IGB: Addressing The Gaps In Labeling, Features, Heterogeneity, and Size of Public Graph Datasets for Deep Learning Research

Arpandeep Khatua
UIUC
USA

Vikram Sharma Mailthody
NVIDIA/UIUC
USA

Bhagyashree Taleka
University of Southern California
USA

Tengfei Ma
IBM Research
USA

Xiang Song
Amazon AWS
USA

Wen-mei Hwu
NVIDIA/UIUC
USA

ABSTRACT

Graph neural networks (GNNs) have shown high potential for a variety of real-world, challenging applications, but one of the major obstacles in GNN research is the lack of large-scale flexible datasets. Most existing public datasets for GNNs are relatively small, which limits the ability of GNNs to generalize to unseen data. The few existing large-scale graph datasets provide very limited labeled data. This makes it difficult to determine if the GNN model's low accuracy for unseen data is inherently due to insufficient training data or if the model failed to generalize. Additionally, datasets used to train GNNs need to offer flexibility to enable a thorough study of the impact of various factors while training GNN models.

In this work, we introduce the **Illinois Graph Benchmark (IGB)**, a research dataset tool that the developers can use to train, scrutinize and systematically evaluate GNN models with high fidelity. IGB includes both homogeneous and heterogeneous graphs of enormous sizes, with more than 40% of their nodes labeled. Compared to the largest graph datasets publicly available, the IGB provides over 162× more labeled data for deep learning practitioners and developers to create and evaluate models with higher accuracy. The IGB dataset is designed to be flexible, enabling the study of various GNN architectures, embedding generation techniques, and analyzing system performance issues. IGB is open-sourced, supports DGL and PyG frameworks, and comes with releases of the raw text that we believe foster emerging language models and GNN research projects. An early public version of IGB is available at https://github.com/IllinoisGraphBenchmark/IGB-Datasets.

CCS CONCEPTS

• **Computing methodologies** → **Neural networks**; Knowledge representation and reasoning; **Natural language processing**.

KEYWORDS

Datasets, Graph neural networks (GNNs), Graphs, Deep learning

KDD, August 06–10, 2023, Long Beach, CA

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/XXXXXXX.XXXXXXX

ACM Reference Format:
Arpandeep Khatua, Vikram Sharma Mailthody, Bhagyashree Taleka, Tengfei Ma, Xiang Song, and Wen-mei Hwu. 2023. IGB: Addressing The Gaps In Labeling, Features, Heterogeneity, and Size of Public Graph Datasets for Deep Learning Research. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD).* ACM, New York, NY, USA, 11 pages. https://doi.org/XXXXXXX.XXXXXXX

1 INTRODUCTION

Graph neural networks (GNNs) are a class of neural networks that operate on graph-structured data. GNNs have shown to be effective in addressing a variety of real-world applications such as fraud detection [18, 47, 73], recommendation systems [61, 70, 71], predicting molecular and protein structure [29, 57], knowledge representation [15], and more recently helping in fine-tuning large language models [72]. Their popularity has led to the development of various optimized frameworks and libraries [19, 63, 74] that enable the fast application of GNNs on new domains, making it easier for researchers and practitioners to leverage the power of GNNs in their work.

However, high-quality frameworks and libraries are necessary but not sufficient for enabling fast research progress in GNN. One of the major challenges in GNN research is the lack of large-scale datasets. This is because large graph datasets are typically propriety and most publicly available ones are rather small. The small size of these datasets makes it difficult to train GNNs that can handle large-graph structures and prevents the use of powerful pre-training techniques [16, 32, 59, 65]. These challenges make it difficult to fully leverage GNN potential and its applications.

To address these challenges, recent work such as OGBN and MAG [26, 27] have proposed open large graph benchmark suites providing up to 244 million nodes and 1.7 billion edge graphs. These datasets contain a diverse set of graphs and have been widely used in the research community to benchmark GNNs' performance. However, most existing datasets, including OGBN, provide very limited labeled data. As GNN downstream tasks are often trained as supervised learning tasks, having large labeled data matters, especially for multi-label classification problems. However, both OGBN and MAG use Arxiv [26, 27] class labels which provide 1.4 million labeled nodes, meaning only about 1% of the overall dataset is labeled! With such small labeled data usage during training, it becomes challenging to determine if the GNN model's low accuracy for unseen data is inherently due to insufficient training data or if the model itself fails to generalize [28, 37, 45, 62, 76].

Furthermore, the lack of flexible datasets hinders the researcher's ability to scrutinize and systematically evaluate the scalability of the GNN models, frameworks, and systems. Ideally, a dataset should provide (a) capability to study the impact of embedding generation techniques and their properties on the GNN model's accuracy, (b) provide sub-datasets of varying graph sizes and embeddings maintaining consistent homophily, and (c) provide a range of multi-class classification tasks with varying degrees of complexity. Without such flexible datasets, it is difficult to train models on small graph datasets and then evaluate their accuracy and execution efficiency on larger data corpus, a common scenario in industrial settings. Moreover, the framework and system scalability problems encountered with small datasets are different from those with large datasets, making it challenging to study the system requirements of GNNs.

To this end, this work proposes **Illinois Graph Benchmark (IGB)**, a research dataset tool that the researchers can use to scrutinize and systematically evaluate the GNN models and their execution efficiency. IGB provides access to enormous graph datasets for deep learning research consisting of both homogeneous and heterogeneous graphs, providing a diverse range of graph structures for training and evaluating GNN models. IGB homogeneous graph dataset (`IGB`) has up to 269 million nodes and about four billion edges. IGB heterogeneous (`IGBH`) graph dataset has up to 547 million nodes and about six billion edges. Both of these datasets come with more than 220 million labeled nodes created with a novel approach and with two complex node classification tasks (19 and 2983 unique classes). Compared to the largest graph dataset publicly available [26, 27], IGB provides over 162× more labeled data, providing ample opportunities for GNN and deep learning practitioners to generalize the models to unseen data.

Furthermore, as a research tool, IGB offers flexible design choices for GNN model developer to investigate and systematically assess their models' performance and execution efficiency. IGB provides variable-sized embeddings, can generate embeddings from different language models and provides a range of variable-size graphs with consistent homophily. To demonstrate the usefulness of such flexibility, we conduct an extensive ablation study to show the impact of node embedding generation on the GNN model accuracy. We find that using the Roberta NLP embeddings provides better accuracy on GNN models and that a larger embedding dimension further assists in the GNN model accuracy. However, for users who are memory constrained, we show that applying PCA dimensionality reduction can reduce the embedding vectors from 1024-dimensions to 384-dimensions, resulting in a 2.67× reduction in memory footprint with a maximum loss of 3.55% in the GNN model accuracy.

Lastly, this work also discusses the system-level challenges faced when training and inferring GNN models on large graph datasets like IGB. As the dataset size increases, we observe a reduction in the effective GPU utilization due to the increased time spent waiting for the embedding sampling and gathering operation to complete. This is particularly pronounced with the full-scale IGB datasets which require over 1TB of memory space in a system and necessitates memory mapping from the storage. Our profiling shows the existing systems fail to adequately support efficient training and inference of GNN models when the datasets exceed host CPU memory.

Overall this work makes the following key contributions:

(1) We propose IGB, a research dataset tool that innovatively fills the critical gap in labeling, features, heterogeneity, and size of public graph datasets.
(2) We show IGB offers flexible design choices enabling the exploration of different GNN designs, embedding generation schemes, the effect of labeled data, and how system performance evolves with increasing dataset size.

IGB is compatible with popular GNN frameworks like DGL [63] and PyG [19], and comes with several predefined popular models. IGB is available for public usage including raw text used to generate embedding and with a public leaderboard soon [30].

2 BACKGROUND AND MOTIVATION

In this section, we provide a brief overview of GNNs and their applications. We then cover existing GNN datasets and discuss the importance of high-quality, large-scale datasets.

2.1 GNN overview

GNNs are inspired by the idea of extending neural networks that are usually defined for vector inputs to graph-structured inputs. The key idea of GNNs is to pass messages between the nodes in a graph. They use a neural network to propagate information along the edges of the graph while updating the representations of the nodes as the model learns. This enables the GNN to take into account the relationships between the nodes while learning their representations and properties. GNNs can be used to solve a wide variety of tasks such as node classification, link prediction, and graph classification, commonly found in applications such as fraud detection [18, 47, 73], protein structure discovery [29, 57], and recommendation system [61, 70, 71].

