An LLM-Guided Framework for Link Prediction in Homogeneous Graphs

Atul Kumar[1], Md Zamilur Rahman[1,2,3*], Asish Mukhopadhyay[1]

[1*]School of Computer Science, University of Windsor, 401 Sunset Avenue, Windsor, N9B 3P4, ON, Canada.
[2]Department of Computer Science and Mathematics, Algoma University, 1520 Queen Street East, Sault Ste. Marie, P6A 2G3, ON, Canada.
[3]NORDIK Institute, 1520 Queen Street East, Sault Ste. Marie, P6A 2G3, ON, Canada.

*Corresponding author(s). E-mail(s): zamilur.rahman@algomau.ca;
Contributing authors: atulkum@uwindsor.ca; asishm@uwindsor.ca;

Abstract

As social networks grow, link prediction has become vital in network analysis, estimating the likelihood of connections between unconnected nodes based on similarity scores. This study explores the intersection of Large Language Models (LLMs) and Graph Learning, with a particular focus on Link Prediction tasks on Homogeneous Networks, where we are using LLMs to analyze social network structures and predict missing links. There have been several studies that leveraged LLMs for Knowledge Graphs, Heterogeneous Graphs, and Text-Attributed Graphs. However, leveraging LLMs for Homogeneous Graphs with no textual information is still an understudied area, which is what we aimed to explore. We developed a framework that leverages LLMs for link prediction tasks requiring no textual information with different learning strategies. Our results demonstrate improvement in model performance for predicting missing links, especially when provided with few examples or fine-tuned on the domain-specific datasets, achieving results on par with state-of-the-art results, even with no fine-tuning.

Keywords: Link Prediction, Large Language Models, Homogeneous Networks, In-context Learning.

1 Introduction

More than half of the population on Earth uses social networks. Due to this rapid growth of social networks, link prediction has emerged as an important research topic in the field of network analysis. It involves calculating how similar the two nodes are, and based on that predicts the likelihood of a link between two unconnected nodes. In this paper, we aim to analyze the structure of social networks to compute several similarity scores between connected pairs of nodes in the network and predict the links between unconnected pairs of nodes based on those scores with the help of Large Language Models (LLMs). Although there have been several studies published studying Knowledge Graphs, Heterogeneous Graphs, and Text Attributed Graphs and leveraging LLMs for them. To the best of our knowledge, leveraging LLMs for Homogeneous graphs with no textual information has still not been explored yet, which is what we aimed to explore for the scope of this research. Although there are network embedding-based algorithms like VERSE (Tsitsulin et al., 2018), SEAL (Zhang and Chen, 2018), and LINE (Tang et al., 2015), that have been studied in the past and reported as the best performance to date for link prediction tasks (to the best of our knowledge). However, due to the LLMs potential, we developed an innovative approach for utilizing their potential for link prediction tasks, especially in homogeneous networks and with no textual information provided about the nodes or edges. We will begin by discussing the related works that have been done on the graph network by leveraging LLMs and our contributions. Then we provide the methodology of our approach, which will involve a detailed description of computed similarity coefficients, structuring data in the proper format, and prompt engineering, followed by approaches we performed without fine-tuning and with fine-tuning. Following this, we will be providing the pre-processing steps for the dataset in order to work with LLMs. Finally, we will analyze how effective our model is in predicting missing links in the social network by calculating prediction scores and discussing them.

2 Related Works

Several studies have explored the application of Large Language Models (LLMs) to link prediction tasks, demonstrating their potential in identifying and predicting relationships in social networks. So far, all the studies that utilize LLMs for the link prediction tasks have been done only on Knowledge graphs, Heterogeneous Graphs, or Graphs having textual attributes associated with nodes and/or edges. All the listed graphs always contain some type of textual information. However, Homogeneous Graphs with no textual attributes associated are yet to be explored. This is what our study aims to explore and gain insights into.

Bi et al. (2024) introduce LPNL (Link Prediction via Natural Language), a framework leveraging LLMs for scalable link prediction on large-scale heterogeneous graphs. This involved designing natural language prompts and employing a two-stage pipeline to predict the links, outperforming several advanced baselines.

There has been work done by Chen and Shen (2025), where they address link prediction and graph completion tasks in temporal knowledge graphs using LLMs. Their approach leverages the textual information present in knowledge graphs, enabling

LLMs to better understand and reason about temporal relationships. This method has shown promise in improving the performance of link prediction and completion tasks, particularly in scenarios where the graph contains rich textual descriptions.

LinkGPT, introduced by He et al. (2024), addresses key challenges in using LLMs for link prediction by proposing a novel two-stage instruction tuning approach. This method fine-tunes the LLMs understanding of graph structures, significantly improving link prediction performance. However, this framework was proposed for text-attributed graphs (TAGs) only, which always have some text associated with nodes and/or edges. This framework leverages the power of LLMs to effectively capture both graph structure and textual semantics to make accurate predictions.

There has been another work which is recently done by Shu et al. (2024), focusing on multi-hop link prediction in knowledge graphs. Their proposed framework, the Knowledge Graph Large Language Model (KG-LLM), leverages LLMs by converting structured knowledge graph data into natural language prompts. This approach enables LLMs to reason through intermediate connections for enhanced multi-hop link prediction.

Tsitsulin et al. (2018) propose VERSE, a versatile and efficient graph embedding method designed to preserve vertex-to-vertex similarity measures. Unlike previous approaches that adapted methods from word embeddings to graphs without clear graph-specific objectives, VERSE directly focuses on maintaining the distribution of selected similarity measures among graph nodes. The method trains a single-layer neural network and offers both a scalable, sampling-based variant and a full-information variant. Wu et al. (2022) demonstrated through an experimental survey that VERSE excels in link prediction tasks on complex networks, ranking first across all 36 datasets considered in their study.

While heuristic algorithms were once predominant, Graph Neural Networks (GNNs) for Link Prediction have been prevalent in the past few years. Khanna et al. (2024) explored link prediction in social networks by comparing ten feature extraction techniques, including structural embeddings, graph neural networks, and graph heuristics. Their study found that combining heuristic-based features with learned representations enhances link prediction accuracy.

Graph Convolutional Network (GCN) based models have gained attention for their effectiveness in link prediction within knowledge graphs. For instance, Lin et al. (2022) proposed ConGLR that enhanced inductive relation prediction by incorporating context graphs and logical reasoning, making it well-suited for scenarios where new relationships must be inferred. Similarly, Le et al. (2023), ConvRot integrate relational rotation techniques with convolutional operations, leading to improved link prediction performance by better capturing the structural and relational patterns in knowledge graphs. These approaches underscore the potential of GCN-based models in advancing knowledge graph analysis.

All the existing work, that considers LLM as an ML model instead of conventional ML algorithms, has focused on knowledge graphs, heterogeneous graphs, and text-attributed graphs, but there has been limited to almost no research that leverages LLMs for homogeneous graphs that have only one type of node and no textual information. This is what we have aimed to explore and get the insights.

3 Methodology

A homogeneous graph is a type of graph where all nodes and edges are of the same type, i.e. it contains a single category of entities and relationships throughout the graph. According to Daud et al. (2020), we can classify the methods of link prediction into the following four categories:

1. **Similarity:** Similarity method is one of the popular approaches in link prediction, where we calculate how similar the two nodes are and determine the likelihood of a connection between them. The greater the similarity score of two nodes, the higher the chances of forming a link between nodes. For this, we calculate several similarity metrics such as Common Neighbor, Jaccard Coefficient, etc.
2. **Probabilistic:** It involves building a model that is based on the characteristics of the network structure, which aims to identify the underlying pattern of link formation in the given network. Then we calculate the probability value of each pair of nodes that is unconnected. Based on that probability value, we calculate the likelihood of a connection between two nodes. The greater the probability value of two nodes, the higher the chances of forming a link between nodes.
3. **Algorithmic:** This approach is widely used in predicting the links between the nodes. It involves analyzing the structure of the network and predicting the link based on a set of rules and procedures to identify the patterns in the network. One of the algorithmic methods used for predicting the links is machine learning techniques.
4. **Hybrid:** Method is said to be hybrid if we combine two or more of the previously mentioned methods for enhancing the accuracy and robustness of the approach.

