Semantic Priors for Drug–Drug Interaction Prediction Using Compact Graph Encoders

Annika Viswesh
Department of Computing and Mathematical Sciences
California Institute of Technology
Pasadena, CA 91125
aviswesh@caltech.edu

Abstract

Accurate prediction of drug–drug interactions (DDI) is critical to patient safety. Graph-based models show promise for DDI link prediction, with prior work exploring both structure-only encoders and those augmented with semantic information. However, there is limited evaluation of whether semantic priors enable smaller encoders to reach the performance of larger models. We investigate whether domain-aware biomedical text embeddings, that are task-optimized and used for node initialization, enable compact encoders to achieve predictive accuracy comparable to that of much larger graph-based models. We precompute drug node embeddings by encoding DrugBank text with a frozen SciBERT, refine these embeddings with a small contrastive MLP, and use the resulting task-oriented embeddings to initialize node representations in a Graph Neural Network. During training, the model learns structural information on top of this semantic prior, with node embeddings regularized to remain close to their initial values. On the ogbl-ddi benchmark, our model attains test performance approaching the best published structure-only result, while using only 0.52% of the parameters (5.08 million vs. 976 million). Among published models on ogbl-ddi, our approach lies on the Pareto frontier of performance versus size and outperforms 94% of the existing entries. These findings suggest that semantic priors from pretrained scientific language models with task-optimized refinement can support resource-efficient, competitive encoders for DDI link prediction.

1 Introduction

The simultaneous administration of multiple drugs, commonly known as polypharmacy, is widespread in clinical practice, especially among older adults [1]. Large-scale studies estimate that nearly 40% of adults aged 65 years and older in developed countries are prescribed five or more medications concurrently, with prevalence continuing to rise each year [2, 3, 4]. Polypharmacy increases the risk of drug-drug interactions (DDIs), which can alter efficacy or cause severe adverse drug reactions (ADRs) [5]. As a consequence, ADRs contribute to preventable hospitalizations, higher healthcare costs, and significant patient morbidity [6, 7].

Identifying unknown DDIs in advance remains a fundamental challenge for drug development, clinical care, and reducing healthcare costs [7]. Predicting novel drug–drug interactions has traditionally relied on experimental and observational methods, including in vitro and in vivo studies and analyses of prior clinical data [8, 9]. These approaches are costly and do not scale [8, 10]. The number of pairs grows quadratically with the number of drugs, and higher-order combinations grow combinatorially [11]. Even for 1,000 drugs, exhaustive pairwise screening would require 499,500 tests. Moreover,

39th Conference on Neural Information Processing Systems (NeurIPS 2025) Workshop: New Perspectives in Graph Machine Learning.

since many DDI databases are incomplete and slow to incorporate new drugs and indications, they constrain the detection of novel interactions and limit the ability to integrate pharmacokinetic and pharmacodynamic variability [12, 13, 14, 15]. This necessitates the use of computational methods to predict unknown interactions from partially observed data [9, 10, 16].

In many studies, DDI identification is treated as a link prediction problem on drug-drug interaction network graphs [9, 10, 17]. Graph Neural Networks (GNNs) are especially suited for this problem. GNNs aggregate both local and global information, so that nodes with similar structural roles learn similar embeddings. The drug network, which serves as input to the GNN, represents drugs as nodes and observed interactions as edges. GNN-based models have shown strong results on biomedical prediction tasks, including DDIs, drug–target interaction, and drug repurposing, and they perform well on benchmarks such as ogbl-ddi [16, 18].

A common approach for GNN-based link prediction uses only graph topology and topology-derived features [19, 20, 21, 22]. Other methods incorporate external domain knowledge from auxiliary sources, such as knowledge graphs or embeddings produced by pretrained language models [23, 24, 25]. Although these methods can improve performance, they often incur substantial computational and memory costs, either from large model sizes or from expensive pretraining and inference, thereby limiting training and deployment in settings with limited computational capacity [26, 27]. For clinical applications, compact encoders are especially important, as they lower the computational burden, reduce memory requirements, and decrease energy and serving costs [28, 29, 30, 31]. These constraints motivate thorough, parameter-matched evaluations of whether augmenting structural models with semantic priors can yield compact encoders without sacrificing performance. To the best of our knowledge, prior work does not report parameter-matched head-to-head comparisons between structure-only models and models initialized with pretrained scientific language model embeddings with task-optimized refinement for DDI link prediction.

The main contributions of our paper are as follows.

1. **Preoptimized embeddings for training a compact GNN.** We start from frozen embeddings of a pretrained scientific language model, SciBERT [32], and refine them with a small two-layer contrastive MLP. These preoptimized embeddings initialize the GNN for DDI link prediction, keeping the GNN's parameter count unchanged. During training, structural features are learned on top of task-optimized semantic priors, while a proximal regularizer maintains node states near their initialization. This results in a compact model that performs competitively with the best, significantly larger prior models.
2. **Pareto Frontier of Size Versus Performance.** Our method achieves performance within 2.19% of the leading structure-only model on the ogbl-ddi benchmark while using only 0.52% of its parameters. On the ogbl-ddi public leaderboard [33], our model ranks third out of 28, and achieves performance within 0.24% of the second-ranked entry while using less than half the parameters. Compared to the third-ranked model, ours achieves 2.1% higher performance with a smaller parameter count. Among published models, our approach lies on the Pareto frontier of performance versus size, outperforming 94% of all published entries.
3. **Resource-Efficient and Clinically Practical.** Our approach yields compact models that support efficient training and inference under computational and memory constraints while maintaining near-SoTA performance on DDI prediction, making the method well-suited for constrained clinical environments.

2 Related Work

Most research on GNNs for link prediction follows two main directions. The first focuses on message passing architectures and training methods that improve accuracy on topology-only benchmarks. Notable models include MPNNs, GCNs, GraphSAGE, and GAT, which are commonly evaluated on standardized topology-only benchmarks such as the Open Graph Benchmark (OGB), which provides unified splits and evaluation metrics [16, 20, 34, 35, 36, 37, 38].

