Graph Data Science for Author Domain Classification and Link Prediction

Basil Ali Khan, Hamza Ansari, Hayyan Khan

Group 6
Graph Data Science, Spring 2025

I. Introduction

The source code used in this project is available on GitHub at the following link: GitHub Repository.

II. Data Cleaning and Preprocessing

The raw dataset, sourced from [1], comprised several CSV files: `authors.csv`, `journal.csv`, `paper.csv`, `topic.csv`, `paper_journal.csv`, `paper_topic.csv`, and `paper_reference.csv`. To ensure data quality and consistency, we performed extensive cleaning and preprocessing using R scripts.

For `paper.csv`, missing values in `paper_citation_count` were imputed with 0, and rows lacking essential fields such as `paper_id`, `paper_year`, or `journal_date` were removed. In `journal.csv`, missing `journal_publisher` entries were set to "Unknown", and duplicates in `journal_name` were consolidated by merging publisher information. Fuzzy matching was employed to standardize publisher names, addressing variations like "Routledge" and "Routledge Journals". This method didn't turn out to be very fruitful so we ended up standardizing using the first significant names of publishers to ensure consistency across `paper_journal.csv`.

For `topic.csv`, duplicates based on `topic_id` and `topic_name` were eliminated, and topic names were standardized to title case. In relationship files (`paper_reference.csv`, `paper_topic.csv`, `paper_journal.csv`), entries with invalid references (e.g., IDs not present in corresponding node files) were discarded to maintain referential integrity. The R scripts for data cleaning are available in the GitHub repository under `gdscleaning.R` and `gdscleaning2.R`.

Post-cleaning, the dataset was refined, ensuring only high-quality data was utilized for graph construction in Neo4j. The number of rows before and after cleaning for each file is summarized in Table I.

TABLE I
Row Counts Before and After Data Cleaning

File	Rows Before	Rows After
`authors.csv`	[38926]	[38926]
`author_paper.csv`	[56541]	[56541]
`journal.csv`	[166]	[166]
`paper.csv`	[693753]	[69362]
`topic.csv`	[6490]	[6471]
`paper_journal.csv`	[32648]	[32648]
`paper_topic.csv`	[41881]	[41723]
`paper_reference.csv`	[1132045]	[1048576]

III. Graph Construction

Using the cleaned dataset, we constructed a heterogeneous graph in Neo4j to represent relationships between authors, papers, topics, journals, and publishers in the academic domain. The graph model was designed to facilitate both node classification and link prediction tasks.

A. Nodes

- **Author**: Represents researchers. Properties include:
  - `id`: Unique identifier.
  - `name`: Author name.
  - `total_citations`: Sum of citations for authored papers.
  - `research_domain`: Most frequent research topic.

  - `last_collab_year`: Most recent collaboration year.
- **Paper**: Represents academic papers. Properties include:
  - `id`: Unique identifier.
  - `title`: Paper title.
  - `year`: Publication year.
  - `citationCount`: Citation count.
- **Topic**: Represents research topics. Properties include:
  - `id`: Unique identifier.
  - `name`: Topic name.
- **Journal**: Represents journals. Properties include:
  - `name`: Journal name.
- **Publisher**: Represents journal publishers. Properties include:
  - `name`: Publisher name.

B. Relationships

- **WROTE**: Connects `Author` to `Paper`.
- **CO_AUTHORED**: Connects two `Author` nodes who collaborated. Includes:
  - `collaboration_year`: Year of collaboration.
- **HAS_TOPIC**: Connects `Paper` to `Topic`.
- **PUBLISHED_IN**: Connects `Paper` to `Journal`.
- **PUBLISHED_BY**: Connects `Journal` to `Publisher`.
- **CITES**: Connects `Paper` to another cited paper.

C. Feature Engineering

Additional features were computed to enhance the graph model:

- **Total Citations**: Sum of citations for an author's papers.
- **Research Domain**: Most frequent topic researched by an author.
- **Last Collaboration Year**: Most recent year of collaboration.

D. Neo4j Script for Graph Creation

The Neo4j script for graph construction is available in the GitHub repository under `graph_model_creation_script.txt`.

