Democratizing Large Language Model-Based Graph Data Augmentation via Latent Knowledge Graphs

Anonymous Author(s)
Affiliation
Address
email

Abstract

Data augmentation is necessary for graph representation learning due to the scarcity and noise present in graph data. Most of the existing augmentation methods overlook the context information inherited from the dataset as they rely solely on the graph structure for augmentation. Despite the success of some large language model-based (LLM) graph learning methods, they are mostly white-box which require access to the weights or latent features from the open-access LLMs, making them difficult to be democratized for everyone as existing LLMs are mostly closed-source for commercial considerations. To overcome these limitations, we propose a black-box context-driven graph data augmentation approach, with the guidance of LLMs — **GPT-Aug**. Leveraging the text prompt as context-related information, we task the LLM with generating knowledge graphs (KGs), which allow us to capture the structural interactions from the text outputs. We then design a dynamic merging schema to stochastically integrate the LLM-generated KGs into the original graph during training. To control the sparsity of the augmented graph, we further devise a granularity-aware prompting strategy and an instruction fine-tuning module, which seamlessly generates text prompts according to different granularity levels of the dataset. Extensive experiments on various graph learning tasks validate the effectiveness of our method over existing graph data augmentation methods. Notably, our approach excels in scenarios involving electronic health records (EHRs), which validates its maximal utilization of contextual knowledge, leading to enhanced predictive performance and interpretability.

1 Introduction

Graph representation learning has received increasing attention in recent years. It achieves great success in solving tasks where relational features are important, such as recommendation systems [2, 54], citation networks [21], and medical records analysis [5, 40]. However, the scarcity and noise present in graph data pose great challenges for effective graph learning, necessitating the development of graph data augmentation algorithms.

Existing graph data augmentation methods focus on graph structures for data augmentation, such as randomly dropping nodes or edges, adding Gaussian noise to the node or edge attributes, or applying graph-based transformations such as sub-sampling and node permutation. While these methods have demonstrated some successes in graph representation learning scenarios, they do not consider the *context* or *attributes* associated with the graph data. This prompts some recent works [30, 57, 19, 26, 68, 70, 87] which leverage LLM for graph representation learning.

Submitted to 38th Conference on Neural Information Processing Systems (NeurIPS 2024). Do not distribute.

Despite their success, they are mostly white-box which require access to the weights or latent features from the LLMs, making them difficult to be democratized as existing LLMs are mostly closed-source for commercial considerations. As a result, the resulting augmented graph becomes less identifiable due to a lack of contextual guidance. Furthermore, most of these augmentation methods leverage in-domain knowledge under a close-world setting, which does not borrow the vast repositories of knowledge in the open world. Additionally, the sparsity of the augmented graph is not well studied, although some methods, such as DropEdge, attempt to sparsify the graph for augmentation. Without proper sparsity control, the augmented graph would be over-sparsified and likely reduced to trivial graphs (i.e., uninformative graphs). These limitations pop the necessity of developing a new graph data augmenter under open-world settings with proper sparsity control, such that the augmented graph can be closer to the true data distribution.

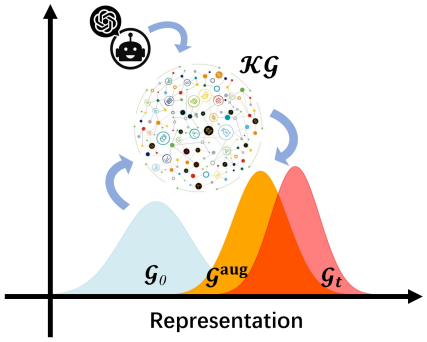

Figure 1: Schematic illustration of the feature distribution of original graph $\mathcal{G}_0$ from observations and $\mathcal{G}^{\text{aug}}$, which represents the augmented graph for $\mathcal{G}_0$ after merging the context knowledge in terms of $\mathcal{KG}$. After performing graph data augmentation with LLM-guided GPT-Aug, $\mathcal{G}^{\text{aug}}$ is closer to the true representation $\mathcal{G}_t$.

In light of the vast development of large language models (LLMs), we propose a novel framework, namely **GPT-Aug**, to perform contextual graph data augmentation with a generative pretrained LLM. Our contributions can be summarized as (1) We introduce a black-box method which leverages extensive knowledge from LLM to perform graph data augmentation without access to model weights or source codes. This is particularly realistic when most LLMs are provided in close-source commercial APIs, enabling the democratization of LLM-based methods. We adopt latent KGs to capture the structural interactions from the text outputs, as well as a compatible data structure for graph data. (2) We design a dynamic merging strategy to stochastically integrate the LLM-generated KGs into the raw graph data during the network training, which guides the optimization trajectory with contextual knowledge. (3) To tackle the sparsity induced by generated KGs, we design a granularity-aware prompting strategy to control the sparsity while maximizing the utility of domain knowledge. Also, we leverage a sequential prompting with instruction fine-tuning strategy to incentivize the LLM to generate the most relevant concepts to the context, and hence high-quality KGs. (4) Extensive experiments on various graph learning tasks validate the effectiveness of our method over existing graph data augmentation methods. (5) Our method demonstrates high scalability across datasets ranging from small to large-scale, consistently delivering satisfactory performance. Notably, our approach excels in scenarios involving electronic health records (EHRs), where our method maximizes the utilization of contextual information, and leads to enhanced predictive performance and interpretability. Codes are anonymously available at `https://anonymous.4open.science/r/GPT-Aug`.

2 Related Works

Graph Neural Networks (GNNs). GNNs are gaining significant success in many problem domains [34, 24, 38, 3, 55, 71]. They learn node representation by aggregating information from the neighboring nodes on the graph topology. Most of the existing GNN architectures are on homogeneous graphs [69, 62, 73, 84]. There are also GNN architectures operating on heterogeneous graphs to learn its enriched structural information and complex relations [66, 24, 25, 79, 52] . However, due to limited samples, it is difficult to approximate the true data distribution, especially in the graph domain. Hence, an effective graph data augmentation algorithm is needed to boost the performance of GNNs.

Graph Data Augmentation. Graph data augmentation (GDA) aims to enhance the utility of the input graph data and produce graph samples close to the true data distribution to alleviate the finite sample bias [7]. Most of the existing works focus on perturbating the graph structures or node features/labels to achieve augmentation, such as node dropping [12], edge perturbation [49, 63], graph rewriting [67, 76, 13], graph sampling [17, 18, 48], graph diffusion [59, 91, 48, 46] or pseudo-labelling [86]. There are also works that adopt a learnable graph data augmenter and design specific losses for

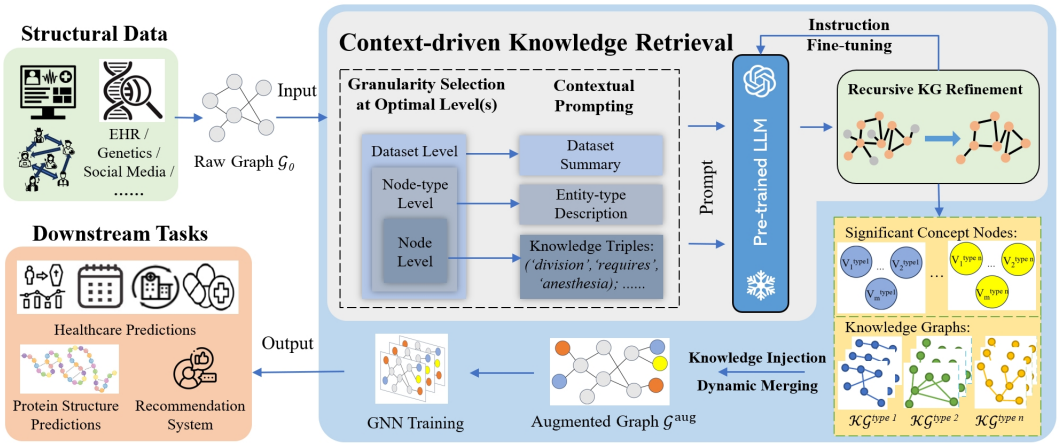

Figure 2: **Overview of our proposed GPT-Aug framework.** Given a dataset, we first construct a graph $\mathcal{G}_0$ to highlight the relational information, and then perform context-driven knowledge retrieval by utilizing the original dataset and a frozen generative pre-trained LLM. We conduct contextual, adaptive, sparsity-controllable and granularity-aware prompt learning on the LLM, thus obtaining either concept-specific KGs or important extra concept nodes at different levels after refinement. For the original graph $\mathcal{G}_0$, we perform graph data augmentation with the domain-knowledge injection procedure. We train a GNN model on the augmented graph $\mathcal{G}^{\text{aug}}$, thus our framework is able to handle a wide range of downstream tasks across various domains depending on the original datasets.

