Leveraging LLM Reasoning for Graph Link Prediction

Andrew Sun
Computer Science
Rice University

Nathan Calzat
Computer Science
Rice University

Henry Xue
Computer Science
Rice University

Abstract

This project explores the integration of large language model (LLM)-based text embeddings with graph-based link prediction to enhance the discovery of relationships between nodes in large-scale web graphs. With recent developments in State-of-the-art LLMs equipped with reasoning capabilities, we hypothesize that rich latent representations of these models contains important semantic information that could be used to improve Graph-based techniques. Leveraging the Common Crawl dataset, we construct a graph where nodes represent web pages and edges represent hyperlinks. We then extract text embeddings from the hidden states of reasoning-equipped LLMs and benchmark their performance against embeddings generated by Sentence-BERT (SBERT). By integrating these semantic embeddings with topological features via a GraphSAGE GNN architecture, our model effectively predicts potential hyperlinks. Systematic evaluations demonstrate an improvement in link prediction performance with LLM-derived embeddings, underscoring the potential of LLM-derived semantic information in navigating large-scale web networks. Our findings further corroborate prior LLM probing work, showing that mid-to-upper transformer layers encode the most discriminative semantic information for downstream tasks.

1 The Problem

The exponential growth of web content has led to the creation of massive, complex networks of information such as the World Wide Web, social media platforms, and citation networks. In these systems, understanding and predicting the relationships between entities—whether they are web pages, users, or documents—is a fundamental task with wide-ranging applications, including recommendation systems, information retrieval, knowledge graph completion, and fraud detection.

A common challenge in these areas is the link prediction problem: given a graph representing entities and their relationships, can we accurately predict which new links are likely to form, or which missing links exist but have not yet been observed? Solving this problem allows us to recommend new friends in social networks, suggest relevant articles to readers, or uncover hidden associations in scientific literature.

1.1 Link Prediction

Traditional link prediction methods primarily exploit the structural properties of the graph, such as node degree, shortest paths, or the number of shared neighbors. While these approaches are computationally efficient and often effective, they are inherently limited: they treat nodes as abstract points, ignoring the rich, descriptive information that often accompanies them. For example, a web page is not just a node in a hyperlink graph—it contains text, metadata, and context that can reveal its topic, intent, and relevance to other pages.

This limitation becomes especially pronounced in large, heterogeneous graphs derived from real-world data, where structural signals may be sparse, noisy, or ambiguous. For instance, two web pages might not share many direct links, but their textual content could indicate a strong semantic relationship (e.g., both discuss the same scientific concept or news event). Conversely, some links may exist for reasons unrelated to content similarity (e.g., advertisements or navigation menus), introducing noise into purely structure-based predictions.

1.2 Large Language Models

Recent advances in natural language processing, particularly the development of large language models (LLMs), have made it possible to extract high-quality, semantically meaningful embeddings from raw text. These embeddings capture subtle relationships and contextual information that are inaccessible to traditional graph algorithms. By integrating LLM-based text embeddings into the link prediction pipeline, we can potentially bridge the gap between structural and semantic analysis, leading to more accurate and robust predictions.

The core problem addressed in this project are twofold:

1. How can we effectively combine large-scale, LLM-derived text embeddings with graph structural features to enhance link prediction in web-scale graphs?
2. What is the quantitative impact of incorporating semantic information from web page content, compared to relying solely on graph topology?

To answer these questions, we leverage the Common Crawl dataset—a vast, open-source repository of web data—to construct a graph where nodes represent web pages and edges represent hyperlinks. We extract and embed the textual content of each page and systematically evaluate the performance of link prediction models that utilize both structural and semantic features. By doing so, we aim to demonstrate the practical value of LLM-based embeddings in real-world graph mining tasks, and to provide insights into the interplay between text and structure in large-scale networks.