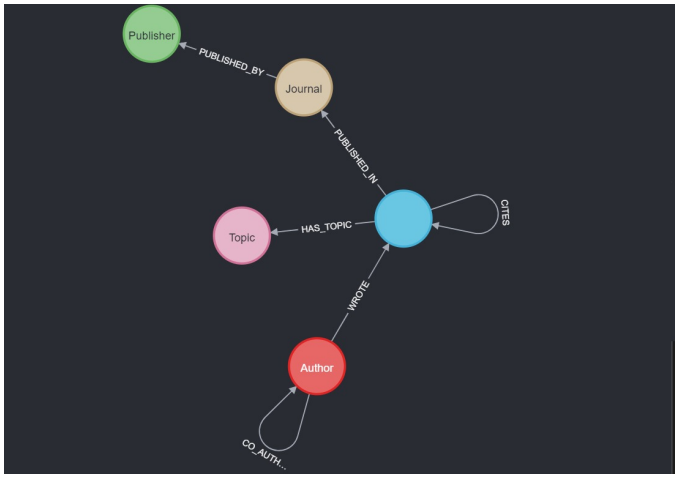

Fig. 1. Data Model for the Graph

IV. Methodology

A. Node Classification

The node classification task aimed to predict the research domain (`domain_id`) of authors based on their co-authorship patterns and other features. We utilized the Neo4j Graph Data Science (GDS) library to train a machine learning model to classify authors into specific research domains. The authors were represented as nodes in a co-authorship graph, where edges between nodes (authors) represented collaborations (co-authorships) in academic papers.

1) Graph Projection: The first step in our methodology was to project the graph as follows:

- **Nodes:**
  - `Author`: Represents an academic author. Each author node has a property `domain_id` (the target classification label).
- **Relationships:**
  - `CO_AUTHOR`: Represents a collaboration between two authors on a paper.
- **Label:** Authors with known research domains were tagged with the label `AuthorWithDomain` for the classification task. Authors without a known `domain_id` were excluded, leaving 12,738 authors (approximately 32.7% of the total 38,925).

2) Feature Engineering: To predict the `domain_id`, we used the following features:

- **Louvain Community ID**: Generated using the Louvain algorithm to detect communities based on co-authorship patterns, useful for domain classification.
- **Graph Embedding**: Generated using Fast Random Projection (FastRP) to transform the graph structure into a vector space, capturing local and global network structure. Embeddings of dimensions 64 and 128 were tested.
- **Domain ID**: The target property for classification.

3) Machine Learning Model: We used a Random Forest classifier, chosen for its robustness to overfitting and effectiveness with high-dimensional data. The training process included:

- **Random Forest Classifier**: Configured with 25 or 40 trees in different iterations.
- **5-Fold Cross-Validation**: To ensure generalization.
- **Evaluation Metrics**: F1-Weighted, Accuracy, Out-of-Bag (OOB) Error.

4) Model Training and Evaluation: The model was trained using a pipeline that included feature extraction (Louvain, FastRP), model training, and cross-validation. Predictions were made on unseen nodes to assess performance.

B. Link Prediction

The link prediction task aimed to forecast future collaborations between authors based on their historical co-authorship patterns and associated features. We leveraged the Neo4j GDS library's link prediction pipeline, integrating graph algorithms and machine learning to predict potential links.

1) Graph Projection: We projected a graph with:

- **Nodes**: `Author` nodes with properties `paper_count` and `citation_count`.
- **Relationships**: `CO_AUTHORED` relationships, representing past collaborations, treated as undirected.

2) Feature Engineering: To capture graph structure, we experimented with embedding techniques:

- **Initial Approach**: A heuristic-based method was developed to compute similarity scores for author pairs. This involved identifying influential authors using PageRank, selecting potential collaborators based on domain and publication criteria, and calculating scores based on normalized paper and citation counts, domain similarity, and citation relationships. This approach provided a baseline for comparison with machine learning models. This score was not used in the final model but could serve as a feature in future pipelines.
- **FastRP Embeddings**: Used in the second iteration to generate 64-dimensional embeddings efficiently.
- **Node2Vec Embeddings**: Used in the final model, generating 64-dimensional embeddings via biased random walks (walk length 80, 10 walks per node) to capture local and global structures.