training [82, 71, 37, 56, 36, 47]. However, these methods mainly focus on the graph structures without considering the contextual information or introducing open-world knowledge. Recent works [30, 19, 70, 87, 68, 58] on LLM-based GDA have achieved promising improvements. However, current LLM-based methods are mostly white-box which require access to the weights or latent features from the LLMs. It is computationally inefficient and impractical, as SOTA LLMs are costly for large-scale experiments and often closed-source. Moreover, these methods mostly focus on node-level context and neglect the higher-order graph structures. Hence, a black-box LLM-based GDA framework with awareness of higher-level graph structure is needed to address these limitations.

Graph Learning in Healthcare. Knowledge distillation from massive EHRs has been a popular topic in healthcare informatics. To address the longitudinal features in the EHR data, several early works [39, 42, 41] attempted to learn the EHR features with recurrent neural networks. Since the EHR data represent relational information between entities (e.g., patients make visits), graphical models turn out to be an ideal approach for representing the EHR data [4, 5]. GRAM [4] is a well-known method that learns robust medical code representations by adopting a graph-based attention mechanism. However, a critical gap remains in these methods: they do not fully incorporate the rich contextual information available in EHR data [20, 6]. This oversight can lead to a lack of nuanced understanding of patient data, impacting the accuracy and applicability of the insights derived [10]. Furthermore, there is a notable absence of effective regularization mechanisms for adjusting to the inherent noise in EHR data, which is cluttered with irrelevant or redundant information.

3 Preliminaries

Graphs. A graph $\mathcal{G}$ is a collection of vertices $\mathcal{V}$ and edges $\mathcal{E}$, typically represented as $\mathcal{G} = (\mathcal{V}, \mathcal{E})$. Each edge $e \in \mathcal{E}$ is an ordered or unordered pair of vertices representing the connection between them. In the context of graph neural networks, each vertex v_i is often associated with a feature vector x_i in the feature space $\mathcal{X}$. A knowledge graph (KG) is a specialized type of graph denoted as $\mathcal{KG} = (\mathcal{V}, \mathcal{E}, \mathcal{R})$, where $\mathcal{R}$ is a set of relation types. A KG can be constructed from a set of triples $\mathcal{T} = \{(h_i, r_i, t_i)\}_{i=1}^{|\mathcal{T}|}$ where h_i, t_i, and r_i are the i-th head and tail nodes respectively, and r_i is the relation type for the i-th triple.

Graph Data Augmentation (GDA). Given $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, GDA aims to derive an augmented graph $\mathcal{G}^{\text{aug}} = (\mathcal{V}^{\text{aug}}, \mathcal{E}^{\text{aug}})$, where $\mathcal{V}^{\text{aug}}$ and $\mathcal{E}^{\text{aug}}$ represent the augmented set of nodes and edges, respectively. The augmentation process should preserve or enhance the inherent structure and properties of $\mathcal{G}$, while facilitating improved performance of a GNN (denoted as $\mathcal{M}$) on downstream tasks.

4 Methodology

Our proposed framework consists of two main modules: a knowledge graph construction module with leveraging knowledge from LLMs, and a graph data augmentation module with dynamic knowledge injection. Figure 2 and Algorithm 1 provide an overview of the workflow of our framework.

4.1 Context-Driven Knowledge Retrieval

General Prompting Strategy. The cornerstone of our framework is the construction of KGs using LLMs. The context-aware KGs serve as enriched contextual domain knowledge that augments the original graph $\mathcal{G}_0$ towards the true representation $\mathcal{G}_t$. The KG construction is facilitated through a prompting mechanism that steers the LLM toward generating subgraphs focused on specific concepts. The generation process in general can be formulated as $\mathcal{T} \leftarrow \text{LLM}(\text{prompt})$, where $\mathcal{T} = \{(h_i, r_i, t_i)\}_{i=1}^{|\mathcal{T}|}$ represents the set of triples indicating the relationships between generated concepts. A knowledge graph $\mathcal{KG}$ can then be constructed from $\mathcal{T}$. We design modularized prompts (with placeholders for the descriptions) that are based on all the available information (e.g., the summary of datasets, task descriptions) of the working graph dataset, such that context knowledge can be maximally utilized. One example of the prompting design on the EHR context is:

Algorithm 1 The training workflow of our graph data augmentation method.

Input: Original graph $\mathcal{G}_0 = (\mathcal{V}_0, \mathcal{E}_0)$ with randomly-initialized node features $\{x_i, \forall i \in \mathcal{V}\}$, granularity level s, number of KGs generated K (per step), ground truth labels y.
Output: Augmented graph $\mathcal{G}^{\text{aug}}$, trained GNN model $\mathcal{M}$.
Initialize $\mathcal{G}^{\text{aug}} = \mathcal{G}_0$
for each epoch **do**
    $\mathcal{V}^{\mathcal{KG}} \leftarrow$ Get concept nodes as augmentation entities,
    $\{\mathcal{KG}\}_{i=1}^{K} \leftarrow$ Load KGs from $\mathcal{V}^{\mathcal{KG}}$,
    $\{\mathcal{KG}\}_{i=1}^{K} \leftarrow$ Perform instruction fine-tuning with customized sparsity control on $\{\mathcal{KG}\}_{i=1}^{K}$,
    $\mathcal{G}^{\text{aug}} \leftarrow \text{merge\_KG}(\{\mathcal{KG}\}_{i=1}^{K}, \mathcal{G}^{\text{aug}})$,
    Update node indices for all node types in $\mathcal{G}^{\text{aug}}$,
    Get prediction from the GNN $\hat{y} = \mathcal{M}(\mathcal{G}^{\text{aug}})$,
    Compute training loss $\mathcal{L}(\hat{y}, y)$,
    Backpropagate $\mathcal{L}$ to $\mathcal{M}$
end for
return Trained GNN $\mathcal{M}$

```
Start with the following prompt on a given medical concept (such as
health condition/treatment procedure/drug) and generate an extensive
array of associated connections based on your domain knowledge. These
connections should help improve prediction tasks in healthcare, e.g.
drug recommendation, mortality prediction, length of stay and
readmission prediction.
Format each association as [ENTITY 1, RELATIONSHIP, ENTITY 2],
ensuring the sequence reflects the direction of the relationship. Both
 ENTITY 1 and ENTITY 2 are to be nouns. Elements within [ENTITY 1,
RELATIONSHIP, ENTITY 2] must be definitive and succinct.
Approach in both breadth and depth. Continue expanding [ENTITY 1,
RELATIONSHIP, ENTITY 2] combinations until reaching a total of 100.
{example}
prompt: {descriptions}
updates:
```

where the variables as placeholders are inside {} — {example} provides an exemplar triple format, {descriptions} provides the contextual information, and "updates:" prompts the LLM to finish the paragraph. This prompt initially instructs the LLM to identify and generate concept entities $\mathcal{V}^{\mathcal{KG}}$ and their interrelations $\mathcal{E}^{\mathcal{KG}}$ driven by the descriptions (e.g., on the dataset or entity) and oriented to the target tasks. Subsequently, the LLM regularizes these relationships into standardized triple formats. Finally, the above prompt expands this structured information both in width and depth, digging into more meaningful and nested relationships, until a pre-defined number of triples is reached. We also prompt example triples to regularize the output formats of $\mathcal{T}$. This multi-step process ensures that the KG is both information-rich and aligned with domain-specific objectives. Notably, this paradigm utilizing placeholders avoids manual prompt customization, thereby reducing human labor costs.