GNNs can be applied to both homogeneous graphs and heterogeneous graphs. Homogeneous graphs consist of a single node type and a single relation, while heterogeneous graph consists of multiple types of nodes and multiple relations. Different types of GNN models have been developed to operate on these graphs. For homogeneous graphs, the most popular GNN models are graph convolutional network (GCN) [32], GraphSAGE [24], and graph attention network [59]. These models primarily differ in how they pass messages between the nodes to learn the representation. For heterogeneous graphs, the most popular ones are relational-GCN [51] and relational-GAT(RGAT) [11]. While there are several GNN models, we can formalize all their update functions into a single equation:

$$h_v^k \leftarrow \texttt{UPDATE}(h_v^k, \texttt{SAMPLE}(h_u^{k-1})) \quad (1)$$

Depending on the number of types of relations and neighbor sampling method we can derive respective updated functions of the popular graph neural net models as shown in Table 1. Note that these equations ignore the self-node consideration and the bias term which remains the same for all models. These simplified representations emphasize the differences between these popular models.

2.2 GNN Dataset Generation Techniques

With the increasing popularity of GNNs, researchers have opted to generate GNN datasets based on their needs to simulate and test their model performance. Some of the popular GNN dataset generation techniques include: **(a) Real-world graph datasets:**

Table 1: Popular GNN models' update function comparison.

GCN [32]	$h_v^k \leftarrow \sigma \overbrace{\sum_{u \in \mathcal{N}(v)} \underbrace{(1/c_{uv})\mathbf{W}^k}_{\text{UPDATE}} h_u^{k-1}}^{\text{SAMPLE}}$
GraphSage [24]	$h_v^k \leftarrow \sigma \underbrace{\mathbf{W}^k AGGREGATE}_{\text{UPDATE}} \overbrace{u \in \mathcal{N}'(v) h_u^{k-1}}^{\text{SAMPLE}}$
GAT [59]	$h_v^k \leftarrow \sigma \overbrace{\sum_{u \in \mathcal{N}(v)} \underbrace{\alpha_{uv}\mathbf{W}^k}_{\text{UPDATE}} h_u^{k-1}}^{\text{SAMPLE}}$
R-GCN [51]	$h_v^k \leftarrow \sigma \overbrace{\sum_{r \in \mathcal{R}} \sum_{u \in \mathcal{N}^r(v)} \underbrace{(1/c_{uv})\mathbf{W_r}^k}_{\text{UPDATE}} h_u^{k-1}}^{\text{SAMPLE}}$
R-GAT [11]	$h_v^k \leftarrow \sigma \overbrace{\sum_{r \in \mathcal{R}} \sum_{u \in \mathcal{N}^r(v)} \underbrace{\alpha_{uv}\mathbf{W_r}^k}_{\text{UPDATE}} h_u^{k-1}}^{\text{SAMPLE}}$

This technique employs taking an existing real-world database such as a transaction database or webpages and extracting graph structure from it [26, 31]. **(b) Graph sampling:** This technique involves selecting a subset of nodes and edges from existing large graphs to create a sample dataset [33, 73]. **(c) Synthetic graph generation:** This method creates synthetic graphs by randomly connecting nodes according to certain probability distributions such as power law or by using Kronecker graph generators [13, 43]. This technique can also be used as data augmentation to increase the dataset diversity. IGB datasets are created using real-world datasets and different configurations are created by performing graph sampling operations. We will describe the IGB dataset generation methodology in §3.

2.3 Existing Datasets And Their Problems

The existing dataset collection for GNNs are relatively small in size limiting the ability of GNN to generalize to unseen data [3, 5–9, 14, 17, 22, 25, 33, 34, 39, 42, 50, 52–55, 58, 60, 66, 68, 69, 73]. More recently, work such as OGBN [26, 27, 36] have proposed datasets providing up to 244 million nodes and 1.7 billion edge graphs. These datasets contain a diverse set of graphs and have been widely used by the research community to benchmark GNN's performance.

Table 2 summarizes the various types of publicly disclosed graph datasets. We define flexibility in datasets as the ability to provide various configurations while maintaining homophily. This includes subgraphs with different node-degree distributions, variable-size embeddings, and different language models used to generate embeddings. Homophily refers to the tendency for individual nodes to be connected to other nodes with similar properties. It is fundamental for GNNs because it allows the model to learn a more accurate representation of the nodes in the graph. With datasets that have similar homophily, researchers can better understand and develop GNN models by training with smaller graphs and evaluating accuracy on larger graphs.

Table 2 covers both synthetic and real large graph datasets that are used to study the GNN model performance in both closed and open-source environments. PinSAGE [70] is one of the "few" publicly disclosed proprietary industrial datasets that have more than three billion nodes and eighteen billion edges with an unknown number of labels. Among the real-world open source datasets, MAG240M from OGB-LSC [26] provides the maximum number of nodes to evaluate multi-class node classification tasks. Yet, as noted from Table 2, most existing datasets including OGBN datasets provide a tiny set of labeled data, provide no flexibility and the embeddings are generated in closed source.

As the GNN downstream tasks are often trained as supervised learning tasks, having large labeled data helps in increasing the accuracy of multi-label classification problems. As several prior works have shown [28, 37, 45, 62], with such a small number of usable labeled nodes during training, it becomes challenging to determine if the GNN's low accuracy for unseen data is primarily due to insufficient training data or inability to generalize.

Besides, it is crucial to have access to flexible datasets when training GNN models in order to fully understand the impact of various factors on their accuracy. One key aspect is the generation of embeddings associated with the nodes or edges in the graph. As NLP models are used to generate the embeddings for GNN models, the quality of the NLP model can have a direct effect on the GNN model's accuracy and their downstream tasks. Thus, the datasets should provide mechanisms to study different embedding generation techniques along with methods to vary their proprieties such as node embedding dimensions, and pruning.

Furthermore, as GNNs become increasingly popular, it is crucial to comprehend how their accuracy and efficiency (wall clock time) scale with the size and complexity of the graph. In this regard, researchers are developing optimized hardware and software frameworks tailored to the needs of the GNN applications [2, 21, 23, 35, 75]. However, the lack of flexible datasets hinders their ability to evaluate the scalability of their systems as the datasets grow larger. Ideally, a dataset should provide sub-datasets of varying sizes that maintain the same graph structural properties to enable such comprehensive research on GNN systems. This is because the challenges faced with small datasets differ from those encountered with large datasets.

For instance, training a small IGB graph can be completed within a reasonable time frame as the graph datasets and features can fit within a single system's memory. On the other hand, training a full version of the IGB heterogeneous dataset with a single system with state-of-the-art software stack is currently challenging due to large memory requirements and complex software management techniques. To fully understand the potential and scalability of GNNs, it is essential to study their performance on a range of graph sizes and complexities.

3 IGB DESIGN

Illinois Graph Benchmark (IGB) datasets are designed to address the limitations discussed in § 2.3. IGB datasets are created using

Table 2: Comparison of IGB with the largest publicly disclosed existing dataset. *Labelled nodes as a percentage of total nodes. *input* implies the sizes are dependent on the values provided to the generator.

Dataset	Date	Availability	Type	#Nodes	Labelled*	#Edges	Dim	RawText	Flexibility	Task
PinSAGE [70]	2018	Private	Real	3000 M	UNK.	18 B	1024	No	No	Multi-class classification
papers100M [36]	2020	Public	Real	111 M	< 1%	1.6 B	128	No	No	172-class classification
mag-240m [26]	2021	Public	Real	244 M	< 1%	1.7 B	768	No	No	153-class classification
GraphWorld [43]	2022	Public	Syn	*input*	UNK.	*input*	N.A	No	No	System Design
SemanticScholar [31]	2023	Public	Real	205 M	UNK.	3 B	768	Yes	No	Multi-class classification
SynGen [13]	2023	Private	Syn	*input*	N.A.	*input*	N.A	No	Yes	System Design
IGB-Homogeneous	2023	Public	Real	> 260 M	> 81%	4 B	128 to	Yes	Yes	19 or 2983-class classification
IGB-Heterogeneous	2023	Public	Real	> 547 M	> 40%	6 B	1024	Yes	Yes	and System Design

real-world data extracted from Microsoft Academic Graph [49] and SemanticScholar datasets [31] as discussed in § 3.2.