Features were extracted using the Hadamard product of node embeddings.

3) Machine Learning Model: We trained a logistic regression model to classify potential links:

- **Logistic Regression**: Chosen for its simplicity and interpretability.
- **Train-Test Split**: 80/20 split with negative sampling (ratio 1.0) to balance the dataset.
- **Evaluation Metrics**: Area Under the Precision-Recall Curve (AUPRC).

4) Model Training and Evaluation: The pipeline included:

- **Pipeline Construction**: Steps for embedding generation, feature extraction, and model training.
- **Training**: Conducted on the projected graph, ensuring unique author pairs (author1.id ¡ author2.id) to avoid self-loops.
- **Evaluation**: Assessed using AUPRC. The fastRP model had a higher AUPRC but produced probabilities clustered around 0.5, indicating poor discrimination. The node2vec model, despite a slightly lower AUPRC, provided varied probability estimates, suggesting better calibration.

The node2vec model was selected, and predictions were made for non-collaborating author pairs, ranked by probability to identify likely future collaborations.

V. Results

A. Node Classification Results

The first node classification model achieved:

- **Test Accuracy**: 0.60
- **Test F1 Score**: 0.54
- **OOB Error**: 0.49

Fig. 2. Output after training pipeline

These initial metrics reflect the model's performance on a held-out validation set during training.

1) Prediction Performance: To further validate the model, we applied it to unseen nodes using the 'predict.stream' procedure in GDS and compared the predicted domain labels with the ground truth labels. The following evaluation metrics were calculated based on the predictions:

- **Accuracy:** 0.77
- **Precision:** 0.77
- **Recall:** 1.00
- **F1 Score:** 0.87

total	correct	accuracy
12738	9858	0.7739048516250588

Fig. 3. Accuracy

precision	recall	f1_score
0.7739048516250588	1.0	0.8725438130642591

Fig. 4. Precision Recall and F1 Score

The high recall value indicates that the classifier successfully predicted nearly all actual instances of each domain class. However, the slightly lower precision suggests that there were some false positives, indicating the need for improved feature representation or model tuning.

2) Iteration 2 – Embedding = 128, Number of Trees = 25:

- **Test Accuracy:** 0.59
- **Test F1 Score:** 0.53
- **Out-of-Bag (OOB) Error:** 0.49

testAccuracy	testF1	testOutOfBagError
0.59392988	0.5305171211445211	0.4965231045311799

Fig. 5. Output after training pipeline with embedding = 128 and trees = 25

- **Accuracy:** 0.78
- **Precision:** 0.78
- **Recall:** 1.00
- **F1 Score:** 0.88

total	correct	accuracy
12738	9932	0.7797142408541372

Fig. 6. Accuracy for embedding = 128 and trees = 25

precision	recall	f1_score
0.7801067671533993	1.0	0.8764718853362734

Fig. 7. Precision Recall and F1 Score for embedding = 128 and trees = 25

The change in embedding does not show any significant change in the model output.

3) Iteration 3 – Embedding = 64, Number of Trees = 40:

- **Test Accuracy:** 0.63
- **Test F1 Score:** 0.56
- **Out-of-Bag (OOB) Error:** 0.44

testAccuracy	testF1	testOutOfBagError
0.62663527	0.5592646420575051	0.44010767160161507

Fig. 8. Output after training pipeline with embedding = 64 and trees = 40

- **Accuracy:** 0.79

- **Precision:** 0.80
- **Recall:** 1.00
- **F1 Score:** 0.88

total	correct	accuracy
12738	10125	0.7948657560056523

Fig. 9. Accuracy for embedding = 64 and trees = 40

precision	recall	f1_score
0.7969853980216675	1.0	0.8870249017038008

Fig. 10. Precision Recall and F1 Score for embedding = 64 and trees = 40

The increase in the number of trees for the random forest model slows a slight betterment in the results of accuracy and precision.