2 Background

2.1 Techniques in Link Prediction

Link prediction is a core problem in network analysis, where the goal is to infer missing or future connections between nodes in a graph. Formally defined, this means given a graph $G = (V, E)$, the task is to rank pairs $(u, v) \notin E$ by the likelihood that an edge should (or will) exist. Classical heuristics use topology only—e.g., Common Neighbours or Jaccard—and are fast but oblivious to node content.

More recently, machine learning approaches have been developed to learn node representations that capture structural patterns. Among these, Graph Neural Networks (GNNs) have emerged as a powerful class of models. GraphSAGE is a popular GNN architecture designed for inductive representation learning on large graphs. [Hamilton et al., 2017] Unlike transductive methods that require the entire graph during training, GraphSAGE learns aggregation functions that can generate embeddings for unseen nodes by sampling and aggregating features from local neighborhoods. This makes it scalable and well-suited for dynamic or web-scale graphs. In the context of link prediction, node embeddings generated by GraphSAGE can be used to compute similarity scores between node pairs, serving as features for predicting the existence of links.

2.2 Large Language Models, Text Embeddings, and Semantic Information

Large Language Models are deep neural networks trained on massive text corpora to predict the next word in a sequence or to fill in masked tokens. These models use transformer architectures, which rely on self-attention mechanisms to capture long-range dependencies and contextual relationships in text. As text is processed through multiple transformer layers, the model's hidden states encode increasingly abstract and semantic information about the input. The internal hidden states can be extracted as dense vector embeddings, which can loosely represent the meaning of words, sentences, or entire documents in a high-dimensional space. These embeddings capture subtle semantic relationships, such as synonymy, topical similarity, and contextual relevance, making them valuable features for downstream tasks such as link prediction.

Many real-world graphs, such as web or citation networks, are accompanied by rich textual information. Incorporating this semantic content can provide additional context for link prediction, especially when structural signals are weak or ambiguous.

In this project, we specifically utilize the DeepSeek-R1-Distill-Llama-8b model,[DeepSeek-AI et al., 2025]. This is an open-source reasoning-capable LLM built by distilling outputs from the DeepSeek-R1 (which has been fine-tuned and RLed for strong reasoning and language understanding capabilities) model onto an 8b Llama-3 model [Meta AI, 2024]. We use this model to generate high-quality semantic embeddings for web page content.

SBERT [Reimers and Gurevych, 2019] is a modification of the BERT architecture specifically designed to generate semantically meaningful sentence embeddings. While vanilla BERT produces contextualized token embeddings, SBERT uses a siamese network structure and is fine-tuned on sentence-pair tasks (such as semantic textual similarity) to produce fixed-size embeddings for entire sentences or paragraphs. In our work, we use the all-MiniLM-L6-v2 model [Reimers and Gurevych, 2021], a lightweight and efficient SBERT variant that produces high-quality sentence embeddings suitable for large-scale applications. These embeddings can be efficiently compared using cosine similarity, making SBERT particularly effective for tasks that require measuring semantic similarity between texts.

3 Methodology

We build an end-to-end pipeline that (1) harvests web pages from the Common Crawl corpus, (2) extracts high-quality text, (3) constructs a hyperlink graph, (4) generates two families of text embeddings (SBERT and DeepSeek), (5) feeds these embeddings into GraphSAGE for inductive link prediction, and (6) evaluates performance with rigorous negative-sampling and cross-validation. We also evaluate the effectiveness of these embeddings using a logistic regression approach for link prediction, as a baseline.

3.1 Graph Construction

To construct our web graph, we utilized the Common Crawl dataset, focusing on a specific crawl segment to ensure manageability. We extracted raw HTML content from selected web pages and employed tools like Trafilatura to clean and extract the main textual content. Simultaneously, we parsed hyperlinks to identify connections between pages. By representing each web page as a node and each hyperlink as a directed edge, we built a graph structure suitable for our analysis. The resulting directed graph had 9202 nodes and 92834 edges.