The second focuses on incorporating GNNs into larger machine learning systems tailored for domain-specific applications, including biomedicine, chemistry, and recommender systems [39, 40, 41, 42]. These works often integrate graphs with other modalities or domain priors and are benchmarked on specialized datasets. Domain-specific pipelines commonly follow four main approaches.

1. Statistical and topology-only embeddings to initialize and train GNNs. A common recipe learns node and edge representations purely from structure, then trains a predictor [21, 43, 44]. Random-walk and factorization methods provide proximity and co-occurrence priors that are widely used for link prediction and as initializations for downstream GNNs [19, 45].

2. Frozen, domain-aware language model embeddings to train GNNs. Cascaded designs precompute text embeddings to inject domain semantics, keep them frozen, and then train only the graph model [23]. In biomedicine, BERT-derived features or similarity graphs from literature or clinical text are paired with Graph Convolution Networks or GNN back ends, improving accuracy without enlarging the graph-based models or co-training the text model [46, 47].

3. Joint training of language models or knowledge graphs with graph encoders. These approaches optimize auxiliary representations together with the graph encoder to inject knowledge end to end, which typically increases parameter count and training cost. Examples include KEPLER, CoLAKE, and systems that co-train language models with GNNs at scale [48, 49].

4. MLPs regularized by pretrained knowledge graphs. PLATO targets high-dimensional tabular settings ($d \gg n$) by regularizing a small MLP with an auxiliary knowledge graph [50]. It infers the first-layer weights from knowledge-graph node embeddings via a lightweight message passing module, so inference remains inexpensive and no GNN is used.

Across the first three approaches, the main goal has been maximizing predictive accuracy, irrespective of the encoder size. The fourth approach, PLATO, also targets compact inference. However, its pipeline pretrains knowledge graph embeddings and depends on a large heterogeneous knowledge graph, which adds a nontrivial training phase even if the predictor is small at test time. Constructing comparable knowledge graphs for new benchmarks often requires multi-source integration and extensive corpus curation that are domain and dataset-specific [50].

We build on these patterns but emphasize compact GNN encoders that leverage domain knowledge from language models, with training and inference designed for deployability in resource-constrained clinical settings. In contrast to knowledge-graph pretraining pipelines such as PLATO, we package domain knowledge as language-model-derived representations and refine them contrastively, making them portable and avoiding custom knowledge graph construction while preserving the inductive biases that make GNNs effective. For comparability and rigor, we follow OGB practices with standardized splits and evaluators.

3 Methods

3.1 Overview

We study link prediction with the premise that task-optimized domain-aware node initialization provides a strong prior for compact graph encoders. Our method comprises three main steps. First, we extract text-derived node features by passing DrugBank drug names [18] through a frozen SciBERT encoder. Next, these node features are refined by a two-layer MLP, trained with a contrastive objective. The resulting embeddings are then used to initialize a Graph Neural Network that both learns structural signals and is regularized to remain close to its semantic prior. This approach allows us to disentangle the effect of semantic priors from learned structural features.

We evaluate our work on the ogbl-ddi dataset, a network where nodes correspond to DrugBank compounds and edges represent drug pairs whose combined effect is different from independent action. The benchmark provides disjoint train, validation, and test edge splits $E_{\text{train}}, E_{\text{valid}}, E_{\text{test}}$ (positives), which are evaluated based on Hits@K. To avoid sampling held-out positives as negatives, we define the negative edge set with respect to the union of all positive edges across splits, $E_{\text{all}} = U(E_{\text{train}} \cup E_{\text{valid}} \cup E_{\text{test}})$. During training, positive labels are only provided from E_{train}, while E_{valid} and E_{test} are held out.

3.2 Text-Derived Node Initialization

For each drug node, we concatenate the associated DrugBank name and description to form an input sequence t_u, which is encoded by a frozen SciBERT model to produce a vector embedding $e_u \in \mathbb{R}^d$. All embeddings are then stacked into a matrix $E \in \mathbb{R}^{n \times d}$ for the n drugs. We do not fine-tune SciBERT during subsequent training. We train a two-layer MLP $g : \mathbb{R}^d \to \mathbb{R}^p$ using a contrastive

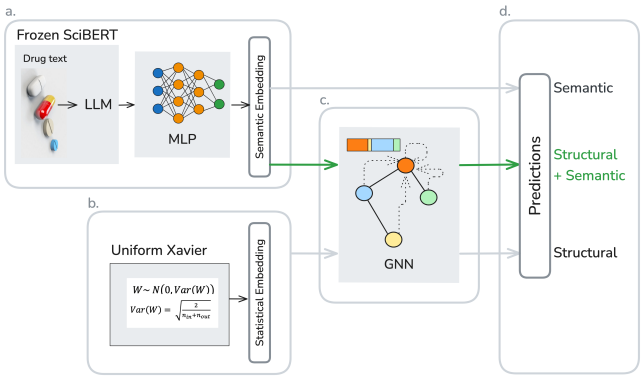

Figure 1: Task-optimized domain-aware link-prediction pipeline. (a) Text-derived semantic node embeddings from a frozen SciBERT refined by a contrastively trained two-layer MLP. (b) Statistical (Xavier) node embeddings. (c) Compact GNN encoder (GraphSAGE-mean with edge-aware messages, Jumping Knowledge (max), Proximal Regularization, Sparse Hard-negative edge sample). (d) Link prediction with ogbl-ddi evaluator using the chosen embeddings.

loss, where positive drug pairs are sampled from the known DDI network and negatives are drawn from the disjoint negative set defined over $\bar{E}_{\text{all}}$. For each anchor, k negatives are drawn from this disjoint set. Additional details on the negative sampling can be found in Appendix A.4.1. Our MLP outputs unit-normalized p-dimensional embeddings for each drug. These downstream features are stacked to form the matrix $X \in \mathbb{R}^{n\times 512}$, which is precomputed and saved. The initial embedding matrix $E_0 = X$ is used as the anchor for the proximal regularizer during GNN training. More details on the architecture and objective can be found in Appendices A.4.2 and A.4.3, respectively.