For this research, we have implemented a hybrid approach for predicting the links, where we have combined the Similarity approach with the Algorithmic approach.

3.1 Similarity

As discussed, the similarity approach involves calculating the similarity scores between the given two nodes based on several similarity metrics. These scores tell how similar those two nodes are, and depending on the score, it predicts whether there should be a link exists between them or not. Nie et al. (2016) similarly explored structural information from users' egocentric networks such as followees or retweet interactions, in order to model the user behavior in social networks, which aligns conceptually with our neigborhood-based similarity metrics approach, such as Jaccard, Adamic-Adar, etc. Similarity metrics can be divided into the following 3 broad categories (Daud et al., 2020):

1. **Local Indices:** These indices only take the immediate neighbors into account to find the similarity scores. That means it requires minimal knowledge of the network structure to find the similarity between two nodes. We calculated several types of scores, such as *Jaccard*, *Common Neighbor*, *Preferential Attachment*, *Sorensen*, *Salton-Cosine*, *Adamic-Adar*, and *k-NN Weights*.
2. **Global Indices:** Unlike local indices, which are just based on the common connections with the immediate neighbors (which may not be that effective), global

indices consider the whole structure of the current network and calculate the similarity scores concerning the whole structure, not just the immediate neighbors. Here we calculated indices like *Katz* and *Node Community*[1].

3. **Quasi-local Indices** Quasi-local indices are used to have additional information, likewise the global indices, but compute the score based on the nodes that are their immediate neighbors. So, the computational cost of calculating the similarity scores using the quasi-local Indices approach is not as high as that of global indices. We calculated scores such as *HITS* and *PageRank*, which determine the importance and relevance of each node in the network.

We calculated all the above-mentioned similarity scores. However, we selected only a subset of them to reduce the model complexity. Also, including all the scores could lead to model hallucinations and may affect the overall interpretability of LLMs. This adheres to a similar principle as Nie et al. (2016), who leveraged the information gain ratio to identify the most important features and selected the top five as representative features. In our case, we identified the most informative features by computing the *Pearson Correlation Coefficient* between each similarity score and the label. Based on this analysis, we found the following scores: *Jaccard*, *Common Neighbor*, *Preferential Attachment*, *Sorensen*, *Salton-Cosine*, *Adamic-Adar*, *Node Community*, and *PageRank* to be highly correlated with labels across all the considered datasets.

3.2 Algorithmic

The algorithmic approach is widely used in predicting the links. The algorithmic method that we have used here for predicting the links is by using Large Language Models (LLMs). Since we are using the hybrid approach, we can use those calculated similarity scores to feed to LLMs and predict the links.

3.3 Large Language Models (LLMs) and Prompt Engineering

Large Language Models have brought advancements in natural language processing, where models can now generate human-like text, understand the context behind it, and perform tasks such as text classification, summarization, question-answering, etc. They are already pretrained on massive amounts of data, which allows them to understand the semantic and syntactic meaning of the asked query and generalize the questions across several domains, returning contextually relevant responses.

This semantic understanding has been proven to be highly effective for several machine-learning tasks across several domains, especially in natural language tasks. For our research, we have utilized this understanding of LLMs for the Link Prediction task while structuring the data in a manner that LLMs can interpret easily, reducing the chances of hallucinations.

In this research, we utilized several LLMs such as *Meta-Llama-3-8B-Instruct* (Dubey et al., 2024), *Meta-Llama-3.1-8B-Instruct* (Dubey et al., 2024), *DeepSeek-R1-Distill-Llama-8B* (DeepSeek-AI et al., 2025), *Gemini-1.5-Pro* (Team

[1]Available at: https://networkx.org/documentation/stable/reference/algorithms/generated/networkx.algorithms.community.modularity_max.greedy_modularity_communities.html

et al., 2024), and *Gemini-1.5-Flash* (Team et al., 2024) to evaluate their effectiveness, individually and collectively, when dealing with the Link Prediction task, using only similarity scores (numerical values) and context about how to deal with it, with no additional textual information about the nodes and edges. We frame the static link prediction problem as a binary classification problem, purely based on the static snapshots of the graph, where existing (i.e., present) edges are treated as positive samples and non-existing (i.e., missing) edges are treated as negative samples.

The positive samples consist of a list of pairs of nodes representing *present* (i.e., existing) edges in the graph, which we label as 1 (or *yes*). However, the original datasets only include *present* edges (i.e., positive samples). That is why we generate the *missing* edges (i.e., negative samples) by randomly selecting a pair of nodes that are neither connected directly nor lie within a 2-hop neighborhood, hence they never have any shared neighbors or short paths between them. In simple words, if either the nodes are connected directly or the path length between them is less than or equal to 2, then it cannot be considered as a *missing* edge (i.e. negative sample). The valid generated missing edge is labeled as 0 (or *no*). Through this process, we generate the negative samples equal to the number of positive samples to maintain the balance. For example, in Figure 1, there could be a missing edge between A and D. However, there can not be any missing edge between A and C, as they have fewer than two intermediary nodes in between.

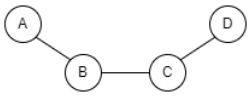

Fig. 1 Sample Graph

Once we have the positive and negative samples in the dataset, we construct the graph, using both positive and negative node pairs, for the link prediction task. Since the negative samples do not represent actual connections, however, included as only contrastive examples to train the model to distinguish between *present* and *missing* links. We then compute similarity scores for all node pairs using the earlier-mentioned similarity-based heuristics to assess the likelihood of a link.

Since it may not lead to the best results by feeding the similarity scores in the raw format without any context, that is why we generate the structured sentences for each *present* and *missing* edge in the dataset. These structured sentences have all the information, depending on what LLMs can decide to classify as an edge, a *present* or a *missing*. The scores are presented in an appropriate format, along with the corresponding similarity coefficient scores for each edge, as shown below:

For nodes u with ID 6583 and v with ID 21030, they belong to the same community. They have a common neighbor score (CN) of 0.8421052631578947, and similarity scores of 0.8421052631578947 for salton similarity score (SCS), 0.4210526315789473 for Sorenson, 0.7272727272727273 for Jaccard, 0.4039308631599071 for Adamic/Adar Index (AAI), and 19.0 for Preferential Attachment. The PageRank scores are 0.0002443 for node u (PR_U) and 0.0002189 for node v (PR_V).

Now, the next step is about designing the prompt so that LLMs know the task provided and can interpret the provided data effectively. Designing a prompt is considered to be a very crucial step, as the behavior of the models relies heavily on this, and it needs to be very specific about all the instructions and constraints. We encapsulated the above sentence inside the prompts so that LLMs know exactly what they have to do with the given information, and what and how to output. Table 1 shows how the prompts for the training set look. For the test set, since we have to predict the value for the "prediction", we just leave the prediction line empty. Detailed examples of the prompt format for each strategy (zero-shot, one-shot, few-shot, and fine-tuning) are provided in the Supplementary Material document.

Prompt:

Table 1 Prompt used across zero-shot, one-shot, few-shot, and fine-tuning approaches in our experiments.

Link Prediction Task
Instructions
Based on the following information, determine whether a link exists between two nodes. Respond strictly with either "yes" or "no", and nothing else.
Consider the following attributes:
-Community membership
-Common Neighbor Score (CN)
-Similarity Scores: Salton Similarity Score (SCS), Sorenson, Jaccard, Adamic/Adar Index (AAI), Preferential Attachment (PAS)
-PageRank Scores for both nodes (PR_U and PR_V)
Below is an example of the information and its corresponding prediction to learn from:
statement: "***statement***"
prediction: "***prediction***"

This is how we transformed our raw input data to a more structured format while providing the context of the problem through prompt engineering and further restricting the model to just generate either *yes* (if a link exists) or *no* (if a link does not exist).