3.1 IGB Overview

The IGB brings unique opportunities to assist in understanding the impact of dataset creation on the GNN model's accuracy. To this end, IGB addresses following set of challenges:

(1) *Open data licensing and text accesses*: The use of datasets containing graphs, text, and embeddings to train emerging GNN models [51, 67, 72] is on the rise. However, data sources used to create the datasets must have open licensing for ease of adaption and derivative product creation. IGB meets open-data-license requirements (ODC-By-1.0) and provides a collection of datasets containing graphs, text, and embeddings for GNN and language model training.
(2) *Large ground-truth:* IGB offers substantial ground-truth labels extracted from human-annotated data, eliminating the need for manually labeling millions of nodes. The majority of the IGB datasets are fully labeled, and the largest datasets have at least 40% of their nodes labeled. To do this, IGB combines multiple databases to form a large labeled dataset while preserving the accuracy of the information.
(3) *Flexibility for ablation study:* The lack of flexible datasets limits the ability to evaluate the performance of GNN models and understand their execution efficiency with different frameworks and libraries. IGB overcomes this by offering a flexible research tool providing variable-sized embeddings, the ability to generate embeddings from different language models, and a range of variable-sized graphs with consistent homophily. These features enable a better understanding and assessment of GNN models.
(4) *Task complexity*: The IGB includes a range of multi-class classification tasks with varying degrees of complexity, which is crucial for evaluating the capabilities of GNN models. This is because while a GNN model can perform well on a coarser binary classification task, it might not be effective for more fine-grain classification tasks. This feature in IGB also enables the investigation of efficient transfer learning techniques for fine-tuning downstream tasks.

3.2 IGB Dataset Generation Methodology

Generating IGB datasets involves curating data from various sources. Each of the IGB dataset has a graph, ground truth labels, and node embeddings. We will first describe how we generate the graph by extracting information from real-world data.

Input Data: While there are a number of publicly available datasets that can be used as input source [1, 39, 53, 60], many of them are either small or lack the necessary information for building large-scale graph datasets for GNN application. Microsoft Academic Graph (MAG) [49] and SemanticScholar Corpus [31] are two publicly available datasets that meet the criteria for building large graph datasets. The MAG database is particularly useful as it contains a wide range of relationships and information about different types of data points, including papers and authors, as well as information about paper citations, authorship, affiliations, and field of study. SemanticScholar corpus [31], while slightly smaller than MAG, also provides a similar set of functionalities. However, what particularly sets both of these datasets apart is the ***explicit permission we received to release the raw text data under the ODC-By-1.0 licensing scheme***[1].

The MAG database comprises over 260 million entries for papers and about four billion relations representing a citation between two papers. The MAG database schema has many tables among which paper, author, affiliation, URLs, conference/journals, and field of study tables are of interest. The paper table contains information such as title, published date, authors, citations, and names of journals or conferences where the article is published. The paper citations are stored in a file using coordinate (COO) graph storage format where the first paper ID cites the paired paper ID. The author table includes data on the author's name, affiliations, and papers they wrote. SemanticScholar corpus [31] has a similar schema providing up to 205 million entries for papers and about three billion relations representing citation.

IGB Dataset Graphs: IGB provides two types of graphs: homogeneous and heterogeneous. The IGB homogeneous graph (`IGB`) is created by extracting only the paper nodes and the paper-cites-paper relation between these nodes. The IGB heterogeneous graph (`IGBH`) is created by extracting four different types of nodes: papers, authors, institutions, and fields of study. These different types of nodes are connected by various types of edges as shown in Figure 1.

[1]It is important to note that the MAG dataset is now fully deprecated and we thank the MAG database owners for giving us explicit open-sourcing permission.

Table 3: IGB-Homogeneous dataset collection metrics. Maximum memory sizes are reported (embd.).

Dataset	#Nodes	% Labeled	#Edges	Degree	Homophily	Emb-dim	#Classes	Mem.(graph/embd./total)
IGB-tiny	100,000	100	547,416	234/1/5.47	56.79%	128 – 1024	19/2983	6.9 MB/393 MB/400 MB
IGB-small	1,000,000	100	12,070,502	4,292/1/12.07	47.75%	128 – 1024	19/2983	185 MB/4.0 GB/4.1 GB
IGB-medium	10,000,000	100	120,077,694	22,315/1/12.00	59.93%	128 – 1024	19/2983	1.8 GB/39.0 GB/40.8 GB
IGB-large	100,000,000	100	1,223,571,364	73,248/1/12.20	58.27%	128 – 1024	19/2983	19 GB/400 GB/401.8 GB
IGB-full	269,346,174	84.3	3,995,777,033	277,194/1/14.90	51.79%	128 – 1024	19/2983	56 GB/1.1 TB/1.15 TB

Table 4: IGB-Heterogeneous dataset collection metrics. Reported maximum memory sizes with 1024-dim embeddings (embd.).

Dataset	#Total Nodes	#Paper Nodes	#Author Nodes	#Inst. Nodes	#FoS Nodes	#Total Edges	#Classes	Mem.(graph/embd./total)
IGBH-tiny	549,999	100,000	357,041	8,738	84,220	2,062,714	19/2983	31.5 MB/2.19 GB/2.2 GB
IGBH-small	3,131,266	1,000,000	1,926,066	14,751	190,449	26,488,616	19/2983	391 MB/12.16 GB/13 GB
IGBH-medium	25,982,964	10,000,000	15,544,654	23,256	415,054	249,492,193	19/2983	3.8 GB/100.8 GB/104 GB
IGBH-large	217,636,127	100,000,000	116,959,896	26,524	649,707	2,104,750,425	19/2983	30.8 GB/844 GB/874 GB
IGBH-full	547,306,935	269,346,174	277,220,883	26,918	712,960	5,812,005,639	19/2983	85 GB/2.2 TB/2.28 TB

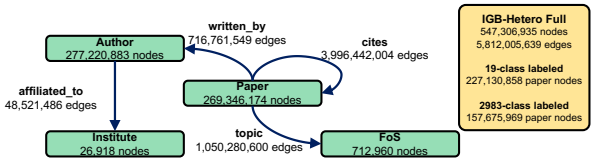

Figure 1: IGB-Heterogeneous dataset schema.

IGB Ground-Truth Labels: The challenge of obtaining a large collection of real-world ground truth labels is a major issue pertaining to dataset generation especially when manual annotation at the million scales is unattainable. Currently, the largest graph datasets such as MAG240M [26] and OGBN-paper100M utilize ArXiv [1] labels and have a maximum of 1.4 million nodes categorized into 172 paper topics and eight subject areas. However, such a tiny set of labeled nodes is insufficient for real-world applications as the model cannot accurately predict unseen data.

To address this, IGB extracts human annotated label information from the MAG [49] and SemanticScholar [31] databases. Extraction of information and aligning two databases pose a challenging problem. First, the two databases are distinct, and merging two databases to create an IGB graph requires to be done carefully. We address this by leveraging the reverse mapping of MAG paper IDs from the SemanticScholar database and merging the databases to create a large-scale IGB graph.

Second, the two databases have different labeling methodologies. To handle this issue, we carefully check each of the labels and create a union of distinct labels from the MAG and SemanticScholar databases. Third, the two databases can have different distinct labels for each paper and can create a merge conflict. We address this merge conflict by carefully checking (a very small number of nodes have this issue) and assigning the right label that appropriately defines the paper from the union of distinct labels from MAG and SemanticScholar databases. Using this merged database, IGB datasets can cover a large portion of the data with labeled nodes. As shown in Table 2, IGB provides more than 81% and 40% of nodes labeled for homogeneous and heterogeneous graphs.

IGB Downstream Tasks: The IGB dataset collections are designed for multi-class classification problems with two different numbers of classes (19 and 2983) depending on the degree of complexity. The 19 class task is curated by combining classes from MAG and SemanticScholar and mapping them into a common structure. Examples of 19-class labels are history, geology, economics, and many more. The 2983 class task is created by bucketing all papers with the same set of paper topics from the set of labels provided by SemanticScholar corpus. Examples of class labels are pediatrics, criminology, and computer engineering. The 19-class task is intended for model development and testing while the 2983-class task can be used to stress test the models and develop robust GNN models for noisy real-world data. Although the IGB dataset comprises node classification problems to solve, it is easy to extend the downstream task to train for edge prediction tasks such as citation recommendation or reviewer recommendation.

IGB Embedding Generation: GNN models operate on graph structure and their embeddings. Embeddings can be associated with a node or an edge and capture the relationship between the nodes and help the GNN modes to extract structural information from the graph. In the past, one-hot vectors and word dictionaries were used to generate these embeddings. However, with the introduction of word2vec embeddings and more recently deep learning-based word and text embeddings, GNNs are being initialized with embeddings generated using foundational language models.

For IGB datasets, we generate embeddings for each node in the graph [2]. Node embeddings are generated by passing the paper titles and abstracts through a Sentence-BERT model [48] and obtaining a 1024-embedding vector for each paper node. Sentence-BERT, based on Siamese network, can generate semantically meaningful sentence embeddings by modifying a pre-trained BERT model [16] such as RoBERTa [38] or GPT [10] while maintaining high accuracy with least run-time overheads.