3.3 Graph Neural Network with Proximal-Anchored Fine-Tuning

We initialize each drug node with the matrix $X \in \mathbb{R}^{n\times 512}$ of unit-normalized embeddings obtained from our contrastive MLP. These serve as the initial node features for the graph neural network. Let $h^{(0)} = X$ represent the input features. Our graph neural network is composed of L layers. Each layer uses a GraphSAGE mean aggregator with edge-aware message passing. At every layer, node features are updated by aggregating information from neighboring nodes and their associated edges. Each layer applies layer normalization, ReLU activation, and dropout. Where necessary, residual connections are added before dropout layers to preserve signal propagation. Detailed equations for each layer can be found in Appendix A.5.1. After L layers, we obtain a sequence of representations for each node. To aggregate information across different layers, we apply max-pooling Jumping Knowledge aggregation to these layer outputs. This results in a final node summary $h_i^\star$ for each node i. For link prediction, we compute a feature vector for each candidate pair (u, v) by concatenating the elementwise product $h_u^\star \odot h_v^\star$, the absolute difference $|h_u^\star - h_v^\star|$, and the node summaries $h_u^\star$ and $h_v^\star$. This concatenated vector is passed through a shallow MLP, which produces a logit $\phi(u, v)$. The probability of an interaction is given by the sigmoid of this logit, $\sigma(\phi(u, v))$.

During training, node representations are computed once per batch over the edge set E_{train} and then used to score all pairs in the batch. For every observed positive edge (u_i, v_i^+), we draw R negative samples $(u_i, v_{i,r}^-)$ i.i.d. from $\bar{E}_{\text{all}}$. To better separate challenging cases, we employ hard-negative mixing. For each batch, the κ negative pairs with the largest predicted logits are selected and replicated to emphasize them in the loss calculation. Our training objective is a weighted sum of three terms: a pairwise Bayesian Personalized Ranking (BPR) loss, a pointwise binary cross-entropy loss, and a proximal penalty that encourages the learned node representations to remain close to their initial semantic values. The loss for a mini-batch is

$$\mathcal{L} \;=\; \mathcal{L}_{\text{BPR}} \;+\; 0.15\,\mathcal{L}_{\text{BCE}} \;+\; \lambda\,\|X_S - E_0(S)\|_2^2 \tag{1}$$

where S indexes the nodes present in the batch and E_0 is the matrix of initial semantic embeddings. This regularization ensures that structural learning in the GNN is anchored to the semantic prior established by text-derived features. Detailed formulations of the loss components and training procedures are provided in Appendix A.5.2.

3.4 Ablations Used

We perform two ablation experiments as follows.

1. **Xavier initialization:** As a control, we replace X in the GNN with a trainable table of Xavier node features. The proximal penalty term is omitted from the loss.
2. **MLP-only scorer:** Here, node embeddings are obtained by training the two-layer MLP g as described above. After training, the matrix $Z = g(E) \in \mathbb{R}^{n \times p}$ is fixed. Link prediction is performed by computing the cosine similarity $r(u, v) = Z_u^\top Z_v$ for each drug pair. No Graph Neural Network layers are used in this setting. Additional evaluation details are provided in the Appendix A.3.1.

4 Results

We compare the performance of our models against those on the ogbl-ddi public leaderboard [33]. All leaderboard models are structure-only and do not leverage domain-specific priors. Leaderboard ranks models by Test Hits@20 for link prediction. Our structure+semantic model, as shown in Table 1, uses 0.52% of the 976 million parameters used by the top-ranked model shown in Appendix Table 2, while achieving Hits@20 within 2.19% of that model. Compared to the second-ranked model, our model reaches within 0.24% of its performance with 51.7% fewer parameters. Additionally, our model outperforms the third-ranked baseline by 2.1% despite using fewer parameters. Our results show that adding semantic priors from pretrained scientific language models with task-optimized refinement allows our model to achieve strong link prediction accuracy with much lower model complexity.

Table 1: Comparing structure-only, semantic-only, and structure+semantic models. Details on how the parameter sizes were chosen for size-controlled comparisons are provided in Appendix A.2. Across all our parameter-matched tests, the structure+semantic models outperformed the structure-only models at Test Hits@20.

Ablation	Model Components	Test Hits@20	Parameters
Structure-only	Xavier embeddings + GNN	0.9292	5,079,041
Semantic-only	SciBERT + contrastive MLP	0.9411	525,057
Structure+Semantic	SciBERT + MLP embeddings + GNN	**0.9753**	**5,079,041**

We conducted ablations that isolate the contributions of the semantic and structural components. The semantic-only model encodes DrugBank text with SciBERT, learns a contrastive objective with a two-layer MLP, and predicts links without any graph neural network layers. The structure-only model, on the other hand, consists of a Graph Neural Network with Xavier-initialized node embeddings, trained with Bayesian Personalized Ranking loss, hard negative sampling, edge-aware message passing, and Jumping Knowledge aggregation layers. Our semantic-only model achieves a 1.2% higher Test Hits@20 performance than our structure-only baseline while using 90% fewer parameters. Despite the simpler architecture, the higher performance of the compact semantic-only model, indicates that language representations are able to efficiently learn features that are directly relevant for predicting drug-drug interactions, even in the absence of explicit structural learning.

Our semantic+structure model shares the GNN backbone, and training setup with the structure-only model, but initializes nodes with contrastively refined SciBERT embeddings and applies a proximal penalty that encourages node representations to remain close to their initial values. The structure+semantic model achieves 3.6% higher Hits@20 performance than the semantic-only model, indicating that interaction topology provides a complementary signal beyond text-derived semantics. Compared to the structure-only model, the structure+semantic model improves Hits@20 by 4.9% for the same parameter count. This can be attributed to the fact that statistical initialization forces the GNN to infer semantics solely from network topology. In contrast, combining contrastively refined, domain-aware embeddings with a proximal anchoring penalty establishes a near-fixed semantic prior, allowing message passing to learn structural residuals relative to this prior more efficiently. This approach results in more accurate DDI predictions than using semantic or structural learning alone.