LLMs can learn through several approaches, such as zero-shot, one-shot, and few-shot, where we do not need any fine-tuning on the dataset. LLMs can either use the pre-trained knowledge or use the examples provided in the prompt for further context. Before further discussion, it is important to note that all the input prompts provided to different models were the same to ensure consistency and fair comparison. For the zero-shot approach, as usual, we provided no examples from the datasets to any of the models and all the predictions were made only using the prior knowledge of the LLMs. Later, we utilized in-context learning for one-shot and few-shot approaches, where all the models were provided with randomly selected one and five examples per class, directly within the input prompt, respectively. This type of learning allows the model

to adapt and make predictions based on the given examples without any need for fine-tuning. The prompts for the one-shot and few-shot were designed the same as given above, except we provided an appropriate number of examples from specific datasets.

Although the above-mentioned learning paradigms returned good results (details in Section 5), even without fine-tuning. However, fine-tuning the LLMs on the specific dataset can also drastically improve the results. This is why we also opted for another learning paradigm, that is, fine-tuning the Large Language Model. In this, we instructional fine-tuned the following models: *Meta-Llama-3-8B-Instruct*, *Meta-Llama-3.1-8B-Instruct*, and *DeepSeek-R1-Distill-Llama-8B*. The reason behind fine-tuning these specific models is that they are open-source and require no cost to fine-tune or do inference, as long as we have enough computational power. This is for what we leveraged *SHARCNET* resources, which were significantly invaluable for our work. Since *Gemini* models require API access and incur costs for every token they generate. That is why we keep limiting ourselves to fine-tuning only the above-mentioned models, whereas we have used *Gemini* for only zero-shot, one-shot, and few-shot approaches.

3.4 Graphical Architecture of our Approach

Let us look at the graphical architecture of both approaches, i.e. without or with fine-tuning.

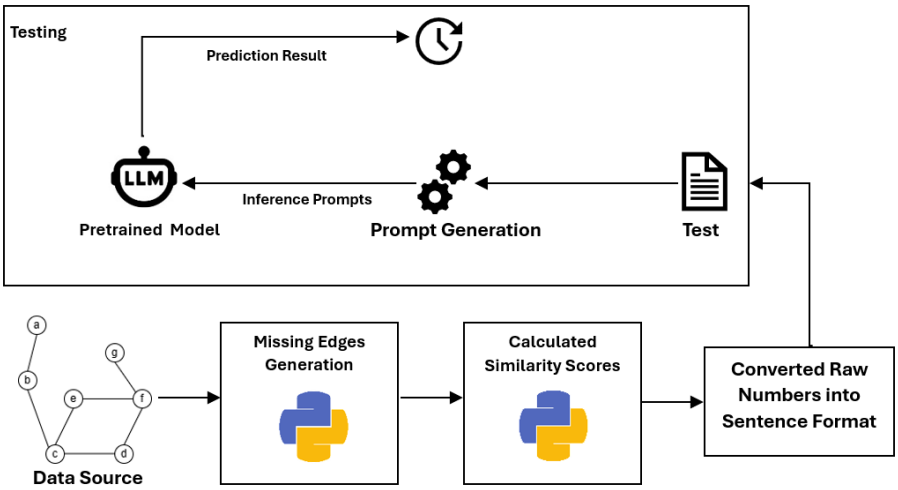

Fig. 2 Diagram showing the workflow for Zero-Shot, One-Shot, and Few-Shot Approaches

1. **Zero-Shot, One-Shot, and Few-Shot Approaches (without fine-tuning):** Firstly, we start with the original dataset and generate the missing edges (labeled as "0"), which means they do not exist. Then, we calculate the similarity scores for both present and missing edges and convert them into a natural language sentence, as we saw earlier. Once we have all the sentences, we split the test, training, and

evaluation sets. However, we use only the test set for these three learning approaches to ensure we can make fair comparisons later with fine-tuning and baseline results. Finally, we encapsulate the sentences into properly structured prompts and pass them through the LLMs to predict the edge status, i.e. whether the edge is present or missing. The overall flow diagram for this approach is shown below in Figure 2

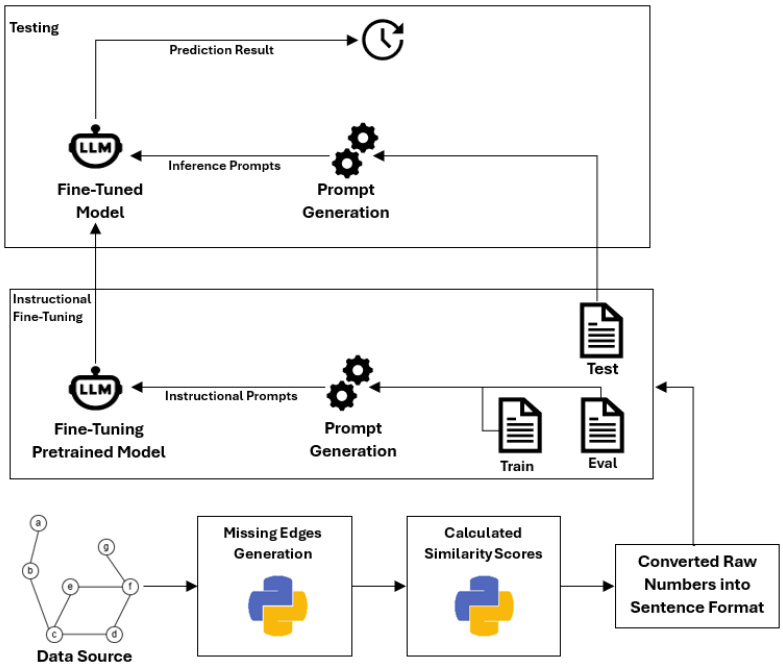

Fig. 3 Diagram showing the workflow for Fine-Tuning Approach

2. **Fine-Tuning Approach:** Now, let us look at the flow diagram for the overall fine-tuning process, where most of the things on the inference part remain the same as above, except that the model we used is fine-tuned on our datasets as shown in Figure 3. In this, again, we start with generating the missing edges, followed by calculating the similarity score, and, later, converting each of them into a properly structured sentence. Following this, we divide our dataset into three parts: training, validation, and test sets. After that, we utilize training and validation sets and encapsulate them in the same prompt format, which we saw earlier, and fine-tune *Meta-Llama-3-8B-Instruct*, *Meta-Llama-3.1-8B-Instruct*, and *DeepSeek-R1-Distill-Llama-8B*, on all three datasets separately. Then, similar to the zero-shot approach, we encapsulate the test dataset in the same way and use the fine-tuned models for performing the inference on the test set.

4 Datasets

We have used three different undirected graph networks from SNAP[2] to measure the effectiveness of our methodology. All three graph networks are homogeneous and with no textual information, where Facebook (McAuley and Leskovec, 2012) is a social network dataset, and Gr-Qc (Leskovec et al., 2007) and Hep-Th (Leskovec et al., 2007) are collaboration networks. Although all three datasets represent undirected graphs, the edge list formats vary. For the Gr-Qc and Hep-Th datasets, the files explicitly include both directions for each edge (i.e., if node A is connected to node B, both the edges $A \rightarrow B$ and $B \rightarrow A$ are listed). However, for the Facebook dataset, we have only one direction per edge, assuming undirected connectivity. To ensure consistency and fairness in experiments, we standardized the edge lists by removing redundant reciprocal edges, aligning the format with that of the Facebook dataset and treating all graphs as undirected, as they are originally intended to be. Each of the graph datasets consists pair of node IDs, where each pair represents the presence of an edge in the graph network. The statistics for these graphs are shown in Table 2:

Table 2 Dataset Statistics

Dataset	Nodes	Edges
Gr-Qc[a] (Leskovec et al., 2007)	5242	14496
Hep-Th[b] (Leskovec et al., 2007)	9877	25998
Facebook[c] (McAuley and Leskovec, 2012)	4039	88234

First and foremost, we pass all three graph datasets through the pre-processing steps to feed to LLMs, which are mentioned as follows:

1. For positive samples, we start by importing the dataset, which consists of a pair of nodes. Each pair represents an existing connection in the graph, also called a present edge, and is labeled as 1 (or *yes*).
2. As detailed earlier in Section 3.3, since the datasets only contain node pairs that are *present*. So to generate the *missing* edges (i.e., negative samples), we randomly select a pair of nodes that are neither connected directly nor lie within a 2-hop neighborhood, in order to create an equal number of negative samples.
3. Once we have the positive and negative samples in the dataset, we construct the graph for the link prediction task. Following this, we compute similarity scores for each node pair and convert them into natural language sentences, as we mentioned earlier.
4. Next, we randomly split the combined datasets, as they are static snapshots, into training, validation, and test sets with a ratio of 80%, 10%, and 10%, respectively. Each set contains equal and randomly chosen positive and negative samples, as prepared in Step 3.