Granularity-Aware Prompting for Sparsity Control. Naively utilizing the prompting strategy in the previous section would mostly lead to a sparse KG, where data points are unevenly distributed with

many gaps or missing links. Hence, we propose a multi-layer augmentation strategy that determines a granularity level prior to generation, such that the sparsity of the KG can be controlled.

Granularity refers to the data scale of detail in the augmentation process, ranging from coarse-grained dataset-level to fine-grained node-level information. Based on the availability of information in the working dataset, we define s as the sparsity level parameter (s increases as the data are more fine-grained), and separate the prompting strategy into three granularity levels, $s_0 < s_1 < s_2$, as follows:

- **Dataset-level Augmentation ($s = s_0$).** At the dataset level, our objective is to identify and propagate overarching themes and concepts that are broadly relevant across the dataset. This macro approach involves curating concepts and triples that reflect high-level semantics and dependencies. This is the most fundamental form of our method since dataset-level information is always available.
- **Type-level Augmentation ($s = s_1$).** Another common scenario is that we have node type level information (e.g., class labels in texts for classification). We distill the most salient concepts and relationships pertinent to each class or node type. By doing so, we gain an in-depth understanding of the node categories, fleshing out their characteristics and the interconnections within them. A node-type level prompting example on the Cora dataset (7 classes) is provided in the appendix.
- **Node-level Augmentation ($s = s_2$).** In some scenarios (e.g., EHR datasets), we have the finest information (e.g., text description) on each node (or medical entity). At this juncture, we aim to enrich individual nodes with highly relevant and specific concepts that are crucial for the particular tasks. This targeted augmentation ensures that nodes are imbued with unique attributes that can drive predictive tasks more effectively.

Concept Pruning via Instruction Fine-tuning. Due to the high complexity of given tasks, LLM's one-time retrieval of KGs may contain low-entropy (i.e., uninformative) concepts (e.g., *is*, *dataset*, or *disease*). We thus instruct LLMs to go through a chain-of-thought process to do multi-stage reasoning and self-improve the quality of KGs. Figure 3 illustrates our concept prompting procedure via instruction fine-tuning. Given the initial generated $\mathcal{KG}$, we refine it by recursively calling the LLM and pruning less relevant nodes and edges, while ensuring that a predefined percentage of the concepts are directly derived from the original dataset. A template for this instruction fine-tuning (IFT) process is given below (we use EHR as an illustrative example). After this procedure, a set of important concept nodes $\mathcal{V}^{\mathcal{KG}}$ is then output for triple construction and KG generation.

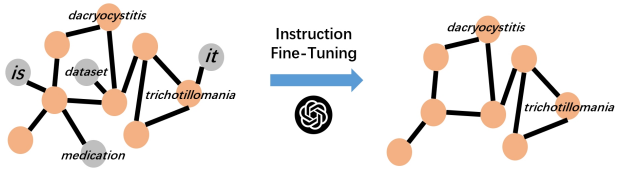

Figure 3: Concept pruning via instruction fine-tuning, where trivial concepts can be pruned by re-prompting the coarse set of concepts to the LLM.

```
Given the list of triples augmented with MIMIC-III dataset, I want to
select '{number_of_concepts}' most important triples from the list.
The importance of a triple is based on your knowledge and inference on
 how it will help improve prediction tasks in healthcare, e.g. drug
recommendation, mortality prediction, length of stay, readmission
prediction. If you think a triple is important, please keep it.
Otherwise, please remove it. You can also add triples from your
background knowledge.
triples: {triples}
updates:
```

4.2 Augmentation with Generated KGs

Dynamic Graph Merging. Given a constructed $\mathcal{KG}$ from $\mathcal{T}$ on a sparsity level s, we design a dynamic merging schema to merge $\mathcal{KG}$ into $\mathcal{G}_0$. This allows the model to see more augmented

samples $\mathcal{G}^{\text{aug}}$ as a different merged graph is obtained in each optimization step. For each concept node $v_c \in \mathcal{V}^{\mathcal{KG}}$ in $\mathcal{KG}$, we select a subset of nodes $\mathcal{V}_0^s = \{z | z \in \mathcal{V}_0\}_{i=0}^{n_c} \subseteq \mathcal{V}_0$, where n_c is the predetermined number of edges per concept node. We connect the concept nodes and the selected nodes from $\mathcal{V}_0^s$ to obtain an edge set

$$\mathcal{E}^{\text{conn}} = \{(v_c, z) | \forall v_c \in \mathcal{V}^{\mathcal{KG}}, z \in \mathcal{V}_0^s\}.$$

After that, the augmented graph $\mathcal{G}^{\text{aug}} = (\mathcal{V}^{\text{aug}}, \mathcal{E}^{\text{aug}})$ can be obtained by joining the edge sets and node sets, i.e., $\mathcal{E}^{\text{aug}} = \mathcal{E}^{\text{conn}} \cup \mathcal{E}_0 \cup \mathcal{E}^{\mathcal{KG}}$ and $\mathcal{V}^{\text{aug}} = \mathcal{V}_0 \cup \mathcal{V}^{\mathcal{KG}}$.

This dynamic merging is not a one-off operation but an iterative process. Each training epoch sees the refreshment of KGs based on the model's current state, thereby keeping the graph data dynamic and contextually rich. As the model training proceeds, it continually refines the edge weights and node features based on the newly incorporated KGs. This iterative update ensures that the model does not overfit and generalizes well on unseen data.

Due to the computation limitations, the number of LLM inferences is limited. Therefore, we precompute $\mathcal{KG}$ offline and merge it with $\mathcal{G}_0$ stochastically during training. Under sufficient computational conditions, the dynamic merging schema allows for online prompting where an up-to-date $\mathcal{KG}$ can be generated after every optimization step. On the other hand, the LLM can also be fine-tuned online with task-specific losses. This allows for more context-related KG generations and hence improved data augmentation performance. It also enables the potential for training open-world GNN models.

Training Paradigm. We use GNN to predict the labels with the augmented graph as the input, $\hat{y} = \mathcal{M}(\mathcal{G}^{\text{aug}})$. We benchmark with different choices of $\mathcal{M}$: graph convolutional network (GCN) [69], graph attention network (GAT) [62], GraphSAGE [17], and graph isomorphism network (GIN) (detailed formulations and descriptions of GNNs in appendix). We compute the loss for backpropagation with the predictive labels. For instance, in a multi-class classification task, we adopt the cross-entropy loss, $L_{\text{ce}} = -\frac{1}{N}\sum_{i=1}^{N}\sum_{c=1}^{C} y_{i,c} \log(\text{softmax}(z_{i,c}))$, where $y_{i,c}$ is the ground truth label for patient i and class c, N is the number of observations, C is the number of classes, and $z_{i,c}$ is logits obtained from the model.

4.3 Adaptability to Other Graph Datasets

Since EHR contains enriched contextual information that allows for flexible prompting design, we use the EHR dataset to illustrate our prompting strategy. However, our prompting strategy is adaptable to other graph datasets, as the placeholders in the modularized prompts can be replaced by information on the target datasets. We can also incrementally enlarge the KG such that knowledge from the existing domain can be leveraged to the target domain. We employ a highly-adaptive customization strategy that tailors the prompt structure based on the specific dataset in use. This strategy includes understanding the data's content and structure and then adjusting the prompts to ensure the generated KGs are optimally suited for the data in question.