We assess the trade-off between model accuracy and parameter size by conducting a Pareto analysis across all published results on the ogbl-ddi leaderboard, as shown in Figure 2. Both our semantic-only and structure+semantic models lie on the Pareto frontier of performance versus size. Our

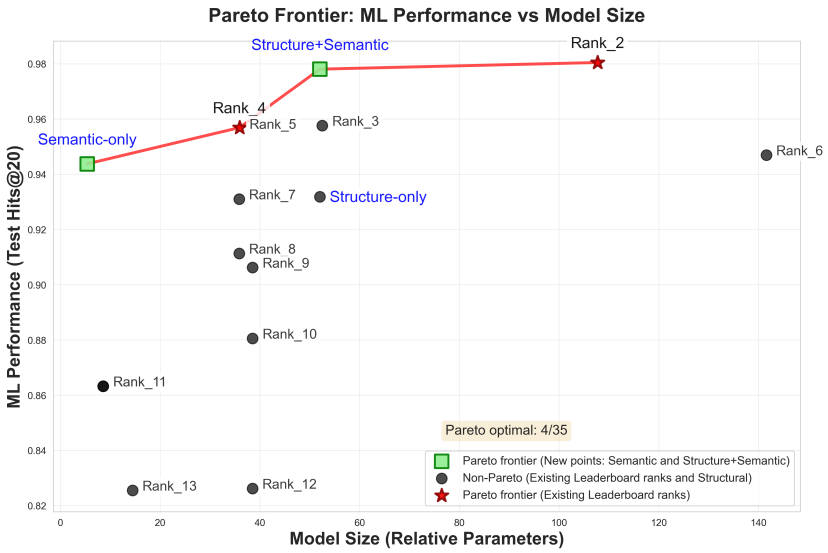

Figure 2: Pareto analysis of model performance versus model size for all published structure-only models on the ogbl-ddi benchmark. For clarity, the outlier corresponding to the top-ranked model (976M parameters, $10,000$ on the x-axis) is omitted. Only non-Pareto points with a performance above 0.8 are included. More details can be found in the Appendix A.1.

semantic-only model establishes a new efficiency benchmark outperforming 81% of the leaderboard entries while using between 63.1% to 99.97% fewer parameters than any other existing model. The structure+semantic model extends the Pareto frontier at higher accuracy, outperforming 94% of the published models. These results suggest that semantic priors, combined with graph structure, can enable more efficient learning than purely structure-based models across model sizes in the performance-efficiency trade-off space.

5 Discussion

We show that compact GNNs, when initialized with task-optimized, domain-aware biomedical text embeddings, attain competitive accuracy on DDI link prediction compared to much larger structure-only GNNs. On the ogbl-ddi benchmark, our structure+semantic model achieves almost comparable performance, while using only 0.52% of the 976 million parameters of the highest performing model. We attribute this efficiency to encoding pharmacological semantics directly into node representations instead of inferring them solely from graph topology. In addition, our semantic-only model, outperforms our structure-only baseline. This result shows that biomedical text provides meaningful information for predicting drug interactions, even when graph-topology is not used. Both the semantic-only and structure+semantic models lie on the Pareto frontier of accuracy versus size when compared to prior work. These results suggest that semantic priors makes GNN-based DDI link prediction feasible in resource-constrained settings without sacrificing accuracy.

Our current approach relies on DrugBank descriptions and a SciBERT language model. These sources can miss important pharmacological properties and may carry biases from the original corpus. As biomedical and scientific language models and curated corpora continue to advance, text-derived priors are likely to become increasingly informative and further improve model performance.

Future directions of our work include the following. First, we aim to extend language-based encoding by incorporating descriptions of DDIs from more sources, such as clinical monographs and other scientific language models. Using this information, we will construct edge embeddings that capture pair-specific details and evaluate whether including these additional edge-level embeddings improves the link prediction performance of compact GNNs. Second, we aim to assess generalization and transfer by applying our semantic initialization approach to other related tasks, including drug-target interaction, adverse event prediction, and drug repurposing. Insights from these studies could further improve the design of compact models for resource-constrained biomedical environments.

6 Acknowledgements

The author would like to thank Professor Adam Wierman for his invaluable feedback on this work.

References

[1] Eric H Young, Samantha Pan, Alex G Yap, Kelly R Reveles, and Kajal Bhakta. Polypharmacy prevalence in older adults seen in united states physician offices from 2009 to 2016. *PLOS ONE*, 16(8):e0255642, 2021.

[2] Xiaowen Wang, Keyang Liu, Kokoro Shirai, Chengyao Tang, Yonghua Hu, Ying Wang, Yuantao Hao, and Jia-Yi Dong. Prevalence and trends of polypharmacy in u.s. adults, 1999–2018. *Global Health Research and Policy*, 8(1):25, 2023.

[3] Robert L. Jr Maher, Joseph T. Hanlon, and Emily R. Hajjar. Clinical consequences of polypharmacy in elderly. *Expert Opinion on Drug Safety*, 13(1):57–65, 2014.

[4] Melanie J. Koren, Neil A. Kelly, Jennifer D. Lau, Chanel K. Jonas, Laura C. Pinheiro, Samprit Banerjee, Monika M. Safford, and Parag Goyal. Association of healthy lifestyle and incident polypharmacy. *American Journal of Medicine*, 2023. In Press.

[5] Huaqiao Jiang, Yanhua Lin, Weifang Ren, Zhonghong Fang, Yujuan Liu, Xiaofang Tan, Xiaoqun Lv, and Ning Zhang. Adverse drug reactions and correlations with drug–drug interactions: A retrospective study of reports from 2011 to 2020. *Frontiers in Pharmacology*, 13:923939, 2022.

[6] Nicholas P Tatonetti, Patrick P Ye, Roxana Daneshjou, and Russ B Altman. Data-driven prediction of drug effects and interactions. *Science Translational Medicine*, 4(125):125ra31, 2012.