3.2 Text Embedding Generation

We embed each page's cleaned text with all-MiniLM-L6-v2—a 384-dimensional SBERT checkpoint fine-tuned for semantic similarity [Reimers and Gurevych, 2019]. Inference runs in batches of 64 on single-GPU mixed precision, producing a dense matrix

$$E_S \in \mathbb{R}^{|V| \times 384},$$

where $|V|$ is the number of nodes in the graph. For our LLM embeddings, because transformer encoders accumulate linguistic abstractions layer-by-layer, we treat the hidden states of the LLM as a stack of candidate embeddings and benchmark layers 4, 12, 20, and 28 independently. For each input we save these layers, apply mean pooling over tokens, and save

$$E_{L_i} \in \mathbb{R}^{|V| \times d},$$

where $d = 4096$ is the DeepSeek-R1-Distill-Llama-8B hidden size. Prior probing studies on BERT-family models show that syntactic cues dominate lower layers, while higher layers gradually shift toward semantic and task-specific representations.[Ethayarajh, 2019] [Vulić et al., 2020] Evaluating each layer therefore lets us pinpoint where the most actionable web-page semantics reside for link prediction.

To further evaluate the usefulness of Language Model-derived embeddings, we also train a learnable embedding layer inside the GraphSAGE model. This layer initializes a trainable d-dimensional vector for every node and updates it jointly with the GNN parameters d-dimensional vector for every node and updates it jointly with the GNN parameters.

3.3 Link-Prediction Model

We adopt a two-layer GraphSAGE encoder with hidden dimension 128 and ReLU activations [Hamilton et al., 2017]. Mean aggregation is used because preliminary tests showed no gain from LSTM or max pooling. Drop-out of 0.3 is applied after each layer.

An MLP with three fully-connected layers (128$\rightarrow$128$\rightarrow$1) scores edges via element-wise product of node embeddings, followed by a sigmoid. The encoder and MLP both use the Adam optimizer.

Random negatives can be trivial when most non-edges have zero common neighbours. We therefore generate hard negatives by picking two-hop nodes with high common-neighbour counts yet no observed edge, inspired by [Zhang et al., 2025]. One hard negative is paired with every positive edge in train/val/test.

We also evaluate using the embedding space with a standard ℓ_2-regularised logistic-regression classifier on the same positive/negative splits used by GraphSAGE. No message passing is performed; the model therefore relies solely on the raw embedding space to infer link likelihood.

3.4 Training and Evaluation

We split the dataset using RandomLinkSplit from PyTorch Geometric, with a 80/10/10 split. To avoid label leakage, validation and test positives are node-disjoint from training positives. Each model trains for 50epochs with mini-batch size 2,048. For evaluation, we compute ROC-AUC and Average Precision (AP). Hits@K is also reported. Experiments were ran on a single NVIDIA A6000 GPU.

4 Results and Discussion

Key Observations of Figure/Table 1

Table 1: Final link–prediction metrics of GraphSAGE with different initial node embeddings. **Bold** indicates the best AUC.

Embedding Type	AUC	Precision	Hits@10
LearnableEmb	0.9350	0.9229	1.0000
SBERT	0.8825	0.9221	0.2000
Layer_04	0.9001	0.9218	1.0000
Layer_08	0.8943	0.9220	1.0000
Layer_12	0.9174	0.9217	1.0000
Layer_16	0.9123	0.9220	1.0000
Layer_20	**0.9810**	0.9223	1.0000
Layer_24	0.9618	0.9222	1.0000
Layer_28	0.9516	0.9224	1.0000

- **Mid–upper transformer layers dominate**. Layer 20 of DeepSeek achieves the top AUC (0.981) while retaining perfect Hits@10 and the highest observed precision. Layers 24 and 28 follow closely, whereas early layers 4–12 trail. Our experiments corroborate previous findings [Ethayarajh, 2019] [Vulić et al., 2020]: mid-to-upper-middle layers deliver the best performance, whereas the final generative layers add little beyond what GraphSAGE already captures, likely because they over-specialise to the language-modeling objective. .
- **Learnable structural baseline is competitive but not optimal**. A purely trainable embedding tied to GraphSAGE attains 0.935 AUC, outperforming SBERT and shallow DeepSeek layers, yet remains four to five points below Layer 20, quantifying the added value of external semantics.