[2] http://snap.stanford.edu/data
[a] Available at: https://snap.stanford.edu/data/ca-GrQc.html
[b] Available at: https://snap.stanford.edu/data/ca-HepTh.html
[c] Available at: https://snap.stanford.edu/data/egonets-Facebook.html

5. Finally, we encapsulate all the sentences into the prompts with the corresponding target labels, except for the test dataset, and validate the performance of the LLMs without fine-tuning (i.e. zero, one, and few-shot approaches), with fine-tuning, and on the baseline models.

Note: We calculate the similarity scores for training, validation, and test sets, however, we use only the graph that we generate from the training set. This step is taken to ensure no information is included from the validation and test sets for further evaluation, simulating a real-world scenario.

5 Experimental Results

5.1 Zero-Shot, One-Shot, and Few-Shot Approaches (Without Fine-Tuning)

As discussed in the previous sections, we employed three popular network datasets and evaluated them with the help of five different LLMs on the link prediction task. The results yielded through zero, one, and few-shot approaches were promising in most cases, although performance varied across datasets and approaches. However, after fine-tuning, the results significantly improved, with some achieving almost 100% accuracy, compared to their non-fine-tuned counterparts. These findings demonstrate the effectiveness of our approach across all considered datasets. While state-of-the-art algorithms like VERSE (Tsitsulin et al., 2018) report the best performance to date, which uses a graph embedding technique by training a single-layer neural network, our study emphasizes the advantage of LLMs. Their advanced neural architectures and the potential for capturing and understanding complex patterns in data make them a compelling alternative for the graph-based problems that may not have any textual data.

5.1.1 Gr-Qc

- Zero-Shot Approach: As we can see in Table 3, in the absence of task-specific examples, both Llama-3-8B-Instruct and Llama-3.1-8B-Instruct models achieved the same accuracy of 90.99%, demonstrating strong zero-shot capabilities, with a good balance between the precision and recall. In contrast, DeepSeek-R1-Distill-Llama-8B, Gemini-1.5-Flash, and Gemini-1.5-Pro had underperformed in maintaining a good balance between precision and recall, compared to the Llama models. In the case of the DeepSeek model, it showed the precision of only 78.10% and a recall of more than 95%, showing it has been more tolerant in predicting positive links and hence creating more false positives. The opposite trend to DeepSeek has been seen for the Gemini Models, where both variants achieved a perfect precision of 100% but very less recall values (only 59.30% for Flash variant and 74.61% for the Pro variant), suggesting it was highly conservative, that is, predicting the links when it was highly confident while classifying rest as non-links.

Table 3 Performance comparison of LLMs and baseline methods on the Gr-Qc dataset across Zero-Shot, One-Shot, Few-Shot, and Instructional Fine-tuning settings. Performance metrics include Accuracy, Precision, Recall, and F1 score, demonstrating the effectiveness of different models and training approaches against baseline methods. The model with the highest accuracy is emphasized in bold, while the model with the second highest accuracy is underlined.

		Gr-Qc			
Category	**Model**	**Accuracy**	**Precision**	**Recall**	**F1**
Zero-Shot	*LLAMA-3-8B-Instruct*	90.99%	92.84%	88.95%	90.85%
	LLAMA-3.1-8B-Instruct	90.99%	92.84%	88.95%	90.85%
	DeepSeek-R1-Distill-Llama-8B	84.26%	78.10%	95.47%	85.92%
	Gemini-1.5-Flash	79.53%	100.00%	59.30%	74.45%
	Gemini-1.5-Pro	87.23%	100.00%	74.61%	85.46%
One-Shot	*LLAMA-3-8B-Instruct*	90.99%	92.84%	88.95%	90.85%
	LLAMA-3.1-8B-Instruct	87.43%	99.10%	75.70%	85.84%
	DeepSeek-R1-Distill-Llama-8B	92.03%	92.99%	91.01%	91.99%
	Gemini-1.5-Flash	87.47%	99.46%	75.50%	85.84%
	Gemini-1.5-Pro	89.71%	99.74%	79.75%	88.63%
Few-Shot	*LLAMA-3-8B-Instruct*	90.99%	92.84%	88.95%	90.85%
	LLAMA-3.1-8B-Instruct	90.99%	92.84%	88.95%	90.85%
	DeepSeek-R1-Distill-Llama-8B	91.02%	92.90%	88.95%	90.88%
	Gemini-1.5-Flash	87.47%	100.00%	75.09%	85.77%
	Gemini-1.5-Pro	90.89%	93.44%	88.06%	90.67%
Instructional Fine-tuning	*LLAMA-3-8B-Instruct*	95.72%	96.97%	94.44%	95.69%
	LLAMA-3.1-8B-Instruct	<u>95.75%</u>	<u>97.57%</u>	<u>93.89%</u>	<u>95.70%</u>
	DeepSeek-R1-Distill-Llama-8B	95.34%	96.03%	94.65%	95.33%
Baseline	*VERSE*	**99.83%**	**99.66%**	**100.00%**	**99.83%**
	FCNN	89.92%	99.91%	80.03%	88.87%

- One-Shot Approach: With minimal supervision, DeepSeek-R1-Distill-Llama-8B showed the most noticeable improvement as shown in Table 3, achieving an accuracy score of over 92% with a good balance between precision and recall. This shows that this architecture benefits even from minimal guidance. On the other hand, both the Llama models showcased similar performance, as in the zero-shot approach. Although a significant drop in recall and a significant rise in the precision have been observed for Llama-3.1-8 B-Instruct, possibly due to overfitting to the single instance provided. Lastly, both variants of the Gemini models have also demonstrated an overall improvement in performance. The Flash variant has achieved an accuracy of 87.47%, almost an 8% increase compared to the zero-shot. This improvement has been substantially gained from an increase in recall by over 16%. Similarly, Gemini-1.5-Pro has also shown an overall improvement in the result, while maintaining almost the same precision as zero-shot, but gaining a 5% increase in the recall.
- Few-Shot Approach: With 5 examples provided for each class (that is, link exists or not), all models showed similar and near-optimal performance, as evident from Table 3. The Llama, DeepSeek, and Gemini-1.5-Flash models maintained the same

performance as the zero-shot and one-shot, suggesting little to no improvement in their stronger understanding from additional examples. However, Gemini-1.5-Pro showed a massive increase in the recall, by more than 13% and 8% compared to zero-shot and one-shot, respectively, with a little drop in precision, achieving almost the same accuracy as Llama and DeepSeek models. This suggests that while models like Llama and DeepSeek may generalize well with just one or no examples at all, Gemini models might need more context to better understand the graph topology and make correct predictions.

Table 4 Performance comparison of LLMs and baseline methods on the Hep-Th dataset across Zero-Shot, One-Shot, Few-Shot, and Instructional Fine-tuning settings. Performance metrics include Accuracy, Precision, Recall, and F1 score, demonstrating the effectiveness of different models and training approaches against baseline methods. The model with the highest accuracy is emphasized in bold, while the model with the second highest accuracy is underlined.