[7] Janet Sultana, Paola Cutroneo, and Gianluca Trifirò. Clinical and economic burden of adverse drug reactions. *Journal of Pharmacology & Pharmacotherapeutics*, 4(Suppl 1):S73–S77, 2013.

[8] Yan Zhao, Jun Yin, Li Zhang, Yong Zhang, and Xing Chen. Drug–drug interaction prediction: databases, web servers and computational models. *Briefings in Bioinformatics*, 25(1):bbad445, 2024.

[9] Huimin Luo, Weijie Yin, Jianlin Wang, Ge Zhang, Wenjuan Liang, Junwei Luo, and Chaokun Yan. Drug–drug interactions prediction based on deep learning and knowledge graph: A review. *iScience*, 27(3):109148, 2024.

[10] Ke Han, Peigang Cao, Yu Wang, Fang Xie, Jiaqi Ma, Mengyao Yu, Jianchun Wang, Yaoqun Xu, Yu Zhang, and Jie Wan. A review of approaches for predicting drug–drug interactions based on machine learning. *Frontiers in Pharmacology*, 12:814858, 2022.

[11] Kevin B. Wood. Pairwise interactions and the battle against combinatorics in multidrug therapies. *Proceedings of the National Academy of Sciences*, 113(37):10231–10233, 2016.

[12] Risha I Patel and Robert D Beckett. Evaluation of resources for analyzing drug interactions. *Journal of the Medical Library Association: JMLA*, 104(4):290–295, 2016.

[13] Sebastien D Kiogou and Terrence J Adam. Clinical data repositories: Primary sources of drug-to-drug interaction detection and risk assessment. *AMIA Annual Symposium Proceedings*, 2022:625–633, 2023.

[14] Serkan Ayvaz, John Horn, Oktie Hassanzadeh, Qian Zhu, Johann Stan, Nicholas P Tatonetti, Santiago Vilar, Mathias Brochhausen, Matthias Samwald, Majid Rastegar-Mojarad, Michel Dumontier, and Richard D Boyce. Toward a complete dataset of drug-drug interaction information from publicly available sources. *Journal of Biomedical Informatics*, 55:206–217, 2015.

[15] Kin Wah Fung, Joan Kapusnik-Uner, Jean Cunningham, Stefanie Higby-Baker, and Olivier Bodenreider. Comparison of three commercial knowledge bases for detection of drug–drug interactions in clinical decision support. *Journal of the American Medical Informatics Association*, 24(4):806–812, 2017.

[16] Wei Hu, Matthias Fey, Marinka Zitnik, Yu Dong, Hang Ren, Bowei Liu, Mauro Catasta, and Jure Leskovec. Open Graph Benchmark: Datasets for Machine Learning on Graphs. *NeurIPS Datasets and Benchmarks*, 2020.

[17] Yang Qiu, Yang Zhang, Yifan Deng, Shichao Liu, and Wen Zhang. A comprehensive review of computational methods for drug–drug interaction detection. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 19(4):1968–1985, 2022.

[18] David S Wishart, Yannick D Feunang, An C Guo, and et al. Drugbank 5.0: a major update to the drugbank database for 2018. *Nucleic Acids Research*, 46(D1):D1074–D1082, 2018.

[19] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 701–710, 2014.

[20] Vijay Prakash Dwivedi, Anh Tuan Luu, Thomas Laurent, Yoshua Bengio, and Xavier Bresson. Graph neural networks with learnable structural and positional representations. In *International Conference on Learning Representations*, 2022.

[21] Aditya Grover and Jure Leskovec. node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 855–864. ACM, 2016.

[22] Muhan Zhang and Yixin Chen. Link prediction based on graph neural networks. In *Advances in Neural Information Processing Systems*, volume 31, 2018.

[23] Aaron Zolnai-Lucas, Jack Boylan, Chris Hokamp, and Parsa Ghaffari. Stage: Simplified text-attributed graph embeddings using pre-trained llms. *arXiv preprint arXiv:2407.12860*, 2024.

[24] Marinka Zitnik, Monica Agrawal, and Jure Leskovec. Modeling polypharmacy side effects with graph convolutional networks. *Bioinformatics*, 34(13):i457–i466, 2018.

[25] Guy Shtar, Asnat Greenstein-Messica, Eyal Mazuz, Lior Rokach, and Bracha Shapira. Predicting drug characteristics using biomedical text embedding. *BMC Bioinformatics*, 23(1):527, 2022.

[26] Wei-Lin Chiang, Xuanqing Liu, Si Si, Yang Li, Samy Bengio, and Cho-Jui Hsieh. Cluster-gcn: An efficient algorithm for training deep and large graph convolutional networks. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 257–266, 2019.

[27] Hanqing Zeng, Hongkuan Zhou, Ajitesh Srivastava, Rajgopal Kannan, and Viktor Prasanna. Graphsaint: Graph sampling based inductive learning method. In *International Conference on Learning Representations*, 2020.

[28] Muskan Garg, Shaina Raza, Shebuti Rayana, Xingyi Liu, and Sunghwan Sohn. The rise of small language models in healthcare: A comprehensive survey. *arXiv preprint arXiv:2504.17119*, 2025.

[29] Amol A Verma, Joshua Murray, Russell Greiner, Joseph Paul Cohen, Kaveh G Shojania, Marzyeh Ghassemi, Sharon E Straus, Chloe Pou-Prom, and Muhammad Mamdani. Implementing machine learning in medicine. *CMAJ*, 193(34):E1351–E1357, 2021.

[30] Terrence C Lee, Neil U Shah, Alyssa Haack, and Sally L Baxter. Clinical implementation of predictive models embedded within electronic health record systems: A systematic review. *Informatics*, 7(3):25, 2020.

[31] Emma Strubell, Ananya Ganesh, and Andrew McCallum. Energy and policy considerations for deep learning in NLP. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3645–3650, Florence, Italy, 2019. Association for Computational Linguistics.

[32] Iz Beltagy, Kyle Lo, and Arman Cohan. Scibert: A pretrained language model for scientific text. *arXiv preprint arXiv:1903.10676*, 2019.