5 Experiments

5.1 Experimental Settings

Datasets and Tasks. (1) We perform experiments on **generic** graph benchmarks (Cora, PPI, Actor, and Citeseer), where we benchmark our method on node classification tasks. (2) We validate the scalability of GPT-Aug on two **large-scale** datasets — OGBN-products and OGBN-arxiv [21] against additional LLM-based methods. Table 8 and 9 provide a summary of these graph datasets from small to large-scales. (3) Additionally, we highlight an application of our method on a **large-scale EHR** dataset — MIMIC-III [32]. It contains a publicly available dataset of 46,520 intensive care unit (ICU) patients over 11 years. We perform four supervised tasks — in-hospital mortality prediction (MORT), readmission prediction (READM), length of stay (LOS) prediction, and drug recommendations (DR), where MORT and READM predictions are approached as binary classification tasks, LOS prediction as a multi-class classification task, and DR as a multi-label classification task. Since the lab events are sparse and introduce heavy noise, we exclude them when constructing the graph. Table 10 in the appendix presents a summary of the types and counts of the entities in the MIMIC-III dataset, and the details of each task.

Table 1: Node classification performance (in common metrics of existing literature) on generic graph datasets with different GNN architectures. Standard deviations are shown in brackets.

GNN Archi.	Augmenter	PPI Micro-F1	Actor Accuracy	Cora Accuracy	Citeseer Accuracy
Graph SAGE	None	60.0 (2.7)	36.7 (1.8)	81.0 (3.3)	70.9 (2.0)
	DropNode [12]	61.5 (2.6)	36.8 (1.5)	80.6 (3.2)	70.1 (2.7)
	DropEdge [50]	63.2 (3.1)	36.8 (2.9)	80.4 (2.8)	71.2 (3.2)
	RandomwalkPE [8]	63.1 (2.7)	37.7 (2.7)	81.2 (3.1)	70.8 (2.6)
	LaplacianPE [9]	63.5 (3.1)	36.7 (2.1)	80.9 (2.2)	70.7 (2.5)
	GPT-Aug (Ours)	**93.6 (2.3)**	**37.9 (1.6)**	**83.3 (1.2)**	**72.6 (2.0)**
GAT	None	97.1 (3.0)	30.3 (2.7)	82.1 (4.3)	72.1 (3.7)
	DropNode [12]	94.0 (3.4)	31.3 (2.2)	80.7 (3.7)	71.9 (3.2)
	DropEdge [50]	85.1 (3.0)	31.2 (3.0)	78.9 (3.9)	69.1 (3.9)
	RandomWalkPE [8]	90.8 (3.6)	31.4 (2.5)	81.2 (3.2)	71.9 (3.2)
	LaplacianPE [9]	90.7 (2.7)	30.9 (2.9)	81.4 (2.4)	71.8 (2.7)
	GPT-Aug (Ours)	**97.2 (3.4)**	**32.2 (2.3)**	**83.6 (2.0)**	**73.1 (2.2)**
GCN	None	53.2 (2.4)	29.8 (2.1)	81.0 (2.7)	69.4 (2.0)
	DropNode [12]	58.9 (1.9)	28.7 (2.5)	78.9 (2.6)	70.5 (2.0)
	DropEdge [50]	54.8 (4.1)	28.9 (3.4)	82.4 (3.5)	71.3 (3.2)
	RandomWalkPE [8]	59.0 (1.6)	29.8 (2.9)	80.0 (2.9)	71.6 (2.2)
	LaplacianPE [9]	59.3 (1.6)	29.6 (2.2)	80.0 (1.9)	71.1 (2.1)
	GPT-Aug (Ours)	**60.3 (1.2)**	**32.4 (2.3)**	**82.9 (1.0)**	**73.1 (1.1)**
GIN	None	70.3 (2.8)	31.9 (2.0)	81.6 (2.0)	70.9 (3.7)
	DropNode [12]	75.2 (3.1)	32.4 (2.2)	78.5 (4.1)	70.6 (4.0)
	DropEdge [50]	78.3 (3.7)	32.7 (2.8)	81.8 (4.4)	71.5 (3.9)
	RandomWalkPE [8]	76.2 (3.5)	33.1 (2.5)	80.9 (2.7)	71.1 (3.8)
	LaplacianPE [9]	74.5 (2.9)	32.9 (2.4)	81.9 (2.7)	71.4 (3.6)
	GPT-Aug (Ours)	**79.2 (2.8)**	**34.8 (2.2)**	**82.3(4.5)**	**72.9 (3.9)**

Table 2: Performance [%] of GPT-Aug on the OGBN-arxiv and OGBN-products datasets.

GNN Archi.	Augmenter	Accuracy	
		OGBN-arxiv	**OGBN-products**
Graph SAGE	DropNode	58.42 (0.20)	54.22 (0.31)
	DropEdge	54.83 (0.19)	55.23 (0.32)
	RandomWalkPE	OOM	OOM
	LaplacianPE	OOM	OOM
	GraphGPT-std	62.58	N/A
	LLM*	73.56 (0.06)	74.40 (0.23)
	TAPE	76.72 (0.07)	81.37 (0.43)
	GPT-Aug (Ours)	**77.24 (0.17)**	**84.22 (0.27)**
GAT	DropNode	57.36 (0.25)	55.43 (0.34)
	DropEdge	58.26 (0.21)	53.36 (0.37)
	RandomWalkPE	OOM	OOM
	LaplacianPE	OOM	OOM
	GraphGPT-std	62.58	N/A
	LLM*	73.56 (0.06)	74.40 (0.23)
	TAPE	77.50 (0.12)	82.34 (0.36)
	GPT-Aug (Ours)	**77.98 (0.22)**	**84.00 (0.32)**
GCN	DropNode	58.57 (0.42)	56.94 (0.45)
	DropEdge	58.15 (0.43)	54.62 (0.47)
	RandomWalkPE	OOM	OOM
	LaplacianPE	OOM	OOM
	GraphGPT-std	62.58	N/A
	GraphGPT-stage2	75.11	N/A
	3-HiGCN	**76.41 (0.53)**	N/A
	LLM*	73.56 (0.06)	74.40 (0.23)
	TAPE	75.20 (0.03)	79.96 (0.41)
	GPT-Aug (Ours)	**76.05 (0.23)**	**82.86 (0.42)**

OOM: out-of-memory. LLM: Using zero-shot ChatGPT with the same prompts of TAPE as the approach, denoted as LLM.

Evaluation Metrics. We evaluate our method with area under the receiver operating curve (AUROC), area under the precision-recall curve (AUPR), accuracy, F1-scores, and Jaccard index, applied as relevant to each task. For robust validation of our results, we employ a five-fold cross-validation strategy in all major experiments. More detailed information on the datasets, tasks and their loss functions, and evaluation metrics is presented in the appendix.

5.2 Compared Methods

We compare our method to the following graph data augmentation methods to validate the empirical performance of GPT-Aug: LaplacianPE [9], RandomWalkPE [8], DropEdge [50], and DropNode [12]. For the EHR analysis benchmark, we also include additional competitors as follows: GraphCare (LLM-based) [30], GRU [43], Transformer [61], GRAM [4], StageNet [15], Concare [42], Adacare [41], Dr. Agent [14], and GRASP [85]. For drug recommendation, we also include additional competitors: MICRON [74], Safedrug [75], and MoleRec [78]. For the large-scale OGBN datasets, additionally, we have included more advanced LLM-based baselines (i.e., GraphGPT [57], LLM, TAPE [19] and HiGCN [26]). We reimplemented the baseline methods, where details of the implementations and descriptions of the baseline methods can be found in the appendix.

5.3 Quantitative Results

Results on Generic Graph Data. Table 1 presents the node classification results of our proposal compared to existing graph data augmentation methods. Table 2 presents the results on the large-scale OGBN-products and OGBN-arxiv datasets against both traditional and LLM-based competitors. We observe that our method achieves satisfactory performance on generic graph classification datasets, as well as large-scale datasets. Some of the traditional GDA methods which operate on whole graphs failed to generalize to large-scale datasets (i.e., encountered out-of-memory error). Our method obtains a 3% improvement on average over all comparable methods with all four GNN architectures (i.e., GCN [69], GAT [62], GIN [73], and GraphSAGE [17]). This shows evidence that leveraging context knowledge, such as dataset summary and class label information, with LLMs can augment graph data to its true data distribution. We also compare among the comparable methods with different GNN architectures. We observe that our method still performs satisfactorily when different GNN architectures are used, demonstrating the robustness of our method.