As IGB graphs are extracted from scientific data [31, 49], it is possible to create domain-specific embeddings instead of general language model-based embeddings. For scientific text, SPECTER [12], an embedding model created using SciBERT [4], a variant of BERT, can be used. SPECTER [12] uses positive and negative samples of the papers from the SemanticScholar corpus [31] to optimize the

[2]Although IGB can generate edge embeddings, from the data management perspective it is quite challenging due to the sheer size of the generated dataset.

embeddings for the scientific text, leading to a 1.5% gain in the F1 score compared to the Sentence-BERT model when tested on papers from the MAG dataset.

For IGB dataset embeddings, we chose to use sentence-BERT embeddings trained on web data, as we want to benchmark the GNN model's ability to extract structural information from the embeddings that do not contain inherent fine-tuning towards a specific domain. This also reflects the practical considerations of real-world industrial settings where fine-tuning language models for each domain-specific task is time-consuming and expensive.

The IGB author node embeddings are generated by taking the average of the node embeddings of all the papers written by the author, a methodology followed in the past work [26]. Institute node embeddings are generated similarly taking the average of the node embeddings of all the authors affiliated with the particular institute. Field of study node embeddings is generated using the topic name provided in the field of study in the database.

3.3 IGB Datasets

The IGB dataset suite offers five datasets for both homogeneous and heterogeneous graphs with varying sizes for deep learning practitioners as shown in Table 3 and Table 4. Each dataset is larger than the previous one (by about an order of magnitude) and is designed for a specific goal. Each of the smaller datasets are created by carefully sampling the graph such that we maintain similar homophily across different dataset variations. Homophily of the IGB dataset varies from 47.75% to 59.93% consistent with the homophily of prior released citation graphs [1, 26]. The tiny dataset is meant for testing and development of GNN models and can be run on a laptop or mobile or an edge device. The small and medium datasets are ideal for training and testing new GNN models during development and can be trained with lower to mid-end GPUs or a powerful CPU. The small and medium datasets are also representative of large-scale labeled datasets currently available to the public [26]. The large dataset requires significant computing resources and can be trained on high-end accelerators and is suitable for building robust GNN models and testing by system designers. The full dataset is a massive dataset for GNN developers to construct practical models and stress test the distributed training platforms.

Each dataset is initialized with 1024-dimensional node embeddings and has two different output classes that increase the difficulty to stress testing the GNN models and optimizing model development. The datasets are randomly split with 60% for training, and 20% each for validation and testing. The dataset also includes the year of publication metadata for every paper node in case the user wants to set a specific splitting rule.

4 IGB DATASET CASE STUDIES

The IGB dataset flexibility enables extensive ablation study to understand the impact of dataset generation technique on the GNN model performance. Through these case studies, we make the following key observations:

- GNN models for node classification tasks observe up to 12.96% boost in their accuracy with large labeled data.

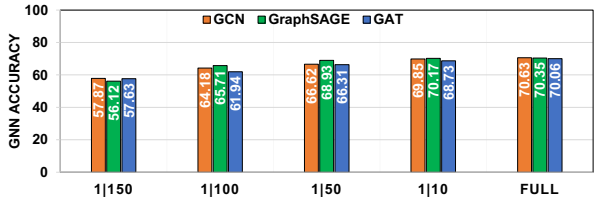

Figure 2: Impact of labeled nodes on GNN model performance using IGB-medium dataset. Having more labeled nodes during training improves the model's accuracy.

- Using an NLP model for initializing node embeddings provides over 40% increase in GNN accuracy over random node embeddings.
- Among the different generic NLP model tests, RoBERTa NLP model provides the best overall performance.
- Reducing the embedding dimension from 1024-dim to 384-dim results in 2.67× memory saving with a maximum 3.55% average loss in the GNN models accuracy.

4.1 Setup

Models: We present performance benchmarks for three commonly used GNN models (GCN [32], GraphSAGE [24], and GAT [59]) on the IGB homogeneous dataset and for three popular GNN models (RGCN [51], RSAGE[3] and RGAT [11]) on the IGB heterogeneous dataset. All models are trained with 0.01 learning rate with two layers. Most of the models are trained to three epochs and numbers are reported. Unless explicitly stated, we used a batch size of 10K to maximize GPU utilization and used IGB-medium as the default dataset. Lastly, most evaluations are reported with homogeneous IGB datasets but are equally applicable to heterogeneous datasets.

System: We conducted our evaluations on a high-performance server system with dual AMD EPYC 7R32 processors, 256GB of DRAM, and an NVIDIA A100-40GB PCIe GPU with NVMe SSDs. The experiments were carried out using the DGLv0.9 framework [63] built on top of PyTorch [44] as obtained from the NVIDIA NGC repository. *To reflect real-world cost considerations in industry, all experiments were run on a single EC2 instance.*

4.2 Impact Of Labeled Nodes

In Section 2.3, we emphasized the importance of labeled data. In this section, the impact of labeled data is evaluated for the IGB medium dataset. To achieve this, sub-datasets with varying fractions of labeled data were created and then trained and evaluated using the GCN, GraphSAGE and GAT models on the 19-class node classification task. 1/150 labeled sub-dataset is representative of the largest publicly available labeled dataset, MAG240M [26]. As shown in Figure 2, the model accuracy improved by up to 10% as more labeled data was added. Increasing the classification task complexity from 19-class to 2983-class shows a similar trend in the performance. This is expected behavior as these models are trained as supervised learning tasks and more labeled data should help in boosting the model's performance.

[3]RSAGE, a GraphSAGE extension to relational graphs is reported for the first time.

Table 5: Impact of labeled nodes on IGB variants. Average accuracy across (R)GCN, (R)SAGE and (R)GAT models are reported for the 1/150 and full labeled (lbl) dataset.

Dataset	# Class	Full-lbl acc	1/150 lbl acc	Diff.
IGB-medium	19	70.34%	58.15%	12.18
IGB-medium	2983	62.35%	49.39%	12.96
IGBH-medium	19	72.35%	61.66%	10.69
IGBH-medium	2983	72.46%	63.26%	9.20

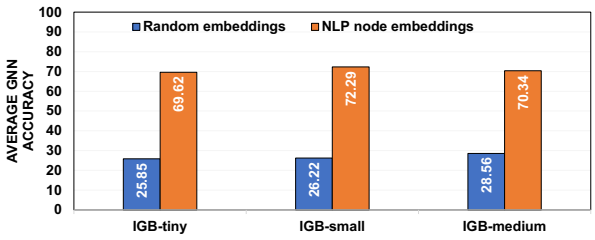

Figure 3: NLP embeddings help the GNN model to learn both structural and node properties while random embeddings can only learn from the structure.

The experiment was also performed on three sub-variants of the homogeneous dataset and on medium size IGB heterogeneous dataset. As shown in Table 5, adding more labeled data helps improve the model performance significantly even with the IGB dataset variants including heterogeneous datasets. This is promising mainly because, with datasets that are 100% labeled, the GNN developer can develop more accurate models and not be constrained by a lack of labeled data.

4.3 Embedding Generation Ablation Study

Embeddings are key to the GNN model's performance but none of the prior work has shown how its generation impacts GNN's performance. In this section, we will discuss the importance of embedding, and its generation process in-depth.

Random vs. NLP embeddings The GNN performance depends on the node information provided to the model, which is initialized as representative node embeddings. In this evaluation, we compared the difference in performance between randomly initialized vectors and RoBERTa [38] NLP-based embeddings on three IGB homogeneous datasets as shown in Figure 3. Each bar graph in Figure 3 is an average accuracy score of three GNN models (GCN, GraphSAGE, and GAT) on the respective dataset. Our results demonstrate that the NLP embeddings significantly increase the model's performance (up to 3×). This is because, with the NLP models, the GNN model learns both the structural information and the node's properties while with random embeddings model can only learn from the graph structure and cannot perform well in the multi-class classification task.

Selecting Right Language Model For Embeddings The quality of embeddings generated by the language model can have a significant effect on the GNN model accuracy. To this end, we evaluated the GNN accuracy using a number of widely used language models (see Table 6) using the sentence transformer package provided by the Huggingface library [64]. Table 6 summarizes the

Table 6: Selected Sentence Transformer Models

Dataset	Emb dim	Avg. Acc	Model size
all-MiniLM-L6-v2	384	58.80%	80 MB
distiluse-base-multilingual	512	45.59%	480 MB
all-mpnet-base-v2	768	63.30%	420 MB
all-roberta-large-v1	1,024	61.64%	1,360 MB

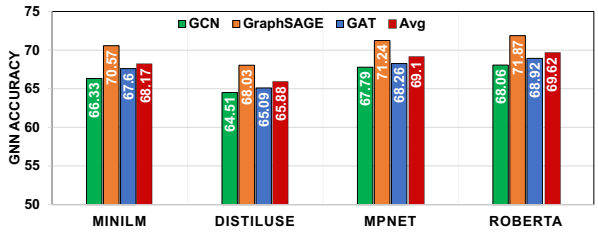

Figure 4: GNN accuracy on IGB-tiny using different NLP embedding models for initializing node embeddings.