[33] Ogb link property prediction leaderboard (ogbl-ddi). `https://ogb.stanford.edu/docs/leader_linkprop/#ogbl-ddi`. Accessed Apr. 18, 2025.

[34] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How powerful are graph neural networks? In *International Conference on Learning Representations*, 2019.

[35] William L Hamilton, Rex Ying, and Jure Leskovec. Inductive representation learning on large graphs. In *NeurIPS*, 2017.

[36] Justin Gilmer, Samuel S Schoenholz, Patrick F Riley, Oriol Vinyals, and George E Dahl. Neural message passing for quantum chemistry. In *Proceedings of the 34th International Conference on Machine Learning*, pages 1263–1272, 2017.

[37] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. Graph attention networks. In *International Conference on Learning Representations*, 2018.

[38] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations*, 2017.

[39] Yu Rong, Yatao Bian, Tingyang Xu, Weiyang Xie, Ying Wei, Wenbing Huang, and Junzhou Huang. Self-supervised graph transformer on large-scale molecular data. In *Advances in Neural Information Processing Systems*, volume 33, 2020.

[40] Seyone Chithrananda, Gabriel Grand, and Bharath Ramsundar. Chemberta: Large-scale self-supervised pretraining for molecular property prediction. *arXiv preprint arXiv:2010.09885*, 2020.

[41] Michelle M. Li, Kexin Huang, and Marinka Zitnik. Representation learning for networks in biology and medicine: Advancements, challenges, and opportunities. *arXiv preprint arXiv:2104.04883*, 2021.

[42] Chen Gao, Yu Zheng, Nian Li, Yinfeng Li, Yingrong Qin, Jinghua Piao, Yuhan Quan, Jianxin Chang, Depeng Jin, Xiangnan He, and Yong Li. A survey of graph neural networks for recommender systems: Challenges, methods, and directions. *arXiv preprint arXiv:2109.12843*, 2021.

[43] Tanish Jain. Online link prediction with graph neural networks. https://medium.com/stanford-cs224w/online-link-prediction-with-graph-neural-networks-46c1054f2aa4, 2022.

[44] Ananth Subramaniam. Understanding gnns for link prediction: From graphsage to seal. https://medium.com/@ananth.subramaniam/gnns-for-link-prediction-3a0fbeeaaed3, 2022.

[45] Sami Abu-El-Haija, Bryan Perozzi, Rami Al-Rfou, and Alexander A. Alemi. Watch your step: Learning node embeddings via graph attention. In *Advances in Neural Information Processing Systems*, volume 31, 2018.

[46] Mark Lennox, Neil Robertson, and Barry Devereux. Modelling drug–target binding affinity using a BERT based graph neural network. In *2021 43rd Annual International Conference of the IEEE Engineering in Medicine & Biology Society (EMBC)*, pages 4348–4353. IEEE, 2021.

[47] Hongyu Kang, Li Hou, Yaowen Gu, Xiao Lu, Jiao Li, and Qin Li. Drug–disease association prediction with literature based multi-feature fusion. *Frontiers in Pharmacology*, 14:1205144, 2023.

[48] Xiaozhi Wang, Tianyu Gao, Zhaocheng Zhu, Zhengyan Zhang, Zhiyuan Liu, Juanzi Li, and Jian Tang. Kepler: A unified model for knowledge embedding and pre-trained language representation. *arXiv preprint arXiv:1911.06136*, 2019.

[49] Tianxiang Sun, Yunfan Shao, Xipeng Qiu, Qipeng Guo, Yaru Hu, Xuanjing Huang, and Zheng Zhang. Colake: Contextualized language and knowledge embedding. In *Proceedings of the 28th International Conference on Computational Linguistics*, 2020.

[50] Camilo Ruiz, Hongyu Ren, Kexin Huang, and Jure Leskovec. Enabling tabular deep learning when $d \gg n$ with an auxiliary knowledge graph. *arXiv preprint arXiv:2306.04766*, 2023.

A Technical Appendices and Supplementary Material

A.1 Pareto Analysis

Table 2 shows the inputs for the Pareto analysis of model performance versus size.

Table 2: Pareto analysis of ogbl-ddi models from the leaderboard and our ablations. The first three rows correspond to our ablation variants. The last three columns, namely Model Size, ML performance (ML Perf), and hardware capacity (HW scale), serve as inputs to the Pareto analysis and are normalized to the maximum Test Hits (0.9972), Parameters (976 M), and TFLOPs (82.6) values shown below. Out of the 32 current entries in the leaderboard only models with a Test Hits@20 score above 0.5 are included in the table.