		Hep-Th			
Category	**Model**	**Accuracy**	**Precision**	**Recall**	**F1**
Zero-Shot	*LLAMA-3-8B-Instruct*	87.60%	90.68%	83.66%	87.03%
	LLAMA-3.1-8B-Instruct	87.60%	90.68%	83.66%	87.03%
	DeepSeek-R1-Distill-Llama-8B	84.08%	77.00%	96.94%	85.83%
	Gemini-1.5-Flash	72.67%	100.00%	45.02%	62.09%
	Gemini-1.5-Pro	86.62%	100.00%	73.09%	84.45%
One-Shot	*LLAMA-3-8B-Instruct*	87.60%	90.68%	83.66%	87.03%
	LLAMA-3.1-8B-Instruct	79.98%	99.55%	60.01%	74.88%
	DeepSeek-R1-Distill-Llama-8B	92.07%	91.54%	92.60%	92.07%
	Gemini-1.5-Flash	86.37%	97.86%	74.22%	84.41%
	Gemini-1.5-Pro	88.16%	99.75%	76.38%	86.52%
Few-Shot	*LLAMA-3-8B-Instruct*	87.60%	90.68%	83.66%	87.03%
	LLAMA-3.1-8B-Instruct	87.60%	90.68%	83.66%	87.03%
	DeepSeek-R1-Distill-Llama-8B	87.62%	90.79%	83.58%	87.04%
	Gemini-1.5-Flash	86.35%	100.00%	72.55%	84.09%
	Gemini-1.5-Pro	87.68%	90.84%	83.66%	87.10%
Instructional Fine-tuning	*LLAMA-3-8B-Instruct*	95.51%	97.23%	93.65%	95.40%
	LLAMA-3.1-8B-Instruct	<u>95.57%</u>	<u>97.04%</u>	<u>93.96%</u>	<u>95.48%</u>
	DeepSeek-R1-Distill-Llama-8B	95.23%	97.10%	93.19%	95.10%
Baseline	*VERSE*	**99.92%**	**99.84%**	**100.00%**	**99.92%**
	FCNN	92.57%	99.59%	85.40%	91.95%

5.1.2 Hep-Th

- Zero-Shot Approach: It is evident from Table 4 that, without access to any examples, both Llama-3-8B-Instruct and Llama-3.1-8B-Instruct performed similarly, achieving an accuracy of over 87%, with high precision (90.68%) and a slightly lower recall (83.66%). This shows the stronger generalization ability of both Llama models in a

zero-shot manner, however, with a slight bias towards avoiding false positives. Contrarily, DeepSeek-R1-Distill-Llama-8B, even though with almost the same accuracy (84.08%) as Llama models, showed an opposite trend and a great imbalance between the precision (77%) and recall (96.94%), suggesting model aggressiveness in predicting positive links by trying to capture more and more true positives at the cost of increasing false positives. Lastly, for the Gemini models, we noticed the same trend as the Gr-Qc dataset. Gemini-1.5-Flash with no examples provided, diverged significantly, recording the lowest accuracy of 72.67%, with a precision of perfect 100%, and extremely low recall of 45.02%, showing the model has been overly cautious while predicting positive links. However, when it does predict a link, it is always correct. Almost similar case was with the Pro variant of Gemini has been noticed, but with much better recall (73.09%) with an accuracy of 86.62%.

- One-Shot Approach: With minimal supervision, once again, DeepSeek-R1-Distill-Llama-8B showed the most substantial improvement, achieving an accuracy of over 92% with a great balance between the precision (91.54%) and recall (92.60%), as detailed in Table 4. It suggests that even with minimal context of a problem, the model could maintain a balance between identifying the actual links correctly and ensuring most of the predictions are accurate. With regards to Gemini models, Gemini-1.5-Flash has also improved significantly with respect to recall, gaining an increase of almost 30%. Hence, being less cautious compared to the zero-shot approach, while still maintaining almost the same precision (97.86%). As shown in the same table, the Pro variant of Gemini also performed a little better than its zero-shot counterpart, showing these models could benefit from even minimal guidance. Noticeably, Llama-3.1 accuracy drops to just around 80%, despite achieving near-perfect precision (99.55%), with recall of just 60.01%. This indicates that the model has become very cautious by just providing one example. Meanwhile, Llama-3 shows no change in the scores compared to its zero-shot results, which suggests greater robustness to changes in prompting.
- Few-Shot Approach: With access to multiple examples, most models showed little to no improvement in terms of overall performance, with the notable exception of Gemini-1.5.Pro (refer to Table 4). The drop in the overall performance that Llama-3.1 has seen during the one-shot approach has been recovered when provided with multiple examples, achieving the same score as the zero-shot approach, where Llama-3 still maintained the same performance. Compared to the one-shot approach, DeepSeek also witnessed a drop of more than 4% in terms of accuracy, primarily due to lower recall (83.58%), suggesting the model has become a little cautious while predicting positive links when provided with some examples. The Flash variant of Gemini also performed almost the same as its one-shot variant. Hence, Llama and Gemini-Flash models indicate that task-specific context reinforces their understanding, but do not imply a drastic change in their behavior. Lastly and interestingly, Gemini-1.5-Pro experienced a significant boost in terms of recall when provided with a few examples per class. It outperformed all the models, in terms of overall performance, under the few-shot approach and has become the second-best choice, out of all pretrained LLMs, for this dataset, after DeepSeek in the one-shot approach, with an accuracy of 87.68%.

Table 5 Performance comparison of LLMs and baseline methods on the Facebook dataset across Zero-Shot, One-Shot, Few-Shot, and Instructional Fine-tuning settings. Performance metrics include Accuracy, Precision, Recall, and F1 score, demonstrating the effectiveness of different models and training approaches against baseline methods. The model with the highest accuracy is emphasized in bold, while the model with the second highest accuracy is underlined.

		Facebook			
Category	**Model**	**Accuracy**	**Precision**	**Recall**	**F1**
Zero-Shot	*LLAMA-3-8B-Instruct*	95.43%	94.74%	96.24%	95.49%
	LLAMA-3.1-8B-Instruct	95.43%	94.74%	96.24%	95.49%
	DeepSeek-R1-Distill-Llama-8B	92.73%	87.41%	99.90%	93.24%
	Gemini-1.5-Flash	85.36%	100.00%	70.83%	82.92%
	Gemini-1.5-Pro	98.01%	100.00%	96.04%	97.98%
One-Shot	*LLAMA-3-8B-Instruct*	95.43%	94.74%	96.24%	95.49%
	LLAMA-3.1-8B-Instruct	95.87%	99.99%	91.78%	95.71%
	DeepSeek-R1-Distill-Llama-8B	96.28%	94.83%	97.93%	96.36%
	Gemini-1.5-Flash	97.86%	99.89%	95.83%	97.82%
	Gemini-1.5-Pro	99.03%	99.95%	98.11%	99.03%
Few-Shot	*LLAMA-3-8B-Instruct*	95.44%	94.74%	96.25%	95.49%
	LLAMA-3.1-8B-Instruct	95.43%	94.74%	96.24%	95.49%
	DeepSeek-R1-Distill-Llama-8B	95.52%	94.94%	96.21%	95.57%
	Gemini-1.5-Flash	97.85%	100.00%	95.71%	97.81%
	Gemini-1.5-Pro	96.55%	97.03%	96.07%	96.55%
Instructional Fine-tuning	*LLAMA-3-8B-Instruct*	**99.98%**	**99.98%**	**99.98%**	**99.98%**
	LLAMA-3.1-8B-Instruct	99.94%	99.93%	99.95%	99.94%
	DeepSeek-R1-Distill-Llama-8B	<u>99.95%</u>	<u>99.95%</u>	<u>99.94%</u>	<u>99.95%</u>
Baseline	*VERSE*	**99.98%**	**99.98%**	**99.99%**	**99.98%**
	FCNN	99.68%	100.00%	99.36%	99.68%

5.1.3 Facebook

- Zero-Shot Approach: In the absence of any examples, once again, both the Llama models showed identical overall performance, with an accuracy over 95% (see Table 5). Both variants were able to maintain a good balance between the precision (94.74%) and recall (96.24%), unlike in the other two datasets, where precision was slightly higher than recall. Once again, both demonstrated a strong generalization, without providing task-specific examples. With regards to DeepSeek, it showed a slightly lower accuracy compared to the Llama models, which was over 92%. However, there has been an imbalance between precision (87.41%) and recall (99.90%), following a similar trend observed in the other two datasets for DeepSeek under the zero-shot approach. This suggests that a model like DeepSeek, with no examples, could be very aggressive in predicting positive links and lead to an increase in false positives. Gemini-1.5-Flash, on the other hand, was again able to attain a perfect 100% precision, while a recall of just around 71%. With this perfect precision and such a lower recall, it suggests that Gemini models, with not much information about the task-specific examples, are very much expected to adopt a very selective strategy in predicting positive links. This surely eliminates false positives, but also

causes a significant number of true positives to be missed. Surprisingly, for the Pro variant of Gemini in the zero-shot setting, it outperformed all the other models in this setting with an accuracy of 98.01%, along with maintaining a balance between precision (100%) and recall (96.04%). This beat both Llama models, DeepSeek, and the Flash variant of Gemini across all three settings, attaining the second-highest performance for this dataset with LLMs, as detailed in Table 5.

