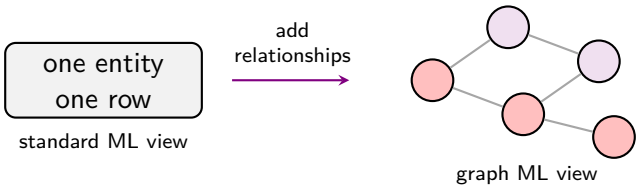
Does the network change the decision?

one entity
one row

standard ML view

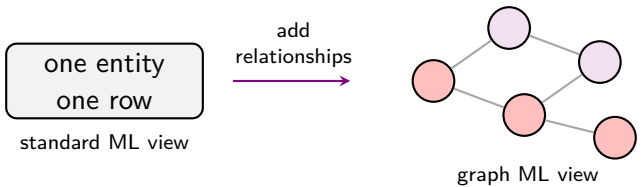
When is Graph ML useful?

Does the network change the decision?



When is Graph ML useful?

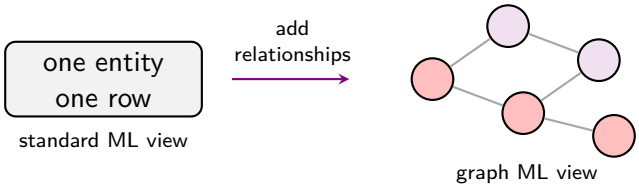
Does the network change the decision?



Good fit: neighbors reveal extra signal **Weak fit:** links are noisy or add nothing.

When is Graph ML useful?

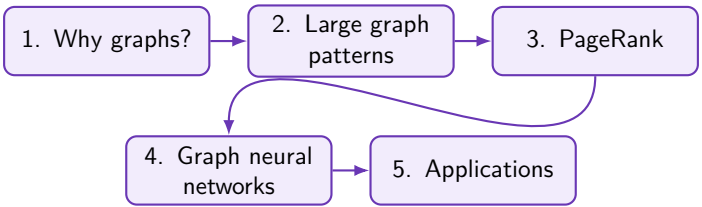
Does the network change the decision?



Good fit: neighbors reveal extra signal **Weak fit:** links are noisy or add nothing.

Rule of thumb: use Graph ML when the neighborhood changes the prediction.

Outline



We start with graph intuition, then move to algorithms, GNNs, and real applications.

Where are we?

Introduction: What are graphs? What can we do with graphs?

Large graph patterns

PageRank

GNNs and foundation models

Applications

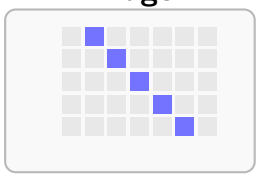
Company opportunities

Conclusion

Large graphs are not random-looking

Images, text, and graphs are not random noise.

Image



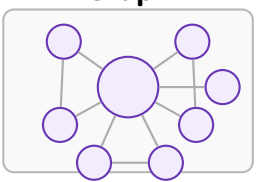
shapes, edges, textures

Text

This is an example of a sentence.

words, grammar, meaning

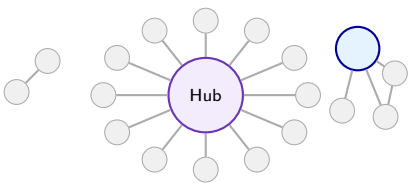
Graph



hubs, communities, short paths

Real graphs have structure. Let us look at it.

Power law: most nodes are small, a few are huge



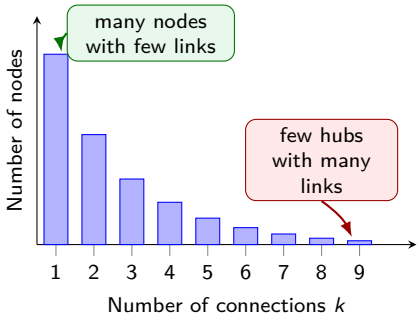
A few nodes collect many connections.

In many real graphs:

- many nodes have few links;
- a few nodes have many links;
- these hubs shape the network.

Examples: web pages, social accounts, products, sellers, search queries.

Heavy-tailed distributions and power laws



Heavy-tailed pattern

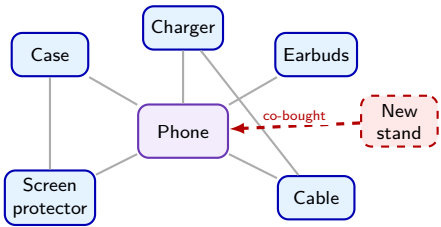
Most nodes have few links, while a few nodes become highly connected hubs.

$$\mathbb{P}(\text{degree} = k) \propto k^{-\gamma}$$

- k: number of connections, or degree.
- γ: controls how fast the curve decreases.