- **SBERT lags in AUC but matches precision**. Despite a high classification precision, SBERT's Hits@10 plummets to 0.20, indicating that many true positives are ranked lower than the top 10 when only SBERT embeddings are used.

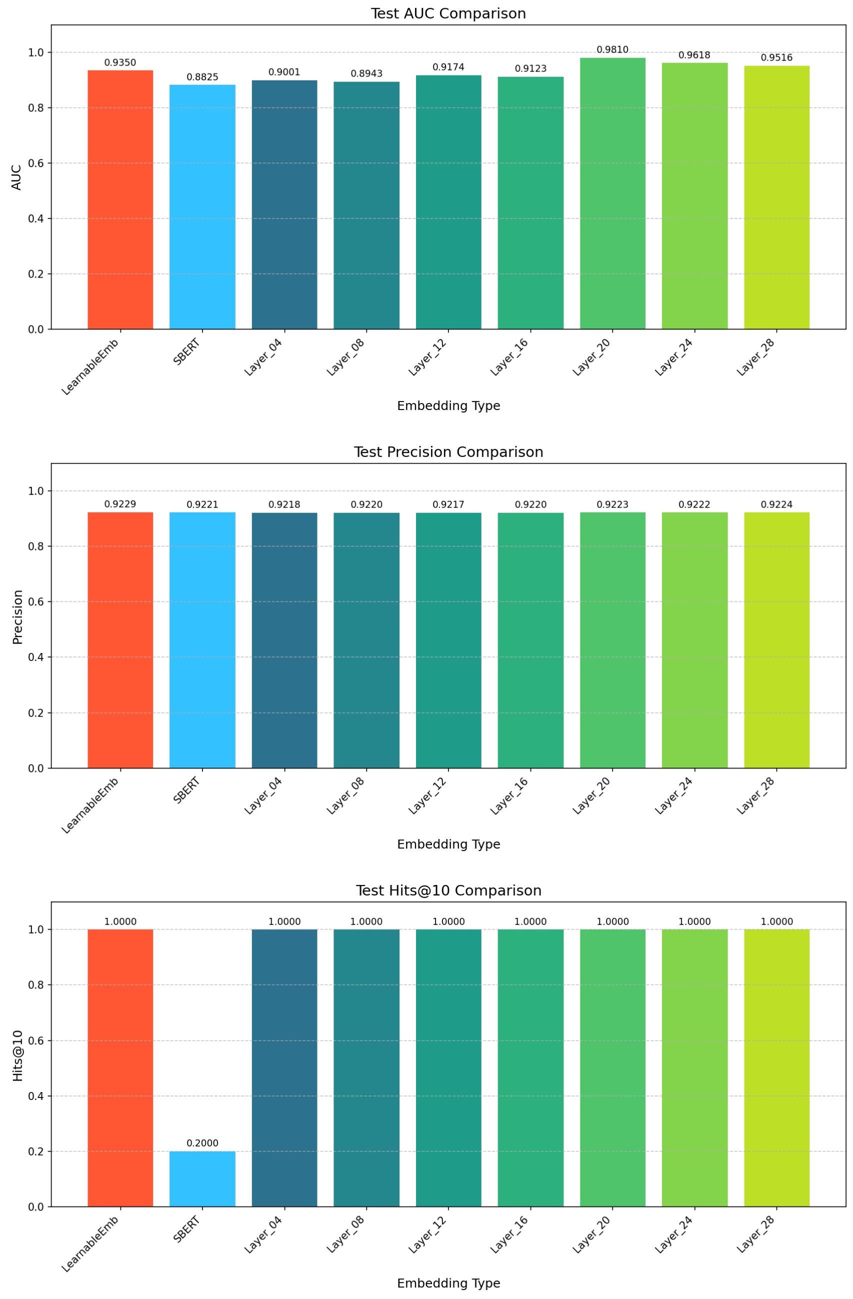

Figure 1: Test–set performance for each embedding type. Labels from left to right represent Learnable Embeddings, SBERT Embeddings, and DeepSeek layers 4-28.