- One-Shot Approach: Given a single example for each class, DeepSeek again experienced one of the most substantial improvements in the one-shot setting, reaching an accuracy of 96.28%, as shown in Table 5, primarily due to improvement in the precision (94.83%) with recall (97.93%). So, for models like DeepSeek, providing one example for each class could be more beneficial than providing a few of them per class. Unlike what we have seen this far, Gemini-1.5-Flash showed the most notable and significant gains by outperforming both Llama and DeepSeek models under this setting, achieving an accuracy of almost 98%. Although the precision (99.89%) was almost the same as the zero-shot, but recall got a significant boost from 70.83% in zero-shot to 95.83% in the one-shot approach. Despite its selective approach in the zero-shot approach, the model was able to significantly improve with just one example. Similarly, Gemini-1.5-Pro also experienced an increase in the overall performance compared to its zero-shot counterpart, by over 1% increase in accuracy, reaching 99%. It attained a nearly-perfect precision of 100% and a recall of over 98%. For this dataset, this model has outperformed all other models across all settings, achieving an accuracy on par with state-of-the-art results without fine-tuning. Llama models showed the same trends as the other datasets. Llama-3-8B-Instruct remained stable at 95.43% accuracy, suggesting greater robustness to changes in prompting. While Llama-3.1 demonstrated a slight increase in the accuracy (95.87%), but while attaining nearly-perfect precision of 100% and more than 4% drop in the recall compared to zero-shot. This shows, with one example, Llama-3.1 gets very selective in picking up true positives and eventually misses them.
- Few-Shot Approach: In reference to the same Table 5, even with several examples per class, none of the models were able to surpass the performance achieved by Gemini-1.5-Pro in the zero-shot and one-shot settings. Llama-3 still remained stable at the same performance as its zero-shot and one-shot counterparts, while Llama-3.1 has been able to achieve the gain in recall that it has lost during one-shot, achieving the same score as its zero-shot variant. While DeepSeek, Gemini-1.5-Flash, and Gemini-1.5-Pro had all seen a slight drop in their performance, reaching accuracies of 95.52%, 97.85%, and 96.55%, respectively. Interestingly, all five models in this case showed a very good balance between precision and recall. This suggests that, with several examples, models can structure the task more effectively, which leads to predictions that neither overestimate nor overlook positive links.

The LLMs demonstrated strong performance, achieving high accuracy scores, particularly impressive given that they had no prior exposure to the data. Overall, all the models performed well, except where recall is sometimes low in the case of Gemini models or recall is way higher than precision under zero-shot approach for the DeepSeek model, meaning the models occasionally encounter challenges in making correct predictions. It is important to note that the Llama model consistently balanced

precision and recall. Meanwhile, for the Facebook dataset, almost all the models have high precision and recall, while Gemini-1.5-Pro exhibited the best results, even with minimal to no examples.

5.2 With Fine-Tuning

As we mentioned, our dataset splits, such that a training set has an allocation of 80%, validation comprises 10%, and test contains 10% as well. We fine-tuned **Meta-Llama-3-8B-Instruct**, **Meta-Llama-3.1-8B-Instruct**, and **DeepSeek-R1-Distill-Llama-8B**, for 2 epochs on all three datasets separately. The reason behind fine-tuning only these models is that they are open-source and require no cost to fine-tune or do the inference, as long as we have enough computational power. All the models demonstrated marked improvements with a good balance between precision and recall, likely due to fine-tuning, which allowed them to understand the task-specific patterns and reduce both the false positives and false negatives. For training and inference, all the models were provided with the same prompt as the zero-shot setting, except that labels were injected in the prompt for the training set.

5.2.1 Gr-Qc

All three models achieved an accuracy of over 95%, while Llama-3.1 was slightly ahead, compared to Llama-3 and DeepSeek, at 95.75%, as detailed in Table 3. It also demonstrated a higher precision (97.57%), showing a slight bias towards avoiding false positives. Meanwhile, Llama-3 attained an accuracy of 95.72%, showing nearly identical performance to its updated counterpart. This model also maintained a tighter balance between precision (96.97%) and recall (94.44%), suggesting a more balanced strategy that avoids an overly aggressive nature in predicting positive links, while not being too cautious either. Lastly, DeepSeek with an accuracy of 95.34%, had attained even tighter balance between precision (96.03%) and recall (94.65%), again indicating even more balanced learning of link patterns.

5.2.2 Hep-Th

Fine-tuning on the Hep-Th dataset showed a similar trend (see Table 4), as Gr-Qc, with respect to overall performance. While Llama-3.1 again outperformed both Llama-3 and DeepSeek, attaining an accuracy of 95.57%, whereas Llama-3 of 95.51% and DeepSeek of 95.23%. Interestingly, with regard to precision and recall, the results for this dataset showed completely opposite results compared to Gr-Qc. For this dataset, Llama-3.1, while having the minimum difference between its precision and recall for Gr-Qc, managed to get the tightest balance between the precision (97.04%) and recall (93.96%). With a slightly lower recall in case of Llama-3 and DeepSeek, there were still managed to have a fair balance between their precision and recall, suggesting a more balanced learning again.

5.2.3 Facebook

As evidenced by the data in Table 5, the most striking results came from the Facebook dataset, where all three models performed nearly identically and perfectly. Out of all

the models, the most remarkable result was achieved by Llama-3, achieving nearly perfect 100% accuracy, precision, and recall on the test set with only 2 false-positives and 2 false-negatives out of over 17,000 predictions, achieving an accuracy on par with state-of-the-art results. Llama-3.1 and DeepSeek followed closely at 99.94% and 99.95% respectively, and maintained approximately the same balance between the precision and recall. This indicates that all three models captured all the positive links while making almost no false predictions and drastically improved their performance compared to their zero-shot, one-shot, and few-shot counterparts.

5.3 Baseline Approaches

To provide further context, we evaluated two non-LLM baselines, that is, Versatile Graph Embeddings from Similarity Measures (VERSE) and Fully Connected Neural Network (FCNN). As we mentioned earlier, VERSE is a versatile and efficient graph embedding method designed to preserve vertex-to-vertex similarity measures, which can later be used in downstream tasks such as link prediction and node classification. The FCNN is based on a Multilayer Perceptron architecture, which also serves as a strong baseline as it helps capture non-linear interactions in the input data, that are similarity scores, through multiple dense layers.

Based on the experiments performed by Wu et al. (2022) across various datasets (including the ones we considered for this work) and methods for link prediction, their experiments showed VERSE outperformed all the other methods, such as GCN, SEAL, and LINE, on almost every dataset they considered. Hence, we considered VERSE as the strongest baseline. We tested VERSE on our datasets, using the source code[3] provided by Tsitsulin et al. (2018). We kept the same hyperparameter settings and implemented the link prediction task as described by the authors. The only adjustment made was to consider the training-test split in the same ratio that we used with our LLM-based experiments. Since VERSE generates the embeddings for every node, which we later use to create edge features by fundamentally combining the embeddings of the source node and the target node. For combining, we experimented with different operators such as *Hadamard*, *Concatenation*, *L1*, and *L2*, as explained by the authors in the original work. However, *Hadamard* has notably outperformed the rest of the three, hence, all the results that we reported here use *Hadamard*.

The results show VERSE achieved near-perfect performance across all three datasets, with accuracy, precision, and recall hovering between 99.83% and 99.98%. This approach shows exceptional behavior by predicting all the positive links, while keeping the false positives and false negatives to a minimum, and precision and recall nearly balanced. On both Gr-Qc and Hep-Th datasets, it was able to attain a recall of 100%, suggesting it did not miss any actual positive cases, but with a few false positives for both datasets (see Tables 3 and 4). With regards to the Facebook dataset, it has also outperformed, achieving nearly-perfect accuracy with only 2 false positives and 1 false negative, as detailed in Table 5.