B. Link Prediction Results

The link prediction task evaluated three approaches to predict future author collaborations: a heuristic method and two machine learning iterations using FastRP and Node2Vec embeddings. We also tried using Fastrp instead of logistic regression for training, the code for which is avaliable on the repository under `LinkPredictionUsingRandomForest.pdf` but discarded it midway due to the model taking taking too much time to train. Each iteration built upon the previous, addressing limitations and improving predictive performance.

1) Heuristic Approach: The heuristic approach employed a rule-based method to identify potential collaborations by computing a similarity score for author pairs without prior co-authorship. The methodology involved:

- **Identifying Influential Authors**: The PageRank algorithm was used to select authors with a score greater than 0.1, indicating high influence in the co-authorship network.
- **Selecting Potential Collaborators**: For each influential author, we considered authors who shared the same research domain (`domain_id`), had not previously collaborated, and had published more than two papers.
- **Calculating Similarity Scores**: A score was computed as:
  - 0.3 × (normalized paper count of author 1 × normalized paper count of author 2).
  - 0.3 × (normalized citation count of author 1 × normalized citation count of author 2).
  - 0.4 if the authors shared the same research domain.
- **Incorporating Citation Relationships**: Added 0.2 if one author's papers cited the other's, and an additional 0.2 for reciprocal citations.
- **Filtering and Ranking**: Pairs with a final score above 0.5 were ranked by score to identify likely collaborations.

This approach leveraged domain knowledge and network structure to provide an interpretable baseline without machine learning. The top predicted collaborations are shown in Figure 11.

Author1	Author2	Domain1	Domain2	Papers1	Papers2	Citations1	Citations2	PredictedScore
's. vertovec"	"p. levitt"	204	204	17	17	724	374	204.549
"p. levitt"	"s. vertovec"	204	204	3	17	298	724	162.767
"p. levitt"	"p. levitt"	204	204	3	17	298	374	84.542000000000
"j. carling"	"thomas faist"	40	40	20	16	344	293	76.954
's. vertovec"	"n. meer"	204	204	17	25	724	115	64.320000000000
"h. gans"	"r. waldinger"	39	39	13	22	329	249	63.098749999999

Fig. 11. Top Predicted Collaborations from Heuristic Approach

As a non-ML method, standard metrics like AUPRC were not computed. The effectiveness was assessed qualitatively through the predicted pairs and their scores, higher predicted scores showed more likely possible future collaborations.

2) Iteration 1 – FastRP Embeddings: The first machine learning iteration introduced Fast Random Projection (FastRP), with features extracted via the Hadamard product and fed into a logistic regression model. The model was trained with an 80/20 train-test split and negative sampling (ratio 1.0). Unlike the heuristic approach, this iteration used a classical ML pipeline, enabling quantitative evaluation.

The performance metrics were:

- **AUPRC (Test)**: 0.73

- **Validation AUPRC (avg)**: 0.72
- **Outer Train AUPRC (avg)**: 0.72

Fig. 12. Model Metrics for FastRP Iteration

Compared to the heuristic approach, this iteration offered scalability and learned representations, achieving a test AUPRC of 0.727. However, the predicted probabilities clustered around 0.5 (Figure 13), indicating limited confidence in distinguishing likely from unlikely collaborations. This clustering suggested that FastRP embeddings might not fully capture the nuanced relationships needed for effective link prediction.

	Author1	Author2	probability
1	"b. franklin"	"s. akashi"	0.5015344359545668
2	"s. sattler"	"s. sattler"	0.5015342971881407
3	"m. digiacomo"	"m. digiacomo"	0.5015341224438363
4	"m. digiacomo"	"l. ramjan"	0.5015335357243107

Started streaming 100 records after 41 ms and completed after 107641 ms.

Fig. 13. Results for FastRP Iteration

The shift to an ML pipeline was a significant improvement over the heuristic method's manual scoring, but the probability clustering highlighted a need for better embedding techniques.

3) Iteration 2 – Node2Vec Embeddings: To address the probability clustering issue, the second iteration employed Node2Vec embeddings (64 dimensions, walk length 80, 10 walks per node), the pipeline remained similar, using Hadamard product feature extraction and logistic regression, with the same 80/20 train-test split and negative sampling. Additionally, this iteration corrected an error from earlier iterations by ensuring unique author pairs (author1.id ¡ author2.id) to avoid self-loops.