The training curves in Fig. 2 confirm these trends: models initialised with high–quality DeepSeek layers converge rapidly (within ≈10 epochs) and stabilise at a higher validation plateau than SBERT or shallow layers.

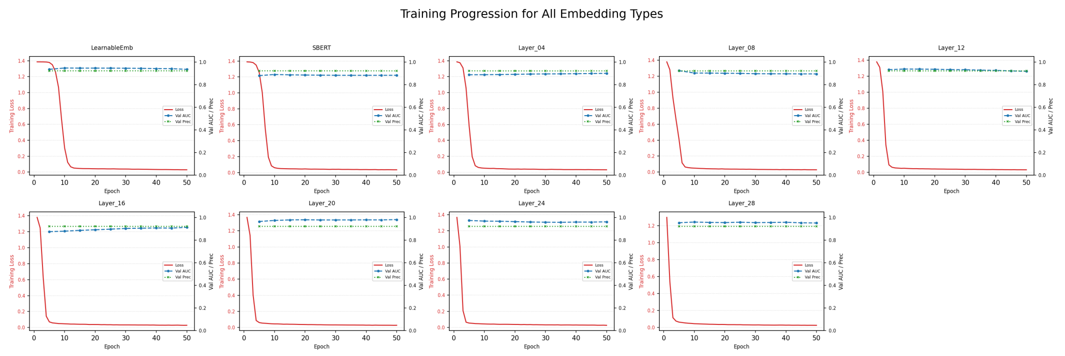

Figure 2: Training loss (red) and validation metrics (AUC in blue, precision in green) across 50 epochs for every embedding configuration.

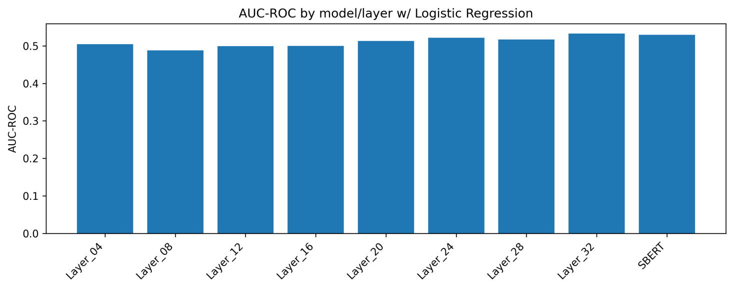

Figure 3: Baseline logistic-regression AUC across embedding layers.

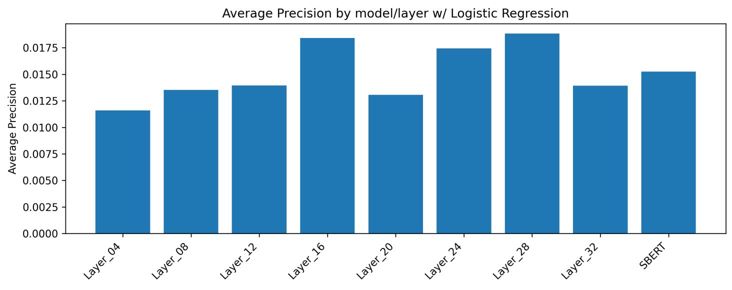

Figure 4: Baseline logistic-regression average precision across embedding layers (note the small scale).

Figures 3–4 show that the logistic baseline plateaus around 0.54 AUC and 1.9×10^{-2} average precision at best (DeepSeek layer 32), only marginally above random guessing. In stark contrast, our GraphSAGE + DeepSeek-L20 system reaches 0.981 AUC and 0.922 precision, improving absolute AUC by +44 points and precision by ≈ 0.90. The gap confirms that (i) richer neighbourhood aggregation and (ii) fine-tuned mid-layer semantics are both critical for high-quality link prediction, whereas a shallow linear model cannot capitalise on the latent information present in isolated embeddings.