As k grows, probability decreases, but hubs do not disappear.

Why do hubs appear? Preferential attachment



Popular products get more chances to appear in future baskets.

Preferential attachment:

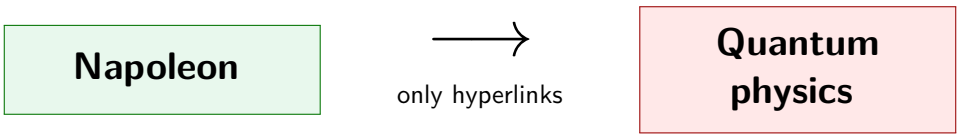
Popular products become even more connected.

In a co-bought graph, a product that already appears in many baskets has more opportunities to be linked with new products.

This is the "rich get richer" effect.

Small-world property: the Wikipedia Game

Can we go from one Wikipedia page to another using only hyperlinks?



How many clicks?

No search bar. Only links inside pages.

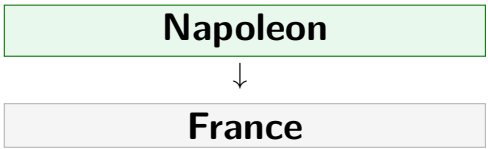
One possible solution

A path can be much shorter than expected.

Napoleon

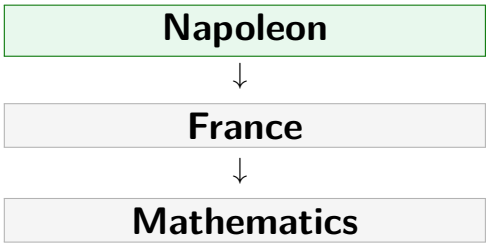
One possible solution

A path can be much shorter than expected.



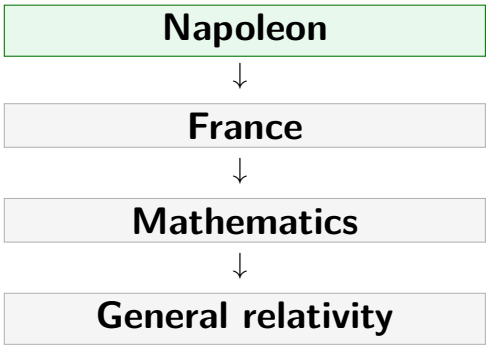
One possible solution

A path can be much shorter than expected.



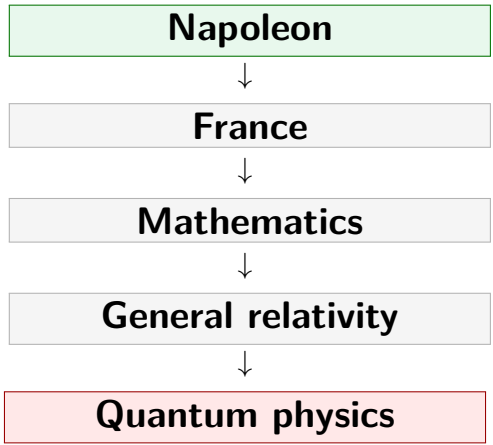
One possible solution

A path can be much shorter than expected.



One possible solution

A path can be much shorter than expected.



Small-world property

Large network, short paths.

Wikipedia intuition

4

clicks in our example

Facebook, 2016

3.57

degrees of separation on average

A huge graph can still feel surprisingly small.

Where are we?

Introduction: What are graphs? What can we do with graphs?

Large graph patterns

PageRank

GNNs and foundation models

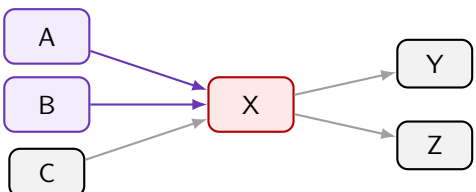
Applications

Company opportunities

Conclusion

PageRank in one sentence

A page is important if important pages link to it.



X receives more authority when important pages point to it.