In contrast, the FCNN showed more uneven performance. As illustrated in Tables 3 and 4, for Gr-QC and Hep-Th, the FCNN was able to attain an accuracy of 89.92% and 92.57%, respectively. While it maintained a high and nearly-perfect precision for

[3] Available at: https://github.com/xgfs/verse

both datasets, however, the recall was significantly lower, compared to the precision, which was only 80.03% for Gr-Qc and 85.40% for Hep-Th. This shows that FCNN was more cautious, tending to predict only when it is highly confident, resulting in causing very few false positives, but missing a significant number of actual positive links. For the Facebook dataset (refer Table 5), surprisingly, the model was still able to achieve 99.68% of accuracy, with perfect precision of 100% and a slightly lower recall of 99.36%.

5.4 Comparative Analysis of Learning Strategies

Our experiments show distinct strengths and trade-offs among different approaches, such as zero-shot, one-shot, few-shot, fine-tuning, and baselines for the link prediction task across three different datasets. Notably, the observed trends and relative performances are highly consistent across all three datasets, demonstrating the robustness of these learning strategies irrespective of the nature of the networks, such as social, citation, etc.

The zero-shot, one-shot, and few-shot prompting strategies demonstrate how giving task-specific contextual examples to LLM could lead to improved model performance and balance the precision-recall trade-off. However, noticeably, Llama models showed little to no improvement across all three learning strategies, and this trend has been noticed across all three datasets. Moreover, the Llama-3.1 performance has become worse when tested in a one-shot setting across all datasets. Since Llama models start strong and perform really well, even with no examples compared to other models, hence, providing them with extra examples may not be sufficient. This could be because the model has strong baseline capabilities already, which might be the reason for limiting the gains achievable through extra examples. Unlike Llama, the overall performance of DeepSeek improves significantly under a one-shot setting. However, under a few-shot setting, it shows little to no improvement when provided with multiple examples of each class. Lastly, for Gemini, the performance consistently improved when provided with more and more examples, preventing overly cautious behavior in both models.

Fine-tuning the LLMs further enhances the overall performance and surpasses zero-shot, one-shot, and few-shot approaches across all metrics, with a great balance between precision and recall, while maintaining high accuracy. This is why fine-tuning LLMs represents a better approach for link prediction compared to its prompting-based strategies. However, this incurs a lot of computation cost.

When compared LLM-based approach with VERSE and FCNN, fine-tuned models demonstrate comparable or slightly lower accuracy on datasets like Gr-Qc and Hep-Th. However, experiments also suggest that prompt-based methods are still efficient and could achieve comparable performance to fine-tuning or baseline variants. Although VERSE remains a powerful graph embedding technique with near-perfect scores across all metrics. However, FCNN results suggest the cautious behavior of the model with nearly-perfect precision and significantly lower recall.

6 Ablation Study

In order to validate the contribution of different topological features for our task, we conducted an ablation study by feeding less number of similarity metrics to LLM. We reduced the features from the original top eight to the top six and the top four, selected based on the Pearson Correlation Coefficient with the target label. For this, we compared the performances of Zero-Shot, One-Shot, and Few-Shot approaches across three LLMs (Llama-3, Llama-3.1, and DeepSeek) and all three datasets. Overall, the results indicate that reducing the number of features does not consistently degrade performance, but, in some cases, it even leads to improvement.

The top 6 features are *Jaccard*, *Common Neighbor*, *Sorensen*, *Salton-Cosine*, *Adamic-Adar*, *Node Community*, selected based on the high correlation with the target label and are consistent across all datasets. However, for the top 4 features, we used *Common Neighbor*, *Salton-Cosine*, *Adamic-Adar*, and *Node Community* for Hep-Th and Facebook. While, for Gr-Qc, we replaced *Adamic-Adar* with *Sorensen*.

For the Gr-Qc dataset, as shown in Figure 4, the results consistently favor the top 8 features, with nearly all three models across all three prompting strategies. When we reduced features from top 8 to top 6, and then further to top 4, we observed a gradual decline in the accuracy. This trend confirms that the full set of top 8 features contributes meaningfully to the model's ability to capture the underlying complex patterns in the graph. For instance, Llama-3 and Llama-3.1 showed consistent performance with 91% accuracy most of the time, with a marginal decline (around 1-2%) with fewer features. However, an interesting exception in the Zero-Shot approach with DeepSeek has been observed, where performance improves significantly as we reduced the features from 84% with top 8 features to 93% with top 6 features, demonstrating model-specific sensitivity to feature redundancy.

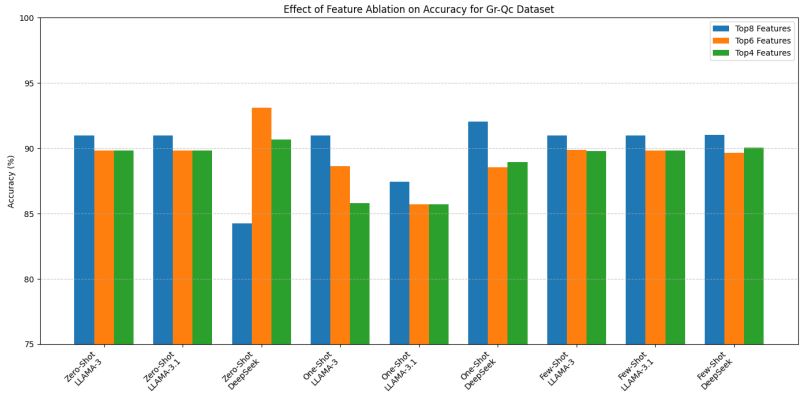

Fig. 4 Effect of Feature Ablation on Accuracy for the Gr-Qc Dataset

For the Hep-Th dataset, the performance across all three feature sets was quite similar, with only a marginal difference for the majority of the models and prompting settings, as shown in Figure 5. In most cases, the top 8 features remain a solid choice,

with a slight improvement in performance with the top 6 features. Similar to the Gr-Qc dataset, a similar exception has been observed for DeepSeek under a zero-shot setting, where the model performed significantly better with few features, attaining an over 10% increase in accuracy with the top 6 and top 4 features. As mentioned earlier, including all the scores could lead to model hallucinations and may reduce generalization, especially with large inputs. A similar trend has also been observed for Llama-3.1 under the One-Shot strategy, where the top 6 features improved the performance gains by almost 5% compared to the actual feature set, but slightly decreased again when we further reduced the features from the top 6 to top 4. It suggests that trimming too much information could also hinder model reasoning.

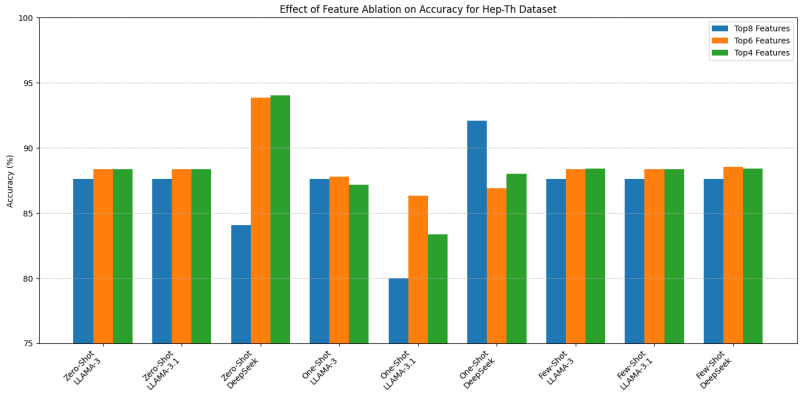

Fig. 5 Effect of Feature Ablation on Accuracy for the Hep-Th Dataset

On the Facebook dataset, the trend diverged slightly where it showed a more mixed pattern. As shown in Figure 6, reducing the number of features does not consistently harm the performance, but also brings performance gains. Similar to other datasets, the performance of DeepSeek under the Zero-Shot setting increased substantially when features were reduced. Similar improvements were observed for the Llama-3.1 and DeepSeek under the One-Shot setting, where the top 6 features yielded the highest accuracy of over 97%.