5 Conclusions, Challenges, and Lessons Learned

Conclusions. Our experiments show that enriching GraphSAGE with mid-layer LLM embeddings (DeepSeek-L20) yields the strongest link-prediction performance, surpassing both a learnable structural baseline and SBERT features by $+\mathbf{0.046}$–$+\mathbf{0.099}$ AUC. The gain confirms that web-page semantics captured by transformer layers can complement topological signals in large-scale graphs.

Moreover, the layer sweep corroborates prior probing studies: contextual representations stabilise in upper-middle layers, whereas the final generative layers provide diminishing returns.

Challenge #1: negative–sampling bias. Early runs with *uniform random* negatives produced deceptively high scores (AUC≈ 0.99, precision≈ 0.95) no matter which embedding we supplied. Because most randomly chosen non-edges are trivially separable from positives, the task becomes too easy. Only after introducing **hard two-hop negatives** did the metric space meaningfully differentiate embedding quality. Lesson: *evaluation must include plausible but unobserved edges*; otherwise, improvements from sophisticated models can be masked.

Challenge #2: computational cost of large LLMs. Even an 8-billion–parameter model required multiple hours of multi-GPU inference to embed merely 9k pages. Scaling to the full Common Crawl (billions of pages) with larger checkpoints (e.g., 70B Llama 3) would be prohibitive without massive distributed infrastructure or aggressive distillation/quantisation. Future work should investigate (i) lightweight instruction-tuned encoders, (ii) on-the-fly embedding caching, and (iii) mixed-precision/batch-sharded serving pipelines.

Additional lessons.

- **Simple baselines remain informative.** The logistic-regression baseline, while weak (AUC≈ 0.54), highlighted that raw embeddings alone cannot exploit graph structure; neighbourhood aggregation is essential.
- **Layer-wise analysis guides model choice.** Instead of defaulting to the final hidden state, selecting an information-rich mid-layer led to a $\approx 7\%$ relative AUC boost over layer 4.

Future directions. Beyond scaling and better negatives, we plan to (i) co-train/fine-tune the encoder and GNN end-to-end, (ii) incorporate temporal crawl information to predict future links for online tasks, and (iii) explore retrieval-augmented message passing that mixes text and structural context at inference time.

References

DeepSeek-AI. Deepseek-r1-distill-llama-8b. `https://huggingface.co/deepseek-ai/DeepSeek-R1-Distill-Llama-8B`, 2025. Available at `https://huggingface.co/deepseek-ai/DeepSeek-R1-Distill-Llama-8B`.