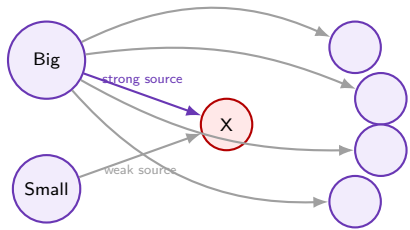
Why not just count links?

Counting links treats all links equally.

But in reality:

- a link from a trusted page matters more;
- a page with many outgoing links spreads its attention;

$\Longrightarrow$ **Authority is recursive.**



PageRank is "reputation flowing through links".

Random surfer intuition

Imagine a person browsing the Web forever.

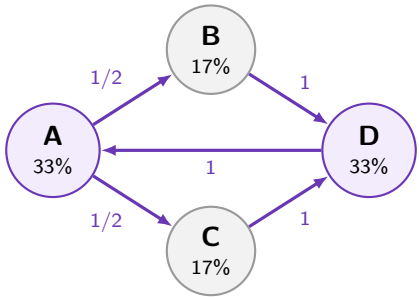
At every page, they either:

1. follow a link on the page;
2. randomly jump somewhere else.

> PageRank is the long-term probability of being on each page.

More visits means higher importance.

PageRank as Probability Flow



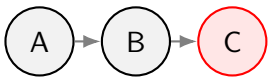
Probability mass moves along outgoing links.
Nodes with more incoming mass (A and D) become larger in the steady state.

Computing authority is an iterative diffusion process.

Does It Converge ?

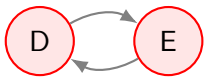
Without teleportation, the probability flow can get trapped.

1. The Dead End



Probability flows into node C
but has nowhere to go.

2. The Infinite Loop



Probability gets stuck bouncing
forever between D and E.

Solution: The Damping Factor

Add a small probability (e.g., 15%) to **randomly jump** to any page instead of following a link.
This acts as an escape hatch, guaranteeing the algorithm eventually stabilizes.

The Random Surfer

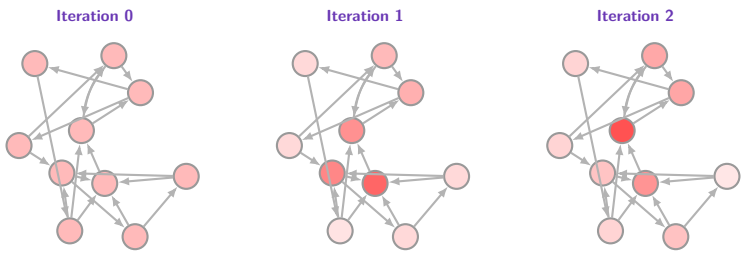
```
current_page = random_page()

loop N times:
    visits[current_page] += 1

    if random() < 0.85 and has_links(current_page):
        current_page = random_outgoing_link(current_page)
    else:
        current_page = random_page() // Teleport!

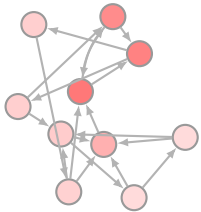
pagerank_scores = normalize(visits)
```

PageRank Authority Flow - Part 1

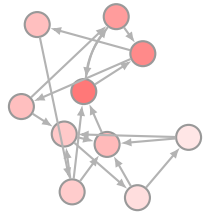


PageRank Authority Flow - Part 2

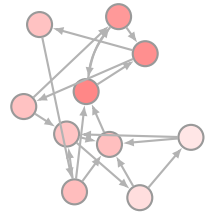
Iteration 3



Iteration 4

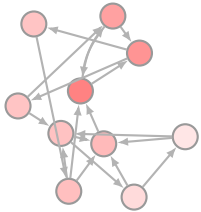


Iteration 5

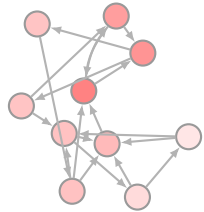


PageRank Authority Flow - Part 3

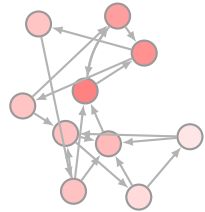
Iteration 6



Iteration 7



Iteration 8 (Converged)



PageRank beyond web search

Search
rank pages or products

Marketplace
rank important sellers

Catalog
rank authoritative brands

Fraud
propagate risk scores

Recommendations
rank nearby relevant items

PageRank is an early example of using graph structure as a business signal.