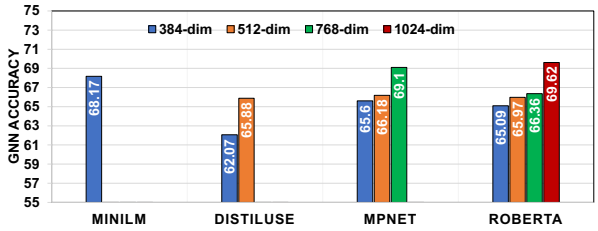

Figure 5: Average GNN accuracy on IGB-tiny dataset using normalized embeddings across different language models. The average accuracy is computed as across the GCN, GraphSAGE, and GAT models.

average language model performance on NLP tasks, their respective model size, and generated text embedding dimensions.

Using these language models, we evaluate the performance of GNN models on the IGB-tiny dataset. To do this, we generated embeddings using each of the language models independently and trained all the GNN models using the respective language embeddings. Figure 4 shows the impact of these language models on the GNN accuracy. The average accuracy is calculated by averaging the reported accuracy of GCN, GraphSAGE, and GAT models. Although with this evaluation, it is not possible to concisely determine if the large performance range is due to the accuracy of the model or the embedding dimensions, it is clear that the language model itself has a direct impact on the GNN performance. In our next evaluation, we will isolate the impact of the model by reducing the dimensions of the embeddings.

Impact Of Embedding Dimension We normalized the different-sized embeddings from various language models using principal component analysis (PCA), a dimensionality reduction technique [20]. We used the GPU implementation of PCA from the NVIDIA cuML

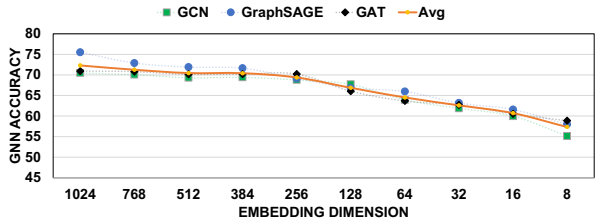

Figure 6: GNN accuracy on RoBERTa [38] embeddings when the dimensions are reduced from 1024 to 8 for IGB-small. Higher embedding dimensions provide better performance.

library [46]. The results of the experiments using GCN, GraphSAGE, and GAT models with varying embedding dimensions are presented in Figure 5 and Figure 6.

From Figure 5, both mpnet [56] and RoBERTa [38] models outperform the other language models while training GNN. This is because both these NLP models are inherently better language models (see Table 6) and their respective accuracy improvements are reflected in the GNN model's accuracy.

From the figures, reducing the embedding dimensions using PCA had an impact on the accuracy across all the GNN models. This is not surprising as the PCA is a lossy pruning technique. For RoBERTa [38] embeddings, reducing the dimensions from 1024 to 384 results in a 3.55%, 1.47%, and 0.58% reduction in accuracy for the GCN, GraphSAGE, and GAT models, respectively. The GAT model was found to be more resilient to reductions in dimensionality when compared to the GraphSAGE and GCN models. A similar observation is noted in other language models.

Reducing the dimensions from 1024 to 384 results in at least 2.67× memory capacity savings during training and inference operation with up to 3.55% loss in accuracy for RoBERTa embeddings. Further reducing the dimensionality offers significant memory footprint reduction but results in a significant drop in the GNN model performance as shown in Figure 6. A similar set of observations are also seen with different IGB datasets including heterogeneous graphs and are not reported. Based on this analysis, the IGB dataset collection provides downloadable embeddings from the RoBERTa language model and provides a toolkit to generate other embeddings in our open-source codebase [30].

4.4 Input Language Influence

Scientific databases such as MAG [49] and SemanticScholar [31] consist of papers that are written in more than 80 languages. For instance, MAG [49] database has more than 12 languages with over 1 million papers and only 50% of the total papers are written in English. In order to understand the impact of language on the accuracy of GNN models, it is important to consider the language models used to generate node embeddings, which are typically trained on web corpus data [10, 38].

Our goal is to determine if the language used in the dataset to train the NLP model makes any difference in the GNN performance. To do this, the study would require a language model trained exclusively on a specific language. However, finding such a pre-trained large language model is currently impossible. As a result, the study used multilingual language models that included or excluded specific languages.

Table 7: Average GNN accuracy with various languages. GNN model accuracy has the least effect on the input language.

Model*	IGB-japanese	IGB-spanish	IGB-french
384 – *eng*	62.89	59.77	60.39
384 – *all*	62.83	59.01	59.48
Average	**62.86**$_{\pm 0.03}$	**59.39**$_{\pm 0.38}$	**59.94**$_{\pm 0.46}$
768 – *eng*	64.82	61.63	61.15
768 – *all*	64.74	61.77	61.25
Average	**64.78**$_{\pm 0.04}$	**61.70**$_{\pm 0.07}$	**61.20**$_{\pm 0.05}$
512 – *v1*	61.82	58.86	57.48
512 – *v2*	61.61	58.42	57.04
Average	**61.72**$_{\pm 0.11}$	**58.64**$_{\pm 0.22}$	**57.26**$_{\pm 0.22}$

Table 8: IGB-Homogeneous datasets benchmark results using the same GNN model parameters (*trained for 3 epochs). `N.G.` stands for cannot finish training in due time.

Model	GCN		SAGE		GAT	
Dataset (# class)	19	2983	19	2983	19	2983
IGB-tiny	68.06	53.13	71.87	59.49	68.92	51.74
IGB-small	70.46	63.28	75.49	68.70	70.93	63.70
IGB-medium*	70.63	62.70	70.35	62.55	70.06	61.81
IGB-large*	50.29	*N.G.*	64.89	*N.G.*	64.59	*N.G.*
IGB-full*	48.59	*N.G.*	54.95	*N.G.*	55.51	*N.G.*

We focused the evaluation on three languages: Japanese, Spanish, and French. Pre-trained multilingual models from the HuggingFace repository [64] were used to create node embeddings. We used six different models, including MiniLM and mpnet, either trained in only English or multiple languages, including English.

We generated two types of embeddings: one using a language model trained in English and the other using a multi-lingual language model that included Japanese, Spanish, and French (multilingual embedding). The results, shown in Table 7, indicate that there is no significant performance improvement achieved by including specific languages, regardless of the model dimensions and language model used. We found similar results when running different embedding dimension models on the French and Spanish datasets. This makes us believe that the GNNs models are likely language agnostic.

4.5 Overall Summary

The overall results of the IGB datasets for all the GNN models with two complex tasks are summarized in Table 8 and Table 9. The IGB-tiny and small datasets are trained for 20 epochs while the rest are trained for three epochs due to the long training time. The accuracy of the models generally decreases in the more complex 2983 tasks as the model must identify finer differences in the nodes to classify them into a large number of classes. We observe up to 14.93% drop in performance for IGB homogeneous dataset. Longer training (more epochs) may help recover some of these accuracy

Table 9: IGB-Heterogeneous datasets benchmark results using the same GNN model parameters(* trained for 3 epochs). `N.G.` stands for cannot finish training in due time.

Model	RGCN		RSAGE		RGAT	
Dataset (# class)	19	2983	19	2983	19	2983
`IGBH-tiny`	66.73	67.66	67.79	68.84	66.80	67.77
`IGBH-small`	71.35	71.64	72.54	72.60	72.61	72.27
`IGBH-medium*`	71.85	71.79	72.65	72.86	72.56	72.74
`IGBH-large*`	*N.G.*	*N.G.*	*N.G.*	*N.G.*	*N.G.*	*N.G.*
`IGBH-full*`	*N.G.*	*N.G.*	*N.G.*	*N.G.*	*N.G.*	*N.G.*

drops, but ideally, better models are needed to handle the finer classification task.

Surprisingly the relational GNN models are tolerant to complex multi-class problems on the heterogeneous dataset. Despite an increase in task complexity from 19 classes to 2983 classes, the accuracy of relational GNN models does not decline. We believe the extra structural information present in the heterogeneous graph enables these models to classify with greater precision. Further investigation is necessary to fully comprehend this phenomenon.