The performance metrics were:

- **AUPRC (Test)**: 0.677
- **Validation AUPRC (min)**: 0.668
- **Validation AUPRC (max)**: 0.676
- **Validation AUPRC (avg)**: 0.671
- **Outer Train AUPRC (min)**: 0.670
- **Outer Train AUPRC (max)**: 0.673
- **Outer Train AUPRC (avg)**: 0.671

Fig. 14. Model Metrics for Node2Vec Iteration

Compared to the FastRP iteration, Node2Vec resulted in a lower test AUPRC (0.677 vs. 0.729) but significantly improved the probability distribution, showing greater variation with probabilities ranging from 0.9536 to 0.9738 (Figure 15). They showed much more variation when top 10 was changed to top 1000 while displaying the probabilities. This wider range indicated better calibration, enabling the model to confidently distinguish between likely and unlikely collaborations. The improvement stemmed from Node2Vec's ability to balance local

and global graph structures through biased random walks, making it more suitable for practical applications despite the lower AUPRC. The correction of self-loops further enhanced the model's reliability.

	Author1	Author2	probability
1	"c. frantz"	"fernando martínez"	0.9574353285828696
2	"c. frantz"	"iván periáñez"	0.9573609762951459
3	"c. rogers"	"fernando martínez"	0.9571775182794837
4	"c. rogers"	"iván periáñez"	0.9571048947589689
5	"c. frantz"	"angeles castaño"	0.9539603299413003

Fig. 15. Results for Node2Vec Iteration

The Node2Vec model was selected as the final model due to its superior probability calibration, offering a robust solution for predicting future collaborations.

VI. Discussion

The node classification task demonstrated that structural features like Louvain community IDs and FastRP embeddings can effectively predict an author's research domain, with the third iteration (64-dimensional embeddings, 40 trees) achieving the best performance (Accuracy: 0.79, F1 Score: 0.88). However, limitations included low-informative embeddings for authors with few connections and ambiguity from interdisciplinary authors, leading to some misclassifications.

For link prediction, the node2vec-based model was preferred over fastRP due to its varied probability estimates, despite a slightly lower AUPRC. The fastRP model's probabilities clustered around 0.5, suggesting limited discriminative power. Early iterations incorrectly included self-loops, which was corrected in the final model by ensuring unique author pairs.

A. Limitations and Challenges

A key limitation was the quality of node embeddings. Several authors—particularly those with few or no co-authors—received zero or low-informative embedding vectors, which hindered classification accuracy. Additionally, the presence of interdisciplinary authors, who collaborate across multiple research domains, introduced ambiguity in both graph structure and labels. These factors contributed to misclassifications and highlight the need for more robust features beyond network structure. Also, the training of the model was a resource heavy task, forcing us to limit our iterations for this task to only 3. Predictions also took really long to generate and we encountered lots of memory errors. Link prediction faced challenges with embedding quality and the need for more diverse features. The heuristic approach in the first iteration, while insightful, was not integrated into the ML pipeline, limiting its impact.

B. Alternative Approaches

For node classification, Node2Vec or Graph Neural Networks (GNNs) could capture more nuanced structures. Additionally, incorporating non-structural node features—such as publication keywords, journal categories, or citation counts—could enrich the feature set and provide context where structural signals are weak. These features could be extracted using NLP techniques applied to paper titles or abstracts.For link prediction, combining heuristic scores (e.g., from the first iteration) with learned embeddings or using GNNs could improve accuracy, clustering coeffecient can be incorporated as well.

C. Possible Extensions

Semi-supervised learning could infer domain labels for unlabeled authors. For link prediction, temporal features (e.g., collaboration recency) or additional metrics (e.g., common neighbors) could be explored. Hybrid models integrating multiple embedding techniques may further enhance both tasks.

References

[1] L. Rothenberger, M. Q. Pasta, and D. Mayerhoffer, "Mapping and impact assessment of phenomenon-oriented research fields: The example of migration research," *Quantitative Science Studies*, vol. 2, no. 4, pp. 1466–1485, 12 2021. [Online]. Available: https://doi.org/10.1162/qss_a_00163