Personalized PageRank

Classic PageRank asks:

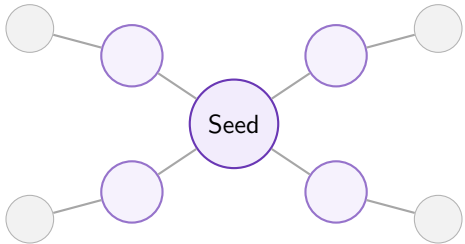
What is globally important?

Personalized PageRank asks:

What is important relative to this user, topic, or seed?

This makes PageRank useful for recommendations and graph-based retrieval.

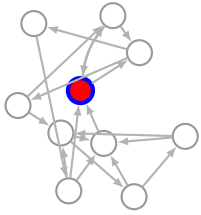
Personalized PageRank: local importance



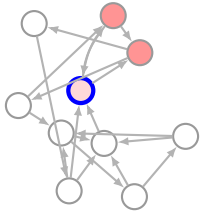
Scores concentrate around the seed node or seed group.

Personalized PageRank Flow - Part 1

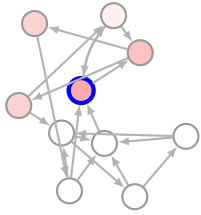
Iteration 0



Iteration 1

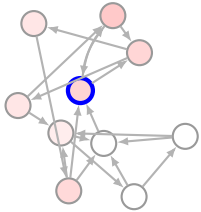


Iteration 2

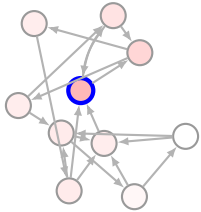


Personalized PageRank Flow - Part 2

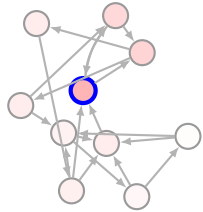
Iteration 3



Iteration 4

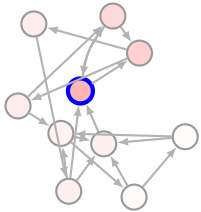


Iteration 5

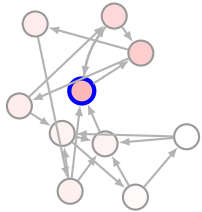


Personalized PageRank Flow - Part 3

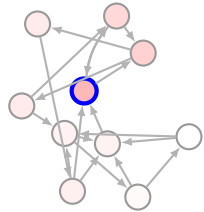
Iteration 6



Iteration 7



Iteration 8 (Converged)



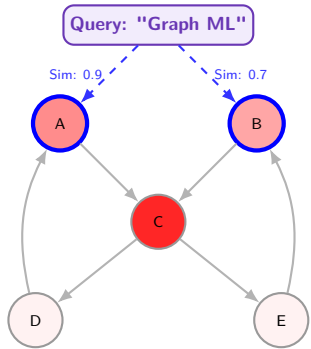
Example: Web Search

1. We initialize the authority scores by semantic similarity

$\Longrightarrow$ What pages are related to my query ?

2. We propagate the semantic scores using the PPR

$\Longrightarrow$ What pages are most influential ?



- - - **Seed Injection** (Semantic Match)
— **Authority Flow** (Web Links)

C ranks highest because it receives graph authority directly from highly relevant pages.

From PageRank to Graph Embeddings

PageRank: Global diffusion for *node authority*.
Graph Embedding: Local diffusion for *pairwise similarity*.

The Goal: Map discrete nodes to continuous dense vectors $z_u \in \mathbb{R}^d$.

- We want vector dot products to approximate graph topology:

$$z_u^\top z_v \approx S_G(u, v)$$

- $S_G(u, v)$ is a similarity metric (e.g., probability of reaching v from u via random walks).

The Scalability Solution (DeepWalk, 2014)

Problem: Exact matrix factorization is $O(N^3)$. Unscalable.

Solution: Borrow **Word2Vec** from NLP. Treat the graph topology as a corpus of text:

- **Word** $\Longleftrightarrow$ Node
- **Sentence** $\Longleftrightarrow$ Random Walk

Simulate diffusion via random walks $\rightarrow$ Optimize with SGD in $O(N)$.

How it Works: 3 Steps

1. **Random Walks:** Simulate paths of length L from each node to capture local neighborhoods.
2. **Context Windows:** Slide a window over the walk sequence to create `(target, context)` training pairs.
3. **Skip-Gram Optimization:** Train an embedding lookup table to predict context nodes given a target node.