Results on EHR Data. Table 3 presents the results of different tasks on the MIMIC-III dataset (detailed results with more evaluation metrics are presented in the appendix). We observe that our proposed framework outperforms alternative methods, thereby validating the effectiveness of contextual LLM augmentation and sparsity-aware instruction prompting. In particular, our method

Table 3: Performance of drug recommendation, length of stay, mortality and readmission prediction on MIMIC-III [%]. Standard deviations are shown in brackets.

Model	Drug Recommendation		Length of Stay		Mortality		Readmission	
	AUROC	AUPR	AUROC	Acc.	AUROC	AUPR	AUROC	AUPR
GRU	96.38 (0.1)	64.75 (0.2)	80.32 (0.2)	42.14 (0.6)	61.09 (0.7)	9.65 (1.5)	65.58 (1.1)	68.57 (1.6)
Transformer	95.87 (0.0)	60.19 (0.1)	79.31 (0.8)	41.68 (0.7)	57.20 (1.3)	10.10 (0.9)	63.75 (0.5)	68.92 (0.1)
Deepr	96.09 (0.0)	62.48 (0.1)	78.02 (0.4)	39.31 (1.2)	60.80 (0.4)	13.20 (1.1)	66.50 (0.4)	68.80 (0.9)
GRAM	94.20 (0.0)	76.70 (0.1)	78.02 (0.4)	39.31 (1.2)	60.40 (0.9)	11.40 (0.7)	64.30 (0.4)	67.20 (0.8)
Concare	95.78 (0.1)	61.67 (0.3)	80.27 (0.3)	42.04 (0.6)	61.98 (1.8)	9.67 (1.5)	65.28 (1.1)	66.67 (1.9)
Dr. Agent	96.41 (0.1)	64.16 (0.5)	79.45 (0.6)	41.40 (0.5)	57.52 (0.4)	9.66 (0.8)	64.86 (2.6)	67.41 (1.0)
AdaCare	95.86 (0.0)	60.76 (0.0)	78.73 (0.4)	40.70 (0.8)	58.40 (1.4)	11.10 (0.4)	65.70 (0.3)	68.60 (0.6)
StageNet	96.05 (0.0)	62.43 (2.4)	77.94 (0.2)	40.70 (0.8)	61.50 (0.7)	12.40 (0.3)	66.70 (0.4)	69.30 (0.6)
GRASP	96.01 (0.1)	62.53 (0.3)	78.97 (0.4)	40.66 (0.3)	59.20 (1.4)	9.90 (1.1)	66.30 (0.6)	69.20 (0.4)
DropNode	97.60 (0.2)	81.41 (0.1)	81.10 (0.5)	41.81 (1.1)	58.06 (0.9)	9.46 (1.7)	64.48 (0.8)	67.75 (0.4)
DropEdge	95.61 (0.1)	72.32 (0.3)	78.41 (0.3)	39.98 (0.8)	57.85 (0.8)	10.34 (1.5)	62.11 (0.6)	67.46 (0.5)
RandomWalkPE	94.89 (0.1)	63.86 (0.2)	78.01 (0.4)	39.47 (0.9)	57.15 (1.2)	9.76 (0.9)	66.20 (0.7)	59.58 (0.6)
LaplacianPE	95.26 (0.2)	69.34 (0.3)	78.22 (0.3)	40.02 (0.9)	57.65 (1.1)	10.05 (1.2)	65.71 (0.6)	63.43 (0.8)
GraphCare	95.00 (0.0)	78.50 (0.2)	79.40 (0.3)	41.90 (0.2)	66.60 (1.1)	14.30 (0.8)	68.10 (0.6)	71.50 (0.7)
GPT-Aug (Ours)	**98.54 (0.2)**	**83.89 (0.1)**	**82.68 (0.2)**	**45.28 (1.0)**	**67.79 (0.6)**	**16.09 (1.6)**	**68.97 (0.4)**	**73.92 (0.4)**

outperforms the competitors by 7.4% (in accuracy) in length-of-stay prediction. Our method can even outperform the methods specifically designed for EHR analysis, including GraphCare [30], a similar method using LLM for personalized healthcare. We elaborate the key differences between our method and GraphCare in the appendix. When integrating the enriched context information (e.g., clinical discharge reports, radiology reports, and lab event reports) in real-world EHR datasets, the performance on clinical task prediction can be further improved.

5.4 Qualitative Results

Embedding Visualization. We visualize the node embeddings of each type of entity to evaluate the performance of feature representation learning. Figure 4 presents the TSNE plot of the embeddings generated by different methods. The task is readmission prediction on the MIMIC-III dataset with a GAT model. It is observed that the embeddings with GPT-Aug are grouped according to their node types, which validates that the embeddings learn the unique representation of each node type, while the embeddings without GPT-Aug are noisy and do not present a clear pattern by the node type.

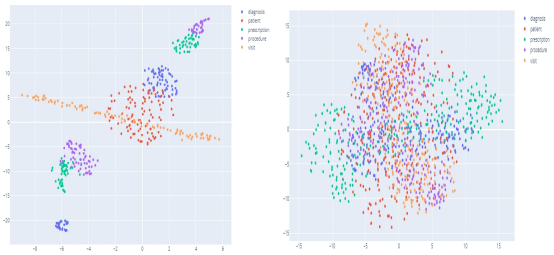

Figure 4: Visualization of the learned node embeddings w/ (left) and w/o (right) our graph data augmentation, respectively. We use MIMIC-III as the example and colour nodes differently by their entity types.

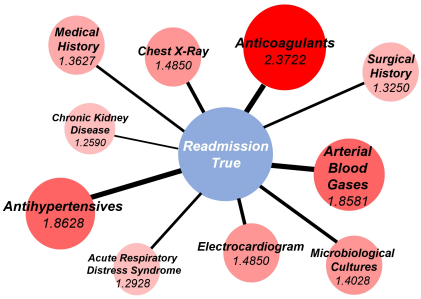

Figure 5: Visualization of the interpretability of GPT-Aug: a visit node (blue) and related concept nodes (red), with attention scores visiualized in size/shade of red nodes.

Network Interpretation. The incorporation of contextual learning enhances the capability of the model by enabling a nuanced understanding and interpretation of the graph data at a deeper level. We analyze the interpretability of our model by considering a specific visit node in the MIMIC-III dataset. As shown in Figure 5, the following are the top augmented corrections (i.e., with the highest attention scores) that exemplify the importance of specific clinical concepts influencing readmission prediction: Antihypertensives (2.3722), Anticoagulants (1.8628), and arterial blood gases (1.8581), where the computed attention scores are shown in brackets. It is observed that the augmentation process can impute context-related concepts so that GAT can select the most important ones. This provides interpretations for the predictive process. This is especially beneficial in the clinical decision context since the enriched open-world knowledge can inspire clinicians with the embedded concepts, and enhance the understanding of patients' behaviours and the potential reasons for certain diseases.

5.5 Ablated Analysis

The Effect of Augmented KGs. We study the effect of augmented KGs on downstream task performance (Table 4), including three scenarios: with KG, without KG, and with a biased (or wrong) KG augmented from another dataset (i.e. PPI). It is observed that the model performs worse than the baseline (i.e., w/o any augmentations) when the wrong context is applied, indicating a biased augmented graph. On the other hand, improved performance is observed when a context-driven KG is applied, thus validating the effectiveness of our method. A visualization of the effect of GPT-Aug on node embeddings can also be found in Figure 4.

The Effect of Dynamic Merging. We evaluate the contribution of the dynamic merging schema, as summarized in Table 5, where static merging means that the KG are merged into $\mathcal{G}_0$ offline before training. We observe that the performance improved on all generic graph datasets with dynamic merging, which validates the contributions of the schema.

Table 4: Performance w/ and w/o augmentation from KG, and w/ a biased KG from another dataset (i.e. PPI), repectively.

	w/o KG		w/ KG		w/ PPI KG	
Dataset	**Acc.**	**F1**	**Acc.**	**F1**	**Acc.**	**F1**
Cora	82.10	81.66	**83.60**	**83.64**	73.70	73.83
Actor	30.33	27.90	**32.21**	**28.91**	30.19	27.82
Citeseer	72.10	69.60	**73.10**	**72.46**	63.40	64.68

Table 5: Performance of node classification (using GAT) with and without dynamic merging, respectively.