Rank	Method	Test Hits	Contact	Params	Hardware	TFLOPs	Model Size	ML Perf.	HW Scale
–	Xavier + GNN	0.9292	A.Viswesh	5.08M	A100 40G	19.5	52.04	0.932	0.236
–	Scibert	0.9411	A.Viswesh	0.53M	A100 40G	19.5	5.38	0.944	0.236
–	Scibert+GNN	0.9753	A.Viswesh	5.08M	A100 40G	19.5	52.04	0.978	0.236
1	HyperFusion	**0.9972**	X. Zhang	976M	RTX 3080	29.8	10000	**1.000**	0.360
2	ELGNN	0.9777	Z.H. Wong	10.5M	A100 80G	19.5	107.7	0.980	0.236
3	GCN	0.9549	S. Liang	5.13M	P6000	12.6	52.51	0.958	0.153
4	GIDN	0.9549	Z. Wang	3.51M	DepGraph	n/a	35.93	0.958	0.153
5	AGDN	0.9538	C. Sun	3.51M	V100 16G	16.4	35.93	0.956	0.199
6	Refined-GAE	0.9443	W. Ma	13.8M	RTX 4090	82.6	141.6	0.947	1.000
7	PSG	0.9284	J. Lv	3.50M	V100 32G	16.4	35.85	0.931	0.199
8	PLNLP	0.9088	Z. Wang	3.50M	P40	12.0	35.83	0.911	0.145
9	GDNN	0.9037	H. Zhou	3.76M	A40	37.4	38.54	0.906	0.453
10	GraphSAGE+E	0.8781	J. Yang	3.76M	V100 32G	16.4	38.54	0.881	0.199
11	CFLP	0.8608	T. Zhao	0.84M	2080 Ti	13.4	8.58	0.863	0.162
12	GraphSAGE+A	0.8239	B. Li	3.76M	V100 16G	16.4	38.53	0.826	0.199
13	NCN	0.8232	X. Wang	1.41M	RTX 4090	82.6	14.47	0.826	1.000
14	ELPH	0.7704	B. Chamberlain	2.91M	K80	8.7	29.82	0.773	0.105
15	DEA+JKNet	0.7672	Y. Yang	1.76M	T4	8.1	18.07	0.769	0.098
16	BUDDY	0.7654	B. Chamberlain	2.71M	K80	8.7	27.80	0.768	0.105
17	GraphSAGE+EPS	0.7495	Q. Huang	1.42M	2080 Ti	13.4	14.56	0.752	0.162
18	LRGA+GCN	0.7385	C. Chen	10.2M	V100 32G	16.4	104.9	0.741	0.199
19	MAD Learning	0.6781	Y. Luo	1.23M	1080 Ti	11.3	12.59	0.680	0.137
20	LRGA+GCN	0.6230	O. Puny	1.58M	P100	9.3	16.15	0.625	0.113
21	GCN+JKNet	0.6056	H. He	1.42M	1080 Ti	11.3	14.56	0.607	0.137
22	NGNN+SAGE	0.5770	Y. Song	1.62M	V100 16G	15.7	16.58	0.579	0.190
23	NGNN+GCN	0.5483	Y. Song	1.49M	V100 16G	15.7	15.24	0.550	0.190
24	GraphSAGE	0.5390	M. Fey	1.42M	RTX 2080	10.1	14.56	0.541	0.122

A.2 Ablations for Parameter Size Selection

Table 3: Ablations performed to select parameter sizes across model families for size-controlled comparisons. Best configurations are bolded. Additional ablations were evaluated but did not outperform those shown and are not included.

Ablation	Model Components	Test Hits@20	Parameters
Semantic-only	**SciBERT + contrastive MLP**	**0.9411**	**525,057**
Semantic-only	SciBERT + contrastive MLP	0.9332	787,713
Structure-only	Xavier embeddings + GNN	0.9203	4,816,685
Structure-only	Xavier embeddings + GNN	0.9292	5,079,041
Structure-only	Xavier embeddings + GNN	0.9300	5,344,769
Structure+Semantic	SciBERT + MLP embeddings + GNN	0.9671	4,816,685
Structure+Semantic	**SciBERT + MLP embeddings + GNN**	**0.9753**	**5,079,041**
Structure+Semantic	SciBERT + MLP embeddings + GNN	0.9709	5,344,769

Table 3 presents the ablations we performed to select parameter sizes for size-controlled comparisons. Performance was measured by Test Hits@20. For the semantic-only model, best performance was obtained at our smallest feasible configuration. Performance decreased with additional capacity. For the structure+semantic model, increasing capacity beyond 5.079M did not yield further gains in performance. The structure-only model had only a slight increase of 0.08% in performance for a 5.32% greater parameter size (5.079M versus 5.344M). Therefore, for size-controlled comparisons, we fixed the parameter count at 5.079M, for both the structure-only and structure+semantic models.

Across all parameter-matched settings, the structure+semantic model consistently outperformed the structure-only baseline on Test Hits@20.

A.3 Common Setup

A.3.1 Evaluation

Hits@K is reported using the official ogbl-ddi evaluators. It supports two negative formats: *flat* (a single list of negatives) and *grouped* (each positive paired with R negatives). Our routine detects both and computes probabilities accordingly. For G validation (or test) positives with associated negatives,

$$\text{Hits@}K \;=\; \frac{1}{G}\sum_{i=1}^{G} \mathbf{1}\big[\,\text{rank}\big(y_i^+\big) \le K\,\big]\,,$$

where y_i^+ denotes the positive score and $\text{rank}(\cdot)$ is computed among the corresponding negative set.

A.3.2 Environment

Our experiments use PyTorch 2.6. Torch Dynamo/compile/JIT paths are disabled to avoid incidental non-determinism, seeds are fixed for Python/NumPy/Torch (and CUDA where applicable), cuDNN benchmarking is turned off, and deterministic algorithms are enabled where available. Deserialization uses `torch.load` with `weights_only=False`. Training was performed on a single NVIDIA A100 (40 GB) GPU.

A.4 Additional Details for Frozen SciBERT + Contrastive MLP

A.4.1 Negative Sampling

Let $\mathcal{N}(u)$ be the undirected neighbors of u in the union edge set (train∪valid∪test). For each training anchor u, we precompute a reservoir $\mathcal{N}^-(u) \subseteq V \setminus \big(\{u\} \cup \mathcal{N}(u)\big)$ by uniform sampling without replacement up to budget S. During training, a round-robin picker returns exactly k negatives per anchor; when a reservoir is exhausted it shuffles and wraps. Anchors with empty reservoirs are skipped in that mini-batch.

A.4.2 MLP Architecture

Given a frozen SciBERT embedding e_u, the two-layer MLP g computes

$$h = \text{ReLU}(W_1 e_u + b_1), \qquad z = W_2\,\text{Dropout}(h) + b_2, \qquad \hat{z} = \frac{z}{\|z\|_2} \in \mathbb{R}^p.$$

A.4.3 Contrastive Objective

Given positives $\{(u_i, v_i^+)\}_{i=1}^B$ and per-anchor negatives $\{v_{i,r}^-\}_{r=1}^k$, we compute unit-normalized projections $\hat{z} = g(e)$ and cosine scores by inner product. For each anchor u_i we form logits and optimize cross-entropy with the positive at index 0. The scale s is *learned*. We parameterize $s = \exp(\gamma)$ where γ initializes to $\log(1/\tau)$ with $\tau = 0.07$. It is clamped before exponentiation for stability. An optional symmetric term swaps anchors/targets; the loss is the mean of the two directions. Full expressions are shown below.