4.6 Existing System Limitations

The existing system is unable to handle the complexity of training GNN models using the full IGB dataset. This is due to the large size of the embedding tables and graphs (see Table 3 and Table 4), which require large memory capacity. Despite the use of state-of-the-art systems and software, training GNN models on the full IGB dataset on a single system is still a challenge and time-consuming task. Let's briefly describe how GNN training occurs before discussing the system limitations.

GNN training operation can be split into three key stages: sampling, aggregation, and computation. The first stage in GNN training is to sample nodes or sub-graphs from the graph to form a mini-batch. In the aggregation stage, information from the neighborhood of each node present in the mini-batch is aggregated to form a node representation. This process involves reading node embeddings and forming an aggregated representation for the node. In the final stage, the aggregated node representation is fed into a neural network to train the model.

As graphs and embeddings come in varying sizes, frameworks such as DGL [63] and PyG [19] provide a different mechanism to optimally place graphs and embeddings in the system for efficient training execution. If the dataset fits in the GPU memory (e.g. IGB(H)-tiny and IGB(H)-small), then both graphs and embeddings can be preloaded to the GPU memory and then GNN models can be trained. Our observation is that, when the datasets fit in the GPU memory, we achieve up to 80% GPU utilization (average 50%) during training.

If the dataset fits in the host CPU memory(e.g. IGB(H)-medium and IGB(H)-large), the graph is either placed in the GPU memory or the host memory, depending on the size of the graph, while the embeddings are loaded in the host memory. During the training operation, the GPU threads can directly sample the graphs from GPU memory and issue zero-copy memory-mapped I/O access to

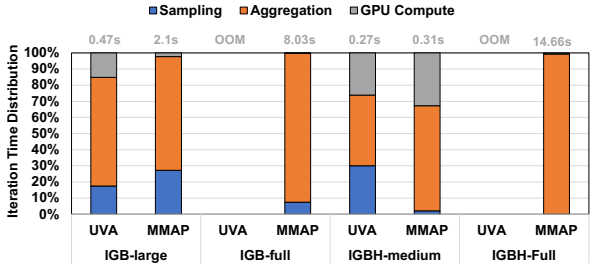

Figure 7: Execution time breakdown of the three stages involved while training GCN and RGCN model. The majority of the time is spent either on the sampling operation or on the aggregation step.

the embeddings using the DGL-UVA [40, 41] technique for efficient execution. Our measurement shows that, in this case, we achieve up to 80% GPU utilization (average 50%).

If the graph and embeddings do not fit in the host memory of a single system, as is the case of the IGB-full dataset, then embeddings can be memory-mapped (`mmap`) and be stored in fast storage medium like NVMe SSDs. The intuition here is that, even though each mini-batch training requires a small working set size (~600MB for IGB(H)-full) in the GPU memory, the entire graph and embeddings need to be accessible during a training operation. This allows the frameworks like DGL [63] and PyG [19] to work on the embedding tables that exceed the host memory capacity of a single system.

Figure 7 illustrates the distribution of the time spent for each of the three stages of training GCN and RGCN models on various IGBs graphs. As shown in Figure 7, the use of `mmap` approach leads to a considerable slowdown, even when the graphs and embeddings fit in the host memory, as much as 4.5× (see IGB-large column). Unlike UVA, the `mmap` abstraction requires the CPU threads to perform the sampling and aggregation stages, accessing the embeddings stored in NVMe SSDs through the operating system page cache, leading to significant overhead. For IGB(H)-full graphs, the node aggregation stage consumes the most significant fraction of iteration time when using the `mmap` approach, causing low average GPU utilization, less than 5%. Profiling reveals that the `mmap` approach can only achieve up to 25% of storage bandwidth (1GBps) and is mainly limited by the system's page fault handler and page-cache throughput.

5 CONCLUSION

This work introduces IGB, a research tool for deep learning practitioners to thoroughly evaluate and test GNN models with accuracy. It provides access to a large graph dataset that includes both homogeneous and heterogeneous graphs, with more than 40% of their nodes labeled. IGB has been designed to be flexible, offering options for the examination of various GNN architectures, embedding generation techniques, and analyzing system performance while training or inferencing GNN models. IGB is open-sourced, compatible with DGL and PyG frameworks, and includes raw text data that can inspire new research at the intersection of natural language processing and graph neural network research topics.

REFERENCES

[1] 2023. ArXiv Bulk data. https://arxiv.org/help/bulk_data

[2] Adam Auten, Matthew Tomei, and Rakesh Kumar. 2020. Hardware Acceleration of Graph Neural Networks. In *2020 57th ACM/IEEE Design Automation Conference (DAC)*. 1–6. https://doi.org/10.1109/DAC18072.2020.9218751

[3] Yunsheng Bai, Hao Ding, Song Bian, Ting Chen, Yizhou Sun, and Wei Wang. 2018. Graph Edit Distance Computation via Graph Neural Networks. *CoRR* abs/1808.05689 (2018). arXiv:1808.05689 http://arxiv.org/abs/1808.05689

[4] Iz Beltagy, Kyle Lo, and Arman Cohan. 2019. SciBERT: A Pretrained Language Model for Scientific Text. (2019). https://doi.org/10.48550/ARXIV.1903.10676

[5] Stephan Bloehdorn and York Sure. 2007. Kernel Methods for Mining Instance Data in Ontologies. In *Proceedings of the 6th International The Semantic Web and 2nd Asian Conference on Asian Semantic Web Conference* (Busan, Korea) *(ISWC'07/ASWC'07)*. Springer-Verlag, Berlin, Heidelberg, 58–71.

[6] L. C. Blum and J.-L. Reymond. 2009. 970 Million Druglike Small Molecules for Virtual Screening in the Chemical Universe Database GDB-13. *J. Am. Chem. Soc.* 131 (2009), 8732.

[7] Aleksandar Bojchevski and Stephan Günnemann. 2017. Deep gaussian embedding of graphs: Unsupervised inductive learning via ranking. *arXiv preprint arXiv:1707.03815* (2017).

[8] Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. *Advances in neural information processing systems* 26 (2013).

[9] Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. 2013. Translating Embeddings for Modeling Multi-relational Data. In *Advances in Neural Information Processing Systems*, C.J. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K.Q. Weinberger (Eds.), Vol. 26. Curran Associates, Inc. https://proceedings.neurips.cc/paper/2013/file/1cecc7a77928ca8133fa24680a88d2f9-Paper.pdf

[10] Paweł Budzianowski and Ivan Vulić. 2019. Hello, It's GPT-2 – How Can I Help You? Towards the Use of Pretrained Language Models for Task-Oriented Dialogue Systems. https://doi.org/10.48550/ARXIV.1907.05774

[11] Dan Busbridge, Dane Sherburn, Pietro Cavallo, and Nils Y Hammerla. 2019. Relational Graph Attention Networks. (2019). arXiv:arXiv:1904.05811 http://arxiv.org/abs/1904.05811

[12] Arman Cohan, Sergey Feldman, Iz Beltagy, Doug Downey, and Daniel S. Weld. 2020. SPECTER: Document-level Representation Learning using Citation-informed Transformers. https://doi.org/10.48550/ARXIV.2004.07180

[13] Sajad Darabi, Piotr Bigaj, Dawid Majchrowski, Pawel Morkisz, and Alex Fit-Florea. 2022. A Framework for Large Scale Synthetic Graph Dataset Generation. https://doi.org/10.48550/ARXIV.2210.01944

[14] Asim Kumar Debnath, Rosa L Lopez de Compadre, Gargi Debnath, Alan J Shusterman, and Corwin Hansch. 1991. Structure-activity relationship of mutagenic aromatic and heteroaromatic nitro compounds. correlation with molecular orbital energies and hydrophobicity. *Journal of medicinal chemistry* 34, 2 (1991), 786–797.

[15] Tim Dettmers, Pasquale Minervini, Pontus Stenetorp, and Sebastian Riedel. 2018. Convolutional 2D Knowledge Graph Embeddings. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence and Thirtieth Innovative Applications of Artificial Intelligence Conference and Eighth AAAI Symposium on Educational Advances in Artificial Intelligence* (New Orleans, Louisiana, USA) *(AAAI'18/IAAI'18/EAAI'18)*. AAAI Press, Article 221, 8 pages.

[16] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. https://doi.org/10.48550/ARXIV.1810.04805

[17] Yuxiao Dong, Nitesh V Chawla, and Ananthram Swami. 2017. metapath2vec: Scalable representation learning for heterogeneous networks. In *Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining*. 135–144.