	Cora		Actor		PPI		Citeseer	
Merging	**Acc.**	**F1**	**Acc.**	**F1**	**Acc.**	**F1**	**Acc.**	**F1**
Static	83.30	83.43	31.45	28.01	96.82	94.67	72.20	71.73
Dynamic	**83.60**	**83.64**	**32.21**	**28.91**	**98.28**	**97.20**	**73.10**	**72.46**

The Effect of Sparsity Control. We demonstrate how different levels of sparsity affect the performance of graph data augmentation. We control the level of sparsity using the number of edges per concept $|\mathcal{E}^{\text{conn}}|$ used for KG generation. Table 6 presents the results of this study. Given a fixed number of concepts, the performance improves when $|\mathcal{E}^{\text{conn}}|$ increases, demonstrating the effectiveness of graph merging. However, when $|\mathcal{E}^{\text{conn}}|$ is too large compared to the original graph size, the augmented graph would be biased from too many noisy connections, and hence the observed performance deteriorates.

The Influence of Different Granularity and Instruction Fine-tuning. We evaluate the influence of different granularity and instruction fine-tuning (IFT) on augmentation performance. From Table 7, it is observed that the performance is improved when an appropriate s is chosen, while adopting a multi-granularity $(s_0 + s_1)$ could potentially lead to over-sparsification. With KG concepts pruned by IFT, the performance is consistently improved on different granularity levels.

Table 6: Performance of node classification (using GAT) with different numbers of edges per concept generated by the KG.

	Cora		Actor		PPI		Citeseer	
$\lvert\mathcal{E}^{\text{conn}}\rvert$	**Acc.**	**F1**	**Acc.**	**F1**	**Acc.**	**F1**	**Acc.**	**F1**
0	81.3	80.7	30.3	28.0	91.6	97.1	72.1	69.6
3	83.6	83.6	32.2	28.9	96.4	97.2	73.1	72.5
30	79.3	79.4	31.0	28.8	97.5	97.2	68.5	68.3
100	75.4	75.5	30.9	28.4	98.3	97.2	66.2	66.2

Table 7: Performance of our framework on Cora node classification with different granularity levels s, and with or without IFT, respectively. We denote s_1 as the class type level, s_0 as the dataset level, and $s_1 + s_0$ as a multi-granularity scheme merging these two levels.

	$s = s_1$		$s = s_0$		$s = s_0 + s_1$	
IFT	**Acc.**	**F1**	**Acc.**	**F1**	**Acc.**	**F1**
w/o IFT	81.40	81.53	82.17	82.05	81.00	81.07
w/ IFT	83.20	83.26	**83.60**	**83.64**	83.15	83.25

6 Conclusion

We propose a novel framework for graph data augmentation, namely GPT-Aug, which leverages the open-world knowledge in LLMs to perform context-driven graph data augmentation. Our method directly operates on knowledge graphs constructed from LLM outputs and does not require access to model weights and features, which enables democratization to most of the closed-access LLMs. To tackle the sparsity induced by generated knowledge graphs, we design a granularity-aware prompting strategy to control the sparsity while maximizing the utility of domain knowledge. Experiments on generic graph datasets and a medical records dataset with an array of GNN architectures validate that our method can better augment the graph data than existing methods. Ablation analysis on key components and hyperparameters of our method validates the significance of our method and robustness to variations. Our method also has a wide range of potential application fields beyond medical record analysis such as molecular chemistry, recommendation, computational biology, social networks, and citation networks etc.

References

[1] Avrim Blum and Tom Mitchell. Combining labeled and unlabeled data with co-training. In *Proceedings of the eleventh annual conference on Computational learning theory*, pages 92–100, 1998.

[2] Xuheng Cai, Chao Huang, Lianghao Xia, and Xubin Ren. Lightgcl: Simple yet effective graph contrastive learning for recommendation. In *The Eleventh International Conference on Learning Representations*, 2023.

[3] Tsai Hor Chan, Chi Ho Wong, Jiajun Shen, and Guosheng Yin. Source-aware embedding training on heterogeneous information networks. *Data Intelligence*, pages 1–14, 2023.

[4] Edward Choi, Mohammad Taha Bahadori, Le Song, Walter F Stewart, and Jimeng Sun. Gram: graph-based attention model for healthcare representation learning. In *Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 787–795, 2017.

[5] Edward Choi, Cao Xiao, Walter Stewart, and Jimeng Sun. Mime: Multilevel medical embedding of electronic health records for predictive healthcare. *Advances in neural information processing systems*, 31, 2018.

[6] Guilherme Del Fiol, Clayton Curtis, James J Cimino, Andrew Iskander, Aditya SD Kalluri, Xia Jing, Nathan C Hulse, Jie Long, Casey L Overby, Connie Schardt, et al. Disseminating context-specific access to online knowledge resources within electronic health record systems. *Studies in health technology and informatics*, 192:672, 2013.

[7] Kaize Ding, Zhe Xu, Hanghang Tong, and Huan Liu. Data augmentation for deep graph learning: A survey. *ACM SIGKDD Explorations Newsletter*, 24(2):61–77, 2022.

[8] 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*, 2021.

[9] Vijay Prakash Dwivedi, Chaitanya K Joshi, Anh Tuan Luu, Thomas Laurent, Yoshua Bengio, and Xavier Bresson. Benchmarking graph neural networks. *Journal of Machine Learning Research*, 24(43):1–48, 2023.

[10] R Scott Evans. Electronic health records: then, now, and in the future. *Yearbook of medical informatics*, 25(S 01):S48–S61, 2016.

[11] Fuli Feng, Xiangnan He, Jie Tang, and Tat-Seng Chua. Graph adversarial training: Dynamically regularizing based on graph structure. *IEEE Transactions on Knowledge and Data Engineering*, 33(6):2493–2504, 2019.

[12] Wenzheng Feng, Jie Zhang, Yuxiao Dong, Yu Han, Huanbo Luan, Qian Xu, Qiang Yang, Evgeny Kharlamov, and Jie Tang. Graph random neural networks for semi-supervised learning on graphs. *Advances in neural information processing systems*, 33:22092–22103, 2020.

[13] Luca Franceschi, Mathias Niepert, Massimiliano Pontil, and Xiao He. Learning discrete structures for graph neural networks. In *International conference on machine learning*, pages 1972–1982. PMLR, 2019.

[14] Junyi Gao, Cao Xiao, Lucas M Glass, and Jimeng Sun. Dr. agent: Clinical predictive model via mimicked second opinions. *Journal of the American Medical Informatics Association*, 27(7): 1084–1091, 2020.

[15] Junyi Gao, Cao Xiao, Yasha Wang, Wen Tang, Lucas M Glass, and Jimeng Sun. Stagenet: Stage-aware neural networks for health risk prediction. In *Proceedings of The Web Conference 2020*, pages 530–540, 2020.

[16] Justin Gilmer, Samuel S Schoenholz, Patrick F Riley, Oriol Vinyals, and George E Dahl. Neural message passing for quantum chemistry. In *International conference on machine learning*, pages 1263–1272. PMLR, 2017.

[17] Will Hamilton, Zhitao Ying, and Jure Leskovec. Inductive representation learning on large graphs. *Advances in neural information processing systems*, 30, 2017.

[18] Will Hamilton, Zhitao Ying, and Jure Leskovec. Inductive representation learning on large graphs. *Advances in neural information processing systems*, 30, 2017.

[19] Xiaoxin He, Xavier Bresson, Thomas Laurent, Adam Perold, Yann LeCun, and Bryan Hooi. Harnessing explanations: Llm-to-lm interpreter for enhanced text-attributed graph representation learning. In *The Twelfth International Conference on Learning Representations*, 2023.

[20] William Hsu, Ricky K Taira, Suzie El-Saden, Hooshang Kangarloo, and Alex AT Bui. Context-based electronic health record: toward patient specific healthcare. *IEEE Transactions on information technology in biomedicine*, 16(2):228–234, 2012.