Result: Nodes sharing structural roles or local neighborhoods converge in the $\mathbb{R}^d$ latent space.

Limitations of Shallow Embeddings

Random-walk embeddings solved scalability, but introduced architectural limits:

- **Feature-Agnostic:** Ignores node attributes (e.g., text, price, user age). Relies 100% on edges.
- **Homogeneous:** Treats all node/edge types the same. Fails to capture rich schemas.
- **Transductive:** Cannot compute vectors for unseen nodes (cold-start problem). Requires full retraining.

$\Rightarrow$ **Next step: Graph Neural Networks (GNNs)** (Inductive, Feature-aware).

Where are we?

Introduction: What are graphs? What can we do with graphs?

Large graph patterns

PageRank

GNNs and foundation models

Applications

Company opportunities

Conclusion

Why Graph Neural Networks?

Modern machine learning works best when the model matches the data structure.

Data	Shape	Model family
Images	grid	CNNs
Text	sequence	Transformers
Graphs	nodes and edges	GNNs

A GNN is a neural network designed for connected data.

What makes graphs different?

In a graph, examples are not isolated.

Classic ML

one row, one prediction
one image, one prediction
one text, one prediction

$\neq$

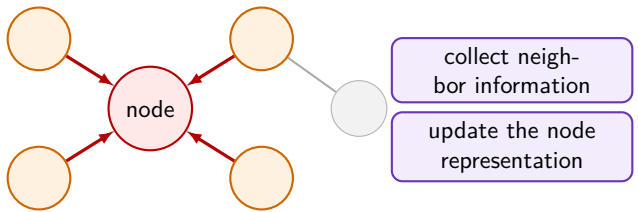
Graph ML

a product depends on buyers
buyers depend on products
sellers connect many products

The context of an entity is part of the data.

GNN in one sentence

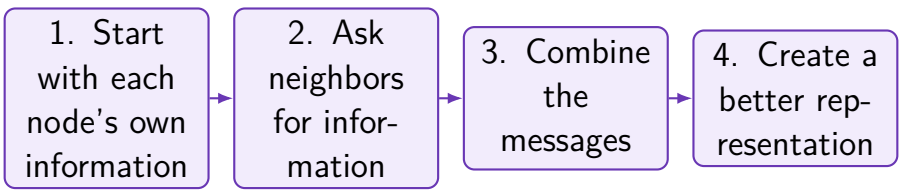
Each node learns from its neighbors.



This mechanism is called **message passing**.

Message passing without formulas

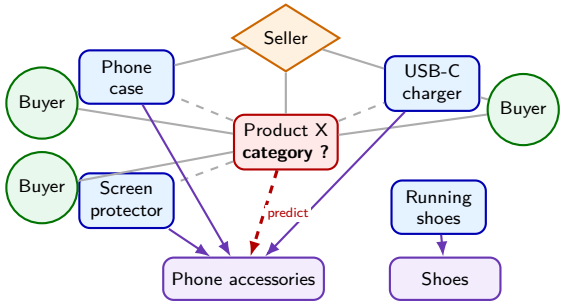
A GNN layer is like a neighborhood conversation.



Repeat this a few times, and each node can use information from a wider neighborhood.

Example: classifying a product from its neighborhood

We know the category of some products. Can we infer the category of Product X?



Nearby evidence:

- same buyers;
- same seller;
- similar products;
- nearby known categories.

Graph intuition:
Product X is surrounded by phone accessories.

Why scaling GNNs is different

In classic ML, examples are easy to batch. In graphs, examples are connected.

Tabular / NLP / Vision

process a batch of independent examples

Example: 1,000 rows, images, or sentences can be processed independently.

GNNs

each node may need its neighbors' information

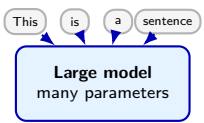
Example: predicting one product may require buyers, sellers, similar products, and categories.

So GNN inference often needs a large part of the graph context.

The practical compromise

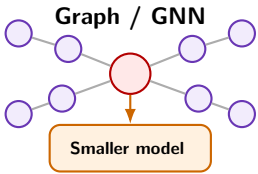
Text models are often heavy because of the **model**.
GNNs are often heavy because of the **graph context**.