[18] Yingtong Dou, Zhiwei Liu, Li Sun, Yutong Deng, Hao Peng, and Philip S. Yu. 2020. Enhancing Graph Neural Network-based Fraud Detectors against Camouflaged Fraudsters. *CoRR* abs/2008.08692 (2020). arXiv:2008.08692 https://arxiv.org/abs/2008.08692

[19] Matthias Fey and Jan Eric Lenssen. 2019. Fast Graph Representation Learning with PyTorch Geometric. (2019). https://doi.org/10.48550/ARXIV.1903.02428

[20] Karl Pearson F.R.S. 1901. LIII. On lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science* 2, 11 (1901), 559–572. https://doi.org/10.1080/14786440109462720

[21] Swapnil Gandhi and Anand Padmanabha Iyer. 2021. P3: Distributed Deep Graph Learning at Scale. In *USENIX Symposium on Operating Systems Design and Implementation*.

[22] C. Lee Giles, Kurt D. Bollacker, and Steve Lawrence. 1998. CiteSeer: An Automatic Citation Indexing System. In *Proceedings of the Third ACM Conference on Digital Libraries* (Pittsburgh, Pennsylvania, USA) *(DL '98)*. Association for Computing Machinery, New York, NY, USA, 89–98. https://doi.org/10.1145/276675.276685

[23] Zhangxiaowen Gong, Houxiang Ji, Yao Yao, Christopher W. Fletcher, Christopher J. Hughes, and Josep Torrellas. 2022. Graphite: Optimizing Graph Neural Networks on CPUs through Cooperative Software-Hardware Techniques. In *Proceedings of the 49th Annual International Symposium on Computer Architecture* (New York, New York) *(ISCA '22)*. Association for Computing Machinery, New York, NY, USA, 916–931.

[24] William L. Hamilton, Rex Ying, and Jure Leskovec. 2017. Inductive Representation Learning on Large Graphs. *CoRR* abs/1706.02216 (2017). arXiv:1706.02216 http://arxiv.org/abs/1706.02216

[25] William L. Hamilton, Rex Ying, and Jure Leskovec. 2017. Inductive Representation Learning on Large Graphs. *CoRR* abs/1706.02216 (2017). arXiv:1706.02216 http://arxiv.org/abs/1706.02216

[26] Weihua Hu, Matthias Fey, Hongyu Ren, Maho Nakata, Yuxiao Dong, and Jure Leskovec. 2021. OGB-LSC: A Large-Scale Challenge for Machine Learning on Graphs. https://doi.org/10.48550/ARXIV.2103.09430

[27] Weihua Hu, Matthias Fey, Marinka Zitnik, Yuxiao Dong, Hongyu Ren, Bowen Liu, Michele Catasta, and Jure Leskovec. 2020. Open Graph Benchmark: Datasets for Machine Learning on Graphs. *CoRR* abs/2005.00687 (2020). arXiv:2005.00687 https://arxiv.org/abs/2005.00687

[28] Ziniu Hu, Yuxiao Dong, Kuansan Wang, Kai-Wei Chang, and Yizhou Sun. 2020. GPT-GNN: Generative Pre-Training of Graph Neural Networks. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery And Data Mining* (Virtual Event, CA, USA) *(KDD '20)*. Association for Computing Machinery, New York, NY, USA, 1857–1867. https://doi.org/10.1145/3394486.3403237

[29] Wengong Jin, Jeremy Wohlwend, Regina Barzilay, and Tommi S. Jaakkola. 2022. Iterative Refinement Graph Neural Network for Antibody Sequence-Structure Co-design. In *International Conference on Learning Representations*. https://openreview.net/forum?id=LI2bhrE_2A

[30] Arpandeep Khatua, Vikram Sharma Mailthody, Bhagyashree Taleka, Tengfei Ma, Xiang Song, and Wen-mei Hwu. 2023. IGB Datasets for public release with leaderboard. https://github.com/IllinoisGraphBenchmark/IGB-Datasets

[31] Rodney Kinney, Chloe Anastasiades, Russell Authur, Iz Beltagy, Jonathan Bragg, Alexandra Buraczynski, Isabel Cachola, Stefan Candra, Yoganand Chandrasekhar, Arman Cohan, Miles Crawford, Doug Downey, Jason Dunkelberger, Oren Etzioni, Rob Evans, Sergey Feldman, Joseph Gorney, David Graham, Fangzhou Hu, Regan Huff, Daniel King, Sebastian Kohlmeier, Bailey Kuehl, Michael Langan, Daniel Lin, Haokun Liu, Kyle Lo, Jaron Lochner, Kelsey MacMillan, Tyler Murray, Chris Newell, Smita Rao, Shaurya Rohatgi, Paul Sayre, Zejiang Shen, Amanpreet Singh, Luca Soldaini, Shivashankar Subramanian, Amber Tanaka, Alex D. Wade, Linda Wagner, Lucy Lu Wang, Chris Wilhelm, Caroline Wu, Jiangjiang Yang, Angele Zamarron, Madeleine Van Zuylen, and Daniel S. Weld. 2023. The Semantic Scholar Open Data Platform. https://doi.org/10.48550/ARXIV.2301.10140

[32] Thomas N. Kipf and Max Welling. 2016. Semi-Supervised Classification with Graph Convolutional Networks. *CoRR* abs/1609.02907 (2016). arXiv:1609.02907 http://arxiv.org/abs/1609.02907

[33] Scott P Kolodziej, Mohsen Aznaveh, Matthew Bullock, Jarrett David, Timothy A Davis, Matthew Henderson, Yifan Hu, and Read Sandstrom. 2019. The suitesparse matrix collection website interface. *Journal of Open Source Software* 4, 35 (2019), 1244.

[34] Srijan Kumar, Bryan Hooi, Disha Makhija, Mohit Kumar, Christos Faloutsos, and VS Subrahmanian. 2018. Rev2: Fraudulent user prediction in rating platforms. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*. ACM, 333–341.

[35] Yunjae Lee, Jinha Chung, and Minsoo Rhu. 2022. SmartSAGE: Training Large-Scale Graph Neural Networks Using in-Storage Processing Architectures. In *Proceedings of the 49th Annual International Symposium on Computer Architecture* (New York, New York) *(ISCA '22)*. Association for Computing Machinery, New York, NY, USA, 932–945.

[36] Jure Leskovec and Andrej Krevl. 2014. SNAP Datasets: Stanford Large Network Dataset Collection. http://snap.stanford.edu/data.

[37] Yayong Li, Jie Yin, and Ling Chen. 2022. Informative Pseudo-Labeling for Graph Neural Networks with Few Labels. *arXiv preprint arXiv:2201.07951* (2022).

[38] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A Robustly Optimized BERT Pretraining Approach. https://doi.org/10.48550/ARXIV.1907.11692

[39] Kyle Lo, Lucy Lu Wang, Mark Neumann, Rodney Kinney, and Dan S Weld. 2019. S2ORC: The semantic scholar open research corpus. *arXiv preprint arXiv:1911.02782* (2019).

[40] Seung Won Min, Vikram Sharma Mailthody, Zaid Qureshi, Jinjun Xiong, Eiman Ebrahimi, and Wen-mei Hwu. 2020. EMOGI: Efficient Memory-access for Out-of-memory Graph-traversal In GPUs. https://doi.org/10.48550/ARXIV.2006.06890

[41] Seung Won Min, Kun Wu, Sitao Huang, Mert Hidayetoğlu, Jinjun Xiong, Eiman Ebrahimi, Deming Chen, and Wen-mei Hwu. 2021. PyTorch-Direct: Enabling GPU Centric Data Access for Very Large Graph Neural Network Training with Irregular Accesses. https://doi.org/10.48550/ARXIV.2101.07956

[42] Christopher Morris, Nils M. Kriege, Franka Bause, Kristian Kersting, Petra Mutzel, and Marion Neumann. 2020. TUDataset: A collection of benchmark datasets for learning with graphs. *CoRR* abs/2007.08663 (2020). arXiv:2007.08663 https:

//arxiv.org/abs/2007.08663

[43] John Palowitch, Anton Tsitsulin, Brandon Mayer, and Bryan Perozzi. 2022. GraphWorld: Fake Graphs Bring Real Insights for GNNs. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. ACM. https://doi.org/10.1145/3534678.3539203

[44] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zach DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. PyTorch: An Imperative Style, High-Performance Deep Learning Library. https://doi.org/10.48550/ARXIV.1912.01703

[45] Yizhuo Rao, Xianya Mi, Chengyuan Duan, Xiaoguang Ren, Jiajun Cheng, Yu Chen, Hongliang You, Qiang Gao, Zhixian Zeng, and Xiao Wei. 2021. *Know-GNN: An Explainable Knowledge-Guided Graph Neural Network for Fraud Detection*. 159–167. https://doi.org/10.1007/978-3-030-92307-5_19

[46] Sebastian Raschka, Joshua Patterson, and Corey Nolet. 2020. Machine Learning in Python: Main developments and technology trends in data science, machine learning, and artificial intelligence. *arXiv preprint arXiv:2002.04803* (2020).