[21] Weihua Hu, Matthias Fey, Marinka Zitnik, Yuxiao Dong, Hongyu Ren, Bowen Liu, Michele Catasta, and Jure Leskovec. Open graph benchmark: Datasets for machine learning on graphs. *arXiv preprint arXiv:2005.00687*, 2020.

[22] Ziniu Hu, Changjun Fan, Ting Chen, Kai-Wei Chang, and Yizhou Sun. Pre-training graph neural networks for generic structural feature extraction. *arXiv preprint arXiv:1905.13728*, 2019.

[23] Ziniu Hu, Yuxiao Dong, Kuansan Wang, Kai-Wei Chang, and Yizhou Sun. Gpt-gnn: Generative pre-training of graph neural networks. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 1857–1867, 2020.

[24] Ziniu Hu, Yuxiao Dong, Kuansan Wang, and Yizhou Sun. Heterogeneous graph transformer. In *Proceedings of The Web Conference 2020*, pages 2704–2710, 2020.

[25] Tiancheng Huang, Ke Xu, and Donglin Wang. Da-hgt: Domain adaptive heterogeneous graph transformer. *arXiv preprint arXiv:2012.05688*, 2020.

[26] Yiming Huang, Yujie Zeng, Qiang Wu, and Linyuan Lü. Higher-order graph convolutional network with flower-petals laplacians on simplicial complexes. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 12653–12661, 2024.

[27] Maximilian Ilse, Jakub Tomczak, and Max Welling. Attention-based deep multiple instance learning. In *International conference on machine learning*, pages 2127–2136. PMLR, 2018.

[28] Guoliang Ji, Shizhu He, Liheng Xu, Kang Liu, and Jun Zhao. Knowledge graph embedding via dynamic mapping matrix. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 687–696, 2015.

[29] Dejun Jiang, Zhenxing Wu, Chang-Yu Hsieh, Guangyong Chen, Ben Liao, Zhe Wang, Chao Shen, Dongsheng Cao, Jian Wu, and Tingjun Hou. Could graph neural networks learn better molecular representation for drug discovery? a comparison study of descriptor-based and graph-based models. *Journal of cheminformatics*, 13(1):1–23, 2021.

[30] Pengcheng Jiang, Cao Xiao, Adam Cross, and Jimeng Sun. Graphcare: Enhancing healthcare predictions with open-world personalized knowledge graphs. *arXiv preprint arXiv:2305.12788*, 2023.

[31] Jin Jing, Wendong Ge, Shenda Hong, Marta Bento Fernandes, Zhen Lin, Chaoqi Yang, Sungtae An, Aaron F Struck, Aline Herlopian, Ioannis Karakis, et al. Development of expert-level classification of seizures and rhythmic and periodic patterns during eeg interpretation. *Neurology*, 100(17):e1750–e1762, 2023.

[32] Alistair EW Johnson, Tom J Pollard, Lu Shen, Li-wei H Lehman, Mengling Feng, Mohammad Ghassemi, Benjamin Moody, Peter Szolovits, Leo Anthony Celi, and Roger G Mark. Mimic-iii, a freely accessible critical care database. *Scientific data*, 3(1):1–9, 2016.

[33] John Jumper, Richard Evans, Alexander Pritzel, Tim Green, Michael Figurnov, Olaf Ronneberger, Kathryn Tunyasuvunakool, Russ Bates, Augustin Žídek, Anna Potapenko, et al. Highly accurate protein structure prediction with alphafold. *Nature*, 596(7873):583–589, 2021.

[34] Ryosuke Kojima, Shoichi Ishida, Masateru Ohta, Hiroaki Iwata, Teruki Honma, and Yasushi Okuno. kgcn: a graph-based deep learning framework for chemical structures. *Journal of Cheminformatics*, 12:1–10, 2020.

[35] Bin Li, Yin Li, and Kevin W Eliceiri. Dual-stream multiple instance learning network for whole slide image classification with self-supervised contrastive learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 14318–14328, 2021.

[36] Qimai Li, Zhichao Han, and Xiao-Ming Wu. Deeper insights into graph convolutional networks for semi-supervised learning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, 2018.

[37] Songtao Liu, Rex Ying, Hanze Dong, Lanqing Li, Tingyang Xu, Yu Rong, Peilin Zhao, Junzhou Huang, and Dinghao Wu. Local augmentation for graph neural networks. In *International Conference on Machine Learning*, pages 14054–14072. PMLR, 2022.

[38] Zheng Liu, Xiaohan Li, Hao Peng, Lifang He, and S Yu Philip. Heterogeneous similarity graph neural network on electronic health records. In *2020 IEEE International Conference on Big Data (Big Data)*, pages 1196–1205. IEEE, 2020.

[39] Fenglong Ma, Radha Chitta, Jing Zhou, Quanzeng You, Tong Sun, and Jing Gao. Dipole: Diagnosis prediction in healthcare via attention-based bidirectional recurrent neural networks. In *Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1903–1911, 2017.

[40] Fenglong Ma, Quanzeng You, Houping Xiao, Radha Chitta, Jing Zhou, and Jing Gao. Kame: Knowledge-based attention model for diagnosis prediction in healthcare. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, pages 743–752, 2018.

[41] Liantao Ma, Junyi Gao, Yasha Wang, Chaohe Zhang, Jiangtao Wang, Wenjie Ruan, Wen Tang, Xin Gao, and Xinyu Ma. Adacare: Explainable clinical health status representation learning via scale-adaptive feature extraction and recalibration. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 825–832, 2020.

[42] Liantao Ma, Chaohe Zhang, Yasha Wang, Wenjie Ruan, Jiangtao Wang, Wen Tang, Xinyu Ma, Xin Gao, and Junyi Gao. Concare: Personalized clinical feature embedding via capturing the healthcare context. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 833–840, 2020.

[43] Larry R Medsker and LC Jain. Recurrent neural networks. *Design and Applications*, 5:64–67, 2001.

[44] Linh Nguyen and Tsukasa Ishigaki. Domain-to-domain translation model for recommender system. *arXiv preprint arXiv:1812.06229*, 2018.

[45] OpenAI. Gpt-4 technical report, 2023.

[46] Hyeonjin Park, Seunghun Lee, Sihyeon Kim, Jinyoung Park, Jisu Jeong, Kyung-Min Kim, Jung-Woo Ha, and Hyunwoo J Kim. Metropolis-hastings data augmentation for graph neural networks. *Advances in Neural Information Processing Systems*, 34:19010–19020, 2021.

[47] Joonhyung Park, Hajin Shim, and Eunho Yang. Graph transplant: Node saliency-guided graph mixup with local structure preservation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 7966–7974, 2022.

[48] Jiezhong Qiu, Qibin Chen, Yuxiao Dong, Jing Zhang, Hongxia Yang, Ming Ding, Kuansan Wang, and Jie Tang. Gcc: Graph contrastive coding for graph neural network pre-training. In *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 1150–1160, 2020.

[49] Yu Rong, Wenbing Huang, Tingyang Xu, and Junzhou Huang. Dropedge: Towards deep graph convolutional networks on node classification. *arXiv preprint arXiv:1907.10903*, 2019.

[50] Yu Rong, Wenbing Huang, Tingyang Xu, and Junzhou Huang. Dropedge: Towards deep graph convolutional networks on node classification, 2020.

[51] Michael Schauperl and Rajiah Aldrin Denny. Ai-based protein structure prediction in drug discovery: impacts and challenges. *Journal of Chemical Information and Modeling*, 62(13): 3142–3156, 2022.

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

[53] Zhuchen Shao, Hao Bian, Yang Chen, Yifeng Wang, Jian Zhang, Xiangyang Ji, et al. Transmil: Transformer based correlated multiple instance learning for whole slide image classification. *Advances in neural information processing systems*, 34:2136–2147, 2021.