Per anchor u_i logits and scale are defined as

$$\ell_i = s\,[\,\hat{z}_{u_i}^\top \hat{z}_{v_i^+};\; \hat{z}_{u_i}^\top \hat{z}_{v_{i,1}^-}; \ldots; \hat{z}_{u_i}^\top \hat{z}_{v_{i,k}^-}\,] \in \mathbb{R}^{1+k}, \quad s = \exp(\gamma),\ \gamma \in [\log(1/100), \log(100)].$$

The one-sided cross-entropy is

$$\mathcal{L}_{\text{NCE}}^{\rightarrow} = \frac{1}{B}\sum_{i=1}^{B}\Big(-\log\frac{\exp(\ell_{i,0})}{\sum_{j=0}^{k}\exp(\ell_{i,j})}\Big).$$

The symmetric loss is

$$\mathcal{L}_{\text{NCE}} = \tfrac{1}{2}\big(\mathcal{L}_{\text{NCE}}^{\rightarrow} + \mathcal{L}_{\text{NCE}}^{\leftarrow}\big).$$

Due to resource constraints, the projector outputs a 256-dimensional, ℓ_2-normalized vector z_u. After training, a lightweight decoder $D : \mathbb{R}^{256} \to \mathbb{R}^{512}$ expands z_u to $x_u = D(z_u)$ with a cosine-similarity target of 0.995. Stacking x_u yields $X \in \mathbb{R}^{n \times 512}$; X is saved, and an epoch-0 snapshot $E_0 = X$ is retained for proximal anchoring.

A.4.4 Optimization and Scheduling

We use AdamW with two parameter groups: (i) the two-layer MLP projector g with weight decay and (ii) a learnable temperature τ with no weight decay. Learning rates warm up linearly for $T_w = 10$ epochs and then follow cosine decay with $\eta_{\min} = 0.05\,\eta_0$ (no restarts).

A.4.5 Hyperparameters

Unless stated otherwise hyperparameters used for MLP are as follows. We use a hidden dimension of 512, projection dimension of 256, two layers, dropout probability of 0.05, 500 epochs, and a batch size of 4096. We also use 400 negatives per anchor, a reservoir size of $S = 4500$, a base learning rate of $\eta_0 = 10^{-4}$ warmup $T_w = 10$ then cosine annealing with $\eta_{\min} = 0.05\,\eta_0$, weight decay 5×10^{-4}; temperature initialization $\tau_{\text{init}} = 0.07$, and an early-stopping patience of 40. Symmetric loss is enabled.

A.5 Additional Training Details - GNN

A.5.1 Encoder Equations and Regularization

Let $h^{(0)} = X$. Each layer forms edge-aware messages

$$m_{j\to i}^{(\ell)} = \mathrm{ReLU}\big(h_j^{(\ell-1)} + W_e^{(\ell)} a_{ij}\big),$$

aggregates by mean and applies a root update

$$\tilde{h}_i^{(\ell)} = \frac{1}{|\mathcal{N}(i)|} \sum_{j \in \mathcal{N}(i)} m_{j\to i}^{(\ell)}, \qquad z_i^{(\ell)} = W_\ell^{(\ell)}\, \tilde{h}_i^{(\ell)} + \mathbf{1}_{\text{root}}\, W_r^{(\ell)}\, h_i^{(\ell-1)}.$$

Per layer, we apply LayerNorm→ReLU, add a residual if dimensions match, then dropout. During training, DropEdge randomly masks each edge with probability $p_{\text{edge}} = 0.05$. After L layers, we apply JK-max:

$$h_i^\star = \mathrm{JK}_{\max}\big(h_i^{(1)}, \ldots, h_i^{(L)}\big),$$

optionally followed by a linear projection to the final hidden dimension.

A.5.2 Losses(full expressions)

Let batch size be B, negatives per anchor R, and S the set of node indices touched by positives and negatives in the batch.

Pairwise ranking (BPR)

$$\mathcal{L}_{\mathrm{BPR}} = \frac{1}{BR} \sum_{i=1}^{B} \sum_{r=1}^{R} \mathrm{softplus}\Big(- \big(\phi(u_i, v_i^+) - \phi(u_i, v_{i,r}^-)\big)\Big).$$

Auxiliary pointwise BCE

$$\mathcal{L}_{\mathrm{BCE}} = -\frac{1}{B + BR} \sum_{i=1}^{B} \Big[\log \sigma\big(\phi(u_i, v_i^+)\big) + \sum_{r=1}^{R} \log\big(1 - \sigma(\phi(u_i, v_{i,r}^-))\big)\Big].$$

Proximal penalty (SciBERT+Contrastive MLP Node Embeddings) With epoch-0 snapshot E_0,

$$\mathcal{R}_{\text{prox}} = \lambda\, \|X_S - E_0(S)\|_2^2.$$

Total loss

$\mathcal{L} = \mathcal{L}_{\mathrm{BPR}} + 0.15\,\mathcal{L}_{\mathrm{BCE}} + \mathcal{R}_{\text{prox}}$ (proximal penalty only for SciBERT+Contrastive MLP Embeddings).

A.5.3 Optimization and Scheduling

We use AdamW with a lower learning rate for the node-embedding table than for the encoder and predictor. Learning rates warm up linearly for $T_w = 10$ epochs and then follow cosine decay (no restarts). Training uses mixed precision (AMP) and gradient-norm clipping at 1.0. To stabilize propagation, node embeddings are frozen during warmup and unfrozen afterward.

A.5.4 Hyperparameters

Unless stated otherwise hyperparameters we use are as follows. We use a hidden dimension $d = 512$, depth $L = 2$, dropout probability 0.2, epochs $= 500$, DropEdge probability $p_{\text{edge}} = 0.05$, $R = 2$ negatives per positive, AdamW learning rates of 10^{-3} (encoder/predictor) and 5×10^{-4} (embeddings), weight decay as 0.01, batch size as $65{,}536$, warmup $T_w = 10$, early-stopping patience as 40, mixed precision (AMP), and gradient-norm clipping as 1.0.