[47] Shebuti Rayana and Leman Akoglu. 2015. Collective Opinion Spam Detection: Bridging Review Networks and Metadata. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (Sydney, NSW, Australia) *(KDD '15)*. Association for Computing Machinery, New York, NY, USA, 985–994.

[48] Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. *CoRR* abs/1908.10084 (2019). arXiv:1908.10084 http://arxiv.org/abs/1908.10084

[49] Microsoft Research. 2022. Microsoft Academic Graphs. https://www.microsoft.com/en-us/research/project/microsoft-academic-graph/

[50] Benedek Rozemberczki, Carl Allen, and Rik Sarkar. 2019. Multi-scale Attributed Node Embedding. *CoRR* abs/1909.13021 (2019). arXiv:1909.13021 http://arxiv.org/abs/1909.13021

[51] Michael Schlichtkrull, Thomas N. Kipf, Peter Bloem, Rianne van den Berg, Ivan Titov, and Max Welling. 2017. Modeling Relational Data with Graph Convolutional Networks. (2017). https://doi.org/10.48550/ARXIV.1703.06103

[52] Michael Schlichtkrull, Thomas N Kipf, Peter Bloem, Rianne van den Berg, Ivan Titov, and Max Welling. 2018. Modeling relational data with graph convolutional networks. In *European semantic web conference*. Springer, 593–607.

[53] Prithviraj Sen, Galileo Namata, Mustafa Bilgic, Lise Getoor, Brian Galligher, and Tina Eliassi-Rad. 2008. Collective classification in network data. *AI magazine* 29, 3 (2008), 93–93.

[54] Prithviraj Sen, Galileo Mark Namata, Mustafa Bilgic, Lise Getoor, Brian Gallagher, and Tina Eliassi-Rad. 2008. Collective Classification in Network Data. *AI Magazine* 29, 3 (2008), 93–106.

[55] Oleksandr Shchur, Maximilian Mumme, Aleksandar Bojchevski, and Stephan Günnemann. 2018. Pitfalls of Graph Neural Network Evaluation. *Relational Representation Learning Workshop, NeurIPS 2018* (2018).

[56] Kaitao Song, Xu Tan, Tao Qin, Jianfeng Lu, and Tie-Yan Liu. 2020. MPNet: Masked and Permuted Pre-training for Language Understanding. https://doi.org/10.48550/ARXIV.2004.09297

[57] Hannes Stärk, Octavian-Eugen Ganea, Lagnajit Pattanaik, Regina Barzilay, and Tommi Jaakkola. 2022. EquiBind: Geometric Deep Learning for Drug Binding Structure Prediction. (2022). https://doi.org/10.48550/ARXIV.2202.05146

[58] Kristina Toutanova, Danqi Chen, Patrick Pantel, Hoifung Poon, Pallavi Choudhury, and Michael Gamon. 2015. Representing text for joint embedding of text and knowledge bases. In *Proceedings of the 2015 conference on empirical methods in natural language processing*. 1499–1509.

[59] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2017. Graph Attention Networks. (2017). https://doi.org/10.48550/ARXIV.1710.10903

[60] Alex D Wade. 2022. The Semantic Scholar Academic Graph (S2AG). In *Companion Proceedings of the Web Conference 2022*. 739–739.

[61] Jianling Wang, Kaize Ding, Liangjie Hong, Huan Liu, and James Caverlee. 2020. Next-Item Recommendation with Sequential Hypergraphs. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval* (Virtual Event, China) *(SIGIR '20)*. Association for Computing Machinery, New York, NY, USA, 1101–1110. https://doi.org/10.1145/3397271.3401133

[62] Jiahui Wang, Yi Guo, Xinxiu Wen, Zhihong Wang, Zhen Li, and Minwei Tang. 2020. Improving graph-based label propagation algorithm with group partition for fraud detection. *Applied Intelligence* 50 (10 2020). https://doi.org/10.1007/s10489-020-01724-1

[63] Minjie Wang, Da Zheng, Zihao Ye, Quan Gan, Mufei Li, Xiang Song, Jinjing Zhou, Chao Ma, Lingfan Yu, Yujie Gai, Tianjun Xiao, Tong He, George Karypis, Jinyang Li, and Zheng Zhang. 2019. Deep Graph Library: A Graph-Centric, Highly-Performant Package for Graph Neural Networks. *arXiv: Learning* (2019).

[64] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2019. HuggingFace's Transformers: State-of-the-art Natural Language Processing. https://doi.org/10.48550/ARXIV.1910.03771

[65] Zonghan Wu, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and Philip S. Yu. 2021. A Comprehensive Survey on Graph Neural Networks. *IEEE Transactions on Neural Networks and Learning Systems* 32, 1 (2021), 4–24. https://doi.org/10.1109/TNNLS.2020.2978386

[66] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. 2018. How Powerful are Graph Neural Networks? *CoRR* abs/1810.00826 (2018). arXiv:1810.00826 http://arxiv.org/abs/1810.00826

[67] Michihiro Yasunaga, Antoine Bosselut, Hongyu Ren, Xikun Zhang, Christopher D. Manning, Percy Liang, and Jure Leskovec. 2022. Deep Bidirectional Language-Knowledge Graph Pretraining. In *Neural Information Processing Systems (NeurIPS)*.

[68] Hao Yin, Austin R. Benson, Jure Leskovec, and David F. Gleich. 2017. Local Higher-Order Graph Clustering. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (Halifax, NS, Canada) *(KDD '17)*. Association for Computing Machinery, New York, NY, USA, 555–564.

[69] Rex Ying, Dylan Bourgeois, Jiaxuan You, Marinka Zitnik, and Jure Leskovec. 2019. GNN Explainer: A Tool for Post-hoc Explanation of Graph Neural Networks. *CoRR* abs/1903.03894 (2019). arXiv:1903.03894 http://arxiv.org/abs/1903.03894

[70] Rex Ying, Ruining He, Kaifeng Chen, Pong Eksombatchai, William L. Hamilton, and Jure Leskovec. 2018. Graph Convolutional Neural Networks for Web-Scale Recommender Systems. *CoRR* abs/1806.01973 (2018). arXiv:1806.01973 http://arxiv.org/abs/1806.01973

[71] Minji Yoon, Théophile Gervet, Baoxu Shi, Sufeng Niu, Qi He, and Jaewon Yang. 2021. Performance-Adaptive Sampling Strategy Towards Fast and Accurate Graph Neural Networks. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery and Data Mining* (Virtual Event, Singapore) *(KDD '21)*. Association for Computing Machinery, New York, NY, USA, 2046–2056. https://doi.org/10.1145/3447548.3467284

[72] Seongjun Yun, Minbyul Jeong, Raehyun Kim, Jaewoo Kang, and Hyunwoo J. Kim. 2019. Graph Transformer Networks. https://doi.org/10.48550/ARXIV.1911.06455

[73] Shijie Zhang, Hongzhi Yin, Tong Chen, Quoc Viet Hung Nguyen, Zi Huang, and Lizhen Cui. 2020. GCN-Based User Representation Learning for Unifying Robust Recommendation and Fraudster Detection. *CoRR* abs/2005.10150. arXiv:2005.10150 https://arxiv.org/abs/2005.10150

[74] D. Zheng, C. Ma, M. Wang, J. Zhou, Q. Su, X. Song, Q. Gan, Z. Zhang, and G. Karypis. 2020. DistDGL: Distributed Graph Neural Network Training for Billion-Scale Graphs. In *2020 IEEE/ACM 10th Workshop on Irregular Applications: Architectures and Algorithms (IA3)*. IEEE Computer Society, Los Alamitos, CA, USA, 36–44.

[75] Hongkuan Zhou, Ajitesh Srivastava, Hanqing Zeng, Rajgopal Kannan, and Viktor Prasanna. 2021. Accelerating Large Scale Real-Time GNN Inference Using Channel Pruning. *Proc. VLDB Endow.* 14, 9 (oct 2021), 1597–1605.

[76] Jie Zhou, Ganqu Cui, Shengding Hu, Zhengyan Zhang, Cheng Yang, Zhiyuan Liu, Lifeng Wang, Changcheng Li, and Maosong Sun. 2020. Graph neural networks: A review of methods and applications. *AI Open* 1 (2020), 57–81.

Received 2 February 2023