Text / Transformer



bounded input,
expensive model

Graph / GNN

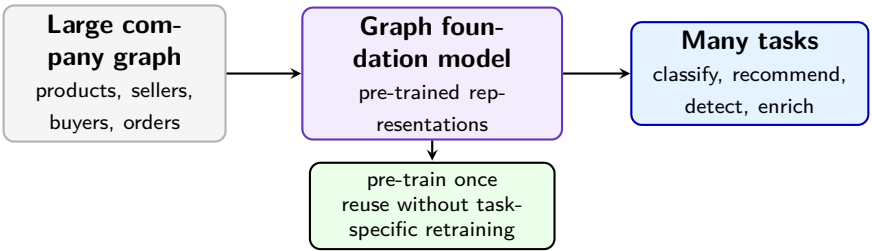


large context, smaller model

Graph foundation models

Can we pre-train once on a large graph, then reuse it for many tasks?

example: Kumo AI



Promise: one reusable model for many relational tasks.
Reality: promising frontier, still less mature than LLMs.

What is still hard?

Research and industry are working on the same big bottlenecks.

Scale
huge graphs, memory, compute

Data quality
missing, noisy, biased edges

Few labels
limited ground truth

Dynamic graphs
graphs change constantly

Explainability
why this prediction?

Graphs + LLMs
combine text and relations

Where are we?

Introduction: What are graphs? What can we do with graphs?

Large graph patterns

PageRank

GNNs and foundation models

Applications

Company opportunities

Conclusion

Application map

Search and information retrieval
PageRank
and authority

Recommendations
user-item graphs

Fraud detection
rings and
shared signals

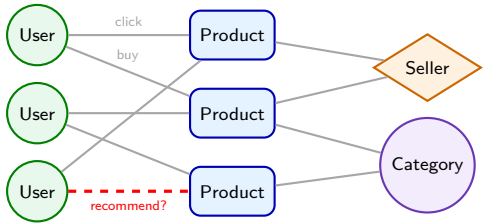
Biology
molecules
and proteins

Maps
roads and traffic

Knowledge graphs
facts and entities

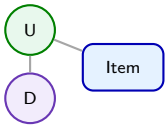
Many more...
Sports analytics, finance, new materials discovery, fluid dynamics, smart grid energy, ...

Recommender systems

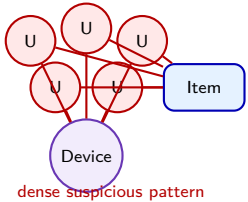


Recommendation often means predicting the next useful edge in a user-product graph.

Fraud detection



normal pattern



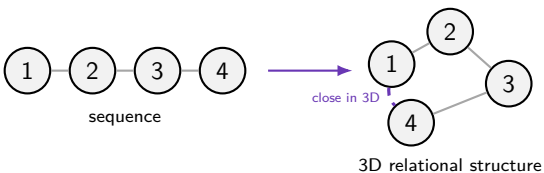
dense suspicious pattern

Fraud is often relational.

The suspicious signal may not be visible in one account alone.

It appears through shared devices, addresses, payment methods, reviews, sellers, or transaction loops.

AlphaFold and biology

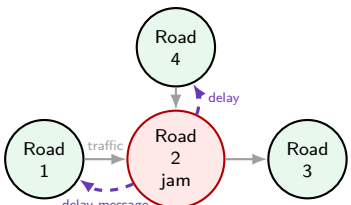


Biology is full of relationships:

- atoms and bonds;
- amino acids and distances;
- proteins and interactions;
- molecules and properties.

Structure often determines function.

Maps and logistics

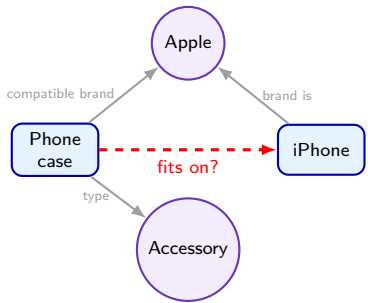


Road networks are graphs.

Traffic on one road segment influences nearby segments.

Graph models can help predict delays, estimated arrival times, and congestion propagation.

Knowledge graphs



Knowledge graphs organize facts. Graph ML can infer missing facts or validate existing ones.

Where are we?

Introduction: What are graphs? What can we do with graphs?