Overall, the top 8 features were leveraged in our experiments due to their consistent performance. However, the ablation study shows that models are somewhat robust to feature reduction and using a reduced set of features can maintain or even increase (in a few cases) the performance of the approach. Although the decision of reduced features would be heavily dependent on factors such as network topology, the models, and the prompting strategy.

7 Conclusion

In conclusion, this study introduces a novel framework that leverages Large Language Models (LLMs) for link prediction tasks in homogeneous networks, not just limited to social networks, by utilizing similarity scores, even in the absence of any textual

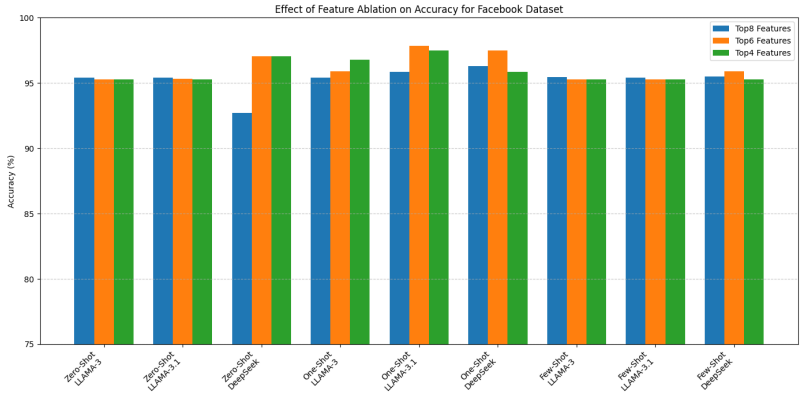

Fig. 6 Effect of Feature Ablation on Accuracy for the Facebook Dataset

information. Most of the considered LLMs performed exceptionally well in most cases, even without any prior training on the considered datasets. However, after fine-tuning, the results improved even further for all the datasets, achieving even up to nearly 100% on the Facebook dataset and above 95% for the rest of the datasets. Therefore, LLMs have shown a great tendency and understanding to solve the link prediction problem even without the presence of any textual data in the raw dataset. Overall, our framework provides valuable insights into the application of LLMs for link prediction and their capabilities in the constantly evolving social networks.

8 Future Work

Future research could be aimed at exploring the link prediction problem in the constantly evolving networks, i.e., Dynamic Networks. It could also be directed toward exploring the potential of fine-tuning the LLMs on the different types of link prediction datasets to develop a dedicated fine-tuned model to handle various types of graphs, such as heterogeneous graphs, homogeneous graphs, and knowledge graphs, with the same effectiveness. Additionally, the approach that we introduced could be extended to other fields such as biological networks, where it can be used to predict the gene-cell interactions.

9 Statements and Declarations

Funding The authors received no specific grant from funding agencies or other organizations for the research, authorship, or publications of this work.

Conflict of interest: The authors declare that they have no financial or non-financial interests that could appear to influence the content or outcomes of this work.

Ethics approval: The study did not involve any experiments with human participants or animals. Hence, ethics approval was not required.

Consent to participate and publish: Not Applicable.

Data and Code availability: We used publicly available datasets extracted from the SNAP repository (http://snap.stanford.edu/data). Specifically, we utilized the following datasets: Gr-Qc (https://snap.stanford.edu/data/ca-GrQc.html), Hep-Th(https://snap.stanford.edu/data/ca-HepTh.html), and Facebook(https://snap.stanford.edu/data/egonets-Facebook.html) datasets. The source code used in this study can be made available by the corresponding author upon request.

Author Contributions: All authors contributed equally to the conception, design, analysis, and writing of this study. All authors read and approved the final version of the manuscript.

References

Bi B, Liu S, Wang Y, et al (2024) LPNL: Scalable link prediction with large language models. In: Ku LW, Martins A, Srikumar V (eds) Findings of the Association for Computational Linguistics: ACL 2024. Association for Computational Linguistics, Bangkok, Thailand, pp 3615–3625, https://doi.org/10.18653/v1/2024.findings-acl.215, URL https://aclanthology.org/2024.findings-acl.215/

Chen Y, Shen Y (2025) Temporal knowledge graph link prediction using synergized large language models and temporal knowledge graphs. In: Zhang H, Li X, Hao T, et al (eds) Neural Computing for Advanced Applications. Springer Nature Singapore, Singapore, pp 33–45, https://doi.org/https://doi.org/10.1007/978-981-97-7007-6_3

Daud NN, Ab Hamid SH, Saadoon M, et al (2020) Applications of link prediction in social networks: A review. Journal of Network and Computer Applications 166:102716. https://doi.org/https://doi.org/10.1016/j.jnca.2020.102716, URL https://www.sciencedirect.com/science/article/pii/S1084804520301909

DeepSeek-AI, Guo D, Yang D, et al (2025) Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. URL https://arxiv.org/abs/2501.12948, arXiv:2501.12948

Dubey A, Jauhri A, Pandey A, et al (2024) The llama 3 herd of models. URL https://arxiv.org/abs/2407.21783, arXiv:2407.21783

He Z, Zhu J, Qian S, et al (2024) Linkgpt: Teaching large language models to predict missing links. URL https://arxiv.org/abs/2406.04640, arXiv:2406.04640

Khanna S, Bhattacharyya S, Ghosh S, et al (2024) Link prediction for social networks using representation learning and heuristic-based features. URL https://arxiv.org/abs/2403.08613, arXiv:2403.08613

Le T, Le N, Le B (2023) Knowledge graph embedding by relational rotation and complex convolution for link prediction. Expert Systems with Applications 214:119122. https://doi.org/https://doi.org/10.1016/j.eswa.2022.119122, URL https://www.sciencedirect.com/science/article/pii/S0957417422021406

Leskovec J, Kleinberg J, Faloutsos C (2007) Graph evolution: Densification and shrinking diameters. ACM Trans Knowl Discov Data 1(1):2–es. https://doi.org/10.1145/1217299.1217301, URL https://doi.org/10.1145/1217299.1217301

Lin Q, Liu J, Xu F, et al (2022) Incorporating context graph with logical reasoning for inductive relation prediction. In: Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval. Association for Computing Machinery, New York, NY, USA, SIGIR '22, p 893–903, https://doi.org/10.1145/3477495.3531996, URL https://doi.org/10.1145/3477495.3531996

McAuley J, Leskovec J (2012) Learning to discover social circles in ego networks. In: Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 1. Curran Associates Inc., Red Hook, NY, USA, NIPS'12, p 539–547

Nie L, Song X, Chua TS (2016) Learning from multiple social networks. Synthesis Lectures on Information Concepts, Retrieval, and Services 8:1–118. https://doi.org/10.2200/S00714ED1V01Y201603ICR048

Shu D, Chen T, Jin M, et al (2024) Knowledge graph large language model (kg-llm) for link prediction. URL https://arxiv.org/abs/2403.07311, arXiv:2403.07311

Tang J, Qu M, Wang M, et al (2015) Line: Large-scale information network embedding. In: Proceedings of the 24th International Conference on World Wide Web, WWW '15, vol 14. International World Wide Web Conferences Steering Committee, p 1067–1077, https://doi.org/10.1145/2736277.2741093, URL http://dx.doi.org/10.1145/2736277.2741093

Team G, Georgiev P, Lei VI, et al (2024) Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context. URL https://arxiv.org/abs/2403.05530, arXiv:2403.05530

Tsitsulin A, Mottin D, Karras P, et al (2018) Verse: Versatile graph embeddings from similarity measures. In: Proceedings of the 2018 World Wide Web Conference. International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, CHE, WWW '18, p 539–548, https://doi.org/10.1145/3178876.3186120, URL https://doi.org/10.1145/3178876.3186120

Wu H, Song C, Ge Y, et al (2022) Link prediction on complex networks: An experimental survey. Data Science and Engineering 7(3):253–278. https://doi.org/10.1007/s41019-022-00188-2, URL https://doi.org/10.1007/s41019-022-00188-2

Zhang M, Chen Y (2018) Link prediction based on graph neural networks. In: Proceedings of the 32nd International Conference on Neural Information Processing Systems. Curran Associates Inc., Red Hook, NY, USA, NIPS'18, p 5171–5181