DeepSeek-AI, Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, Aixin Liu, Bing Xue, Bingxuan Wang, Bochao Wu, Bei Feng, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, Damai Dai, Deli Chen, Dongjie Ji, Erhang Li, Fangyun Lin, Fucong Dai, Fuli Luo, Guangbo Hao, Guanting Chen, Guowei Li, H. Zhang, Han Bao, Hanwei Xu, Haocheng Wang, Honghui Ding, Huajian Xin, Huazuo Gao, Hui Qu, Hui Li, Jianzhong Guo, Jiashi Li, Jiawei Wang, Jingchang Chen, Jingyang Yuan, Junjie Qiu, Junlong Li, J. L. Cai, Jiaqi Ni, Jian Liang, Jin Chen, Kai Dong, Kai Hu, Kaige Gao, Kang Guan, Kexin Huang, Kuai Yu, Lean Wang, Lecong Zhang, Liang Zhao, Litong Wang, Liyue Zhang, Lei Xu, Leyi Xia, Mingchuan Zhang, Minghua Zhang, Minghui Tang, Meng Li, Miaojun Wang, Mingming Li, Ning Tian, Panpan Huang, Peng Zhang, Qiancheng Wang, Qinyu Chen, Qiushi Du, Ruiqi Ge, Ruisong Zhang, Ruizhe Pan, Runji Wang, R. J. Chen, R. L. Jin, Ruyi Chen, Shanghao Lu, Shangyan Zhou, Shanhuang Chen, Shengfeng Ye, Shiyu Wang, Shuiping Yu, Shunfeng Zhou, Shuting Pan, S. S. Li, Shuang Zhou, Shaoqing Wu, Shengfeng Ye, Tao Yun, Tian Pei, Tianyu Sun, T. Wang, Wangding Zeng, Wanjia Zhao, Wen Liu, Wenfeng Liang, Wenjun Gao, Wenqin Yu, Wentao Zhang, W. L. Xiao, Wei An, Xiaodong Liu, Xiaohan Wang, Xiaokang Chen, Xiaotao Nie, Xin Cheng, Xin Liu, Xin Xie, Xingchao Liu, Xinyu Yang, Xinyuan Li, Xuecheng Su, Xuheng Lin, X. Q. Li, Xiangyue Jin, Xiaojin Shen, Xiaosha Chen, Xiaowen Sun, Xiaoxiang Wang, Xinnan Song, Xinyi Zhou, Xianzu Wang, Xinxia Shan, Y. K. Li, Y. Q. Wang, Y. X. Wei, Yang Zhang, Yanhong Xu, Yao Li, Yao Zhao, Yaofeng Sun, Yaohui Wang, Yi Yu, Yichao Zhang, Yifan Shi, Yiliang Xiong, Ying He, Yishi Piao, Yisong Wang, Yixuan Tan, Yiyang Ma, Yiyuan Liu, Yongqiang Guo, Yuan Ou, Yuduan Wang, Yue Gong, Yuheng Zou, Yujia He, Yunfan Xiong, Yuxiang Luo, Yuxiang You, Yuxuan Liu, Yuyang Zhou, Y. X. Zhu, Yanhong

Xu, Yanping Huang, Yaohui Li, Yi Zheng, Yuchen Zhu, Yunxian Ma, Ying Tang, Yukun Zha, Yuting Yan, Z. Z. Ren, Zehui Ren, Zhangli Sha, Zhe Fu, Zhean Xu, Zhenda Xie, Zhengyan Zhang, Zhewen Hao, Zhicheng Ma, Zhigang Yan, Zhiyu Wu, Zihui Gu, Zijia Zhu, Zijun Liu, Zilin Li, Ziwei Xie, Ziyang Song, Zizheng Pan, Zhen Huang, Zhipeng Xu, Zhongyu Zhang, and Zhen Zhang. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning, 2025. URL `https://arxiv.org/abs/2501.12948`.

Kawin Ethayarajh. How contextual are contextualized word representations? comparing the geometry of bert, elmo, and gpt-2 embeddings, 2019. URL `https://arxiv.org/abs/1909.00512`.

William L. Hamilton, Rex Ying, and Jure Leskovec. Inductive representation learning on large graphs. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 30, 2017. URL `https://arxiv.org/abs/1706.02216`.

Meta AI. Meta-llama-3-8b. `https://huggingface.co/meta-llama/Meta-Llama-3-8B`, 2024. Accessed: 2025-05-06.

Nils Reimers and Iryna Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084*, 2019. URL `https://arxiv.org/abs/1908.10084`.

Nils Reimers and Iryna Gurevych. all-minilm-l6-v2. `https://huggingface.co/sentence-transformers/all-MiniLM-L6-v2`, 2021. Accessed: 2025-05-06.

Ivan Vulić, Edoardo Maria Ponti, Robert Litschko, Goran Glavaš, and Anna Korhonen. Probing pretrained language models for lexical semantics, 2020. URL `https://arxiv.org/abs/2010.05731`.

Kaiyu Zhang, Guoming Sang, Junkai Cheng, Zhi Liu, and Yi-Jia Zhang. Negative sampling strategy based on multi-hop neighbors for graph representation learning. *Expert Systems with Applications*, 263:125688, 2025. doi: 10.1016/j.eswa.2024.125688. URL `https://doi.org/10.1016/j.eswa.2024.125688`.