Large graph patterns

PageRank

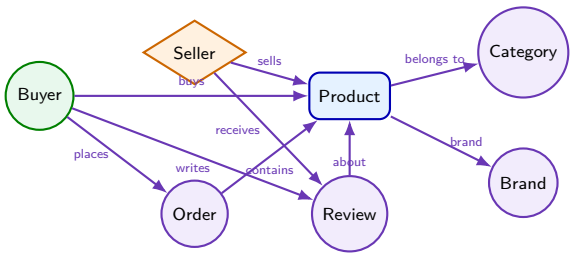
GNNs and foundation models

Applications

Company opportunities

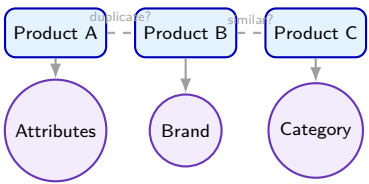
Conclusion

A marketplace is naturally a graph



The value is not only in entities,
but in how they are connected.

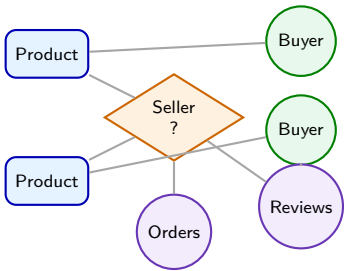
Catalog intelligence



Graph ML can help with:

- product categorization;
- duplicate detection;
- attribute enrichment;
- brand normalization;
- compatibility links;
- catalog quality checks.

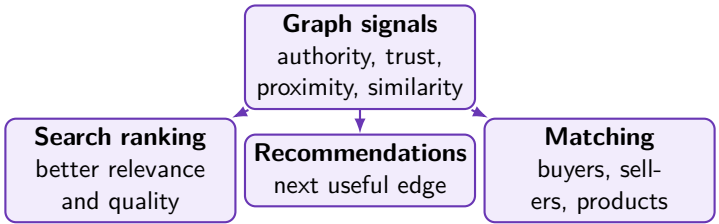
Seller quality and risk



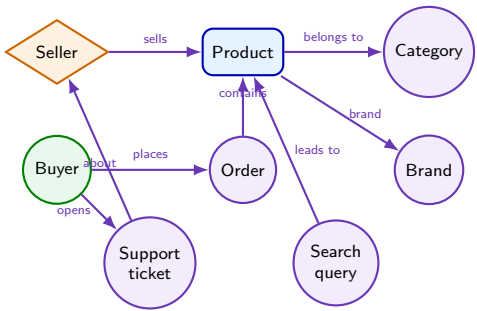
Predict or explain:

- seller risk;
- seller churn;
- growth potential;
- poor fulfillment patterns;
- suspicious connections;
- similarity to top sellers.

Search, ranking, and recommendations



Company knowledge graph



A knowledge graph can support analytics, copilots, search, risk, catalog, and decision support.

How to identify a good internal use case

Ask four questions:

Question	**Why it matters**
What are the nodes?	The entities must be clearly defined.
What are the edges?	Relationships must be meaningful and reliable.
What is the prediction?	The business decision must be concrete.
What is the label?	We need a way to train or evaluate the model.

Where are we?

Introduction: What are graphs? What can we do with graphs?

Large graph patterns

PageRank

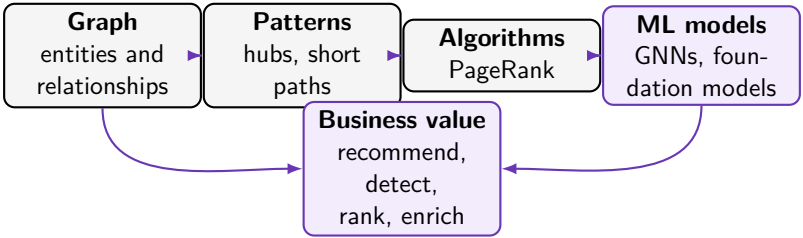
GNNs and foundation models

Applications

Company opportunities

Conclusion

The big picture



Graphs provide structure. Machine learning turns that structure into useful predictions.

Suggested further learning

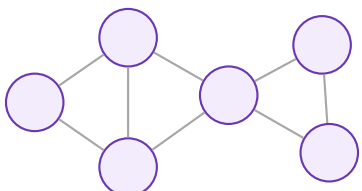
If you want to go deeper:

- Stanford CS224W: Machine Learning with Graphs.
- William Hamilton, *Graph Representation Learning*.
- Petar Veličković, *Everything is Connected: Graph Neural Networks*.
- Practical tutorials from PyTorch Geometric and DGL.

For non-specialists, focus first on use cases, graph construction, and evaluation before diving into model details.

Thank you

Questions?



Where do we already have graph-shaped data?