[54] Chuan Shi, Binbin Hu, Wayne Xin Zhao, and S Yu Philip. Heterogeneous information network embedding for recommendation. *IEEE Transactions on Knowledge and Data Engineering*, 31 (2):357–370, 2018.

[55] Martin Simonovsky and Nikos Komodakis. Graphvae: Towards generation of small graphs using variational autoencoders. In *Artificial Neural Networks and Machine Learning–ICANN 2018: 27th International Conference on Artificial Neural Networks, Rhodes, Greece, October 4-7, 2018, Proceedings, Part I 27*, pages 412–422. Springer, 2018.

[56] Susheel Suresh, Pan Li, Cong Hao, and Jennifer Neville. Adversarial graph augmentation to improve graph contrastive learning. *Advances in Neural Information Processing Systems*, 34: 15920–15933, 2021.

[57] Jiabin Tang, Yuhao Yang, Wei Wei, Lei Shi, Lixin Su, Suqi Cheng, Dawei Yin, and Chao Huang. Graphgpt: Graph instruction tuning for large language models, 2023.

[58] Jiabin Tang, Yuhao Yang, Wei Wei, Lei Shi, Lixin Su, Suqi Cheng, Dawei Yin, and Chao Huang. Graphgpt: Graph instruction tuning for large language models, 2024.

[59] Jake Topping, Francesco Di Giovanni, Benjamin Paul Chamberlain, Xiaowen Dong, and Michael M Bronstein. Understanding over-squashing and bottlenecks on graphs via curvature. *arXiv preprint arXiv:2111.14522*, 2021.

[60] Daniele Toti, Gabriele Macari, Enrico Barbierato, and Fabio Polticelli. Fgdb: a comprehensive graph database of ligand fragments from the protein data bank. *Database*, 2022:baac044, 2022.

[61] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[62] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. Graph attention networks. *arXiv preprint arXiv:1710.10903*, 2017.

[63] Petar Veličković, William Fedus, William L Hamilton, Pietro Liò, Yoshua Bengio, and R Devon Hjelm. Deep graph infomax. *arXiv preprint arXiv:1809.10341*, 2018.

[64] Vikas Verma, Alex Lamb, Christopher Beckham, Amir Najafi, Ioannis Mitliagkas, David Lopez-Paz, and Yoshua Bengio. Manifold mixup: Better representations by interpolating hidden states. In *International conference on machine learning*, pages 6438–6447. PMLR, 2019.

[65] Xiang Wang, Xiangnan He, Yixin Cao, Meng Liu, and Tat-Seng Chua. Kgat: Knowledge graph attention network for recommendation. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 950–958, 2019.

[66] Xiao Wang, Houye Ji, Chuan Shi, Bai Wang, Yanfang Ye, Peng Cui, and Philip S Yu. Heterogeneous graph attention network. In *The World Wide Web Conference*, pages 2022–2032, 2019.

[67] Yiwei Wang, Wei Wang, Yuxuan Liang, Yujun Cai, Juncheng Liu, and Bryan Hooi. Nodeaug: Semi-supervised node classification with data augmentation. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 207–217, 2020.

[68] Wei Wei, Xubin Ren, Jiabin Tang, Qinyong Wang, Lixin Su, Suqi Cheng, Junfeng Wang, Dawei Yin, and Chao Huang. Llmrec: Large language models with graph augmentation for recommendation. In *Proceedings of the 17th ACM International Conference on Web Search and Data Mining*, pages 806–815, 2024.

[69] Max Welling and Thomas N Kipf. Semi-supervised classification with graph convolutional networks. In *J. International Conference on Learning Representations (ICLR 2017)*, 2016.

[70] Peter West, Chandra Bhagavatula, Jack Hessel, Jena D Hwang, Liwei Jiang, Ronan Le Bras, Ximing Lu, Sean Welleck, and Yejin Choi. Symbolic knowledge distillation: from general language models to commonsense models. *arXiv preprint arXiv:2110.07178*, 2021.

[71] Tailin Wu, Hongyu Ren, Pan Li, and Jure Leskovec. Graph information bottleneck. *Advances in Neural Information Processing Systems*, 33:20437–20448, 2020.

[72] Qizhe Xie, Zihang Dai, Eduard Hovy, Thang Luong, and Quoc Le. Unsupervised data augmentation for consistency training. *Advances in neural information processing systems*, 33: 6256–6268, 2020.

[73] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How powerful are graph neural networks? In *International Conference on Learning Representations*, 2018.

[74] Chaoqi Yang, Cao Xiao, Lucas Glass, and Jimeng Sun. Change matters: Medication change prediction with recurrent residual networks. In *30th International Joint Conference on Artificial Intelligence, IJCAI 2021*, pages 3728–3734. International Joint Conferences on Artificial Intelligence, 2021.

[75] Chaoqi Yang, Cao Xiao, Fenglong Ma, Lucas Glass, and Jimeng Sun. Safedrug: Dual molecular graph encoders for recommending effective and safe drug combinations. *arXiv preprint arXiv:2105.02711*, 2021.

[76] Liang Yang, Zesheng Kang, Xiaochun Cao, Di Jin, Bo Yang, and Yuanfang Guo. Topology optimization based graph convolutional network. In *IJCAI*, pages 4054–4061, 2019.

[77] Longqi Yang, Liangliang Zhang, and Wenjing Yang. Graph adversarial self-supervised learning. *Advances in Neural Information Processing Systems*, 34:14887–14899, 2021.

[78] Nianzu Yang, Kaipeng Zeng, Qitian Wu, and Junchi Yan. Molerec: Combinatorial drug recommendation with substructure-aware molecular representation learning. In *Proceedings of the ACM Web Conference 2023*, pages 4075–4085, 2023.

[79] Shuwen Yang, Guojie Song, Yilun Jin, and Lun Du. Domain adaptive classification on heterogeneous information networks. In *IJCAI*, pages 1410–1416, 2020.

[80] David Yarowsky. Unsupervised word sense disambiguation rivaling supervised methods. In *33rd annual meeting of the association for computational linguistics*, pages 189–196, 1995.

[81] Yuning You, Tianlong Chen, Yongduo Sui, Ting Chen, Zhangyang Wang, and Yang Shen. Graph contrastive learning with augmentations. *Advances in neural information processing systems*, 33:5812–5823, 2020.

[82] Yuning You, Tianlong Chen, Zhangyang Wang, and Yang Shen. When does self-supervision help graph convolutional networks? In *international conference on machine learning*, pages 10871–10880. PMLR, 2020.

[83] Han Yue, Chunhui Zhang, Chuxu Zhang, and Hongfu Liu. Label-invariant augmentation for semi-supervised graph classification. *Advances in Neural Information Processing Systems*, 35: 29350–29361, 2022.

[84] Seongjun Yun, Minbyul Jeong, Raehyun Kim, Jaewoo Kang, and Hyunwoo J Kim. Graph transformer networks. *Advances in neural information processing systems*, 32, 2019.

[85] Chaohe Zhang, Xin Gao, Liantao Ma, Yasha Wang, Jiangtao Wang, and Wen Tang. Grasp: generic framework for health status representation learning based on incorporating knowledge from similar patients. In *Proceedings of the AAAI conference on artificial intelligence*, volume 35, pages 715–723, 2021.

[86] Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. *arXiv preprint arXiv:1710.09412*, 2017.

[87] Xikun Zhang, Antoine Bosselut, Michihiro Yasunaga, Hongyu Ren, Percy Liang, Christopher D Manning, and Jure Leskovec. Greaselm: Graph reasoning enhanced language models for question answering. *arXiv preprint arXiv:2201.08860*, 2022.

[88] Tianxiang Zhao, Xiang Zhang, and Suhang Wang. Graphsmote: Imbalanced node classification on graphs with graph neural networks. In *Proceedings of the 14th ACM international conference on web search and data mining*, pages 833–841, 2021.

[89] Weiqi Zhao, Dian Tang, Xin Chen, Dawei Lv, Daoli Ou, Biao Li, Peng Jiang, and Kun Gai. Disentangled causal embedding with contrastive learning for recommender system. *arXiv preprint arXiv:2302.03248*, 2023.

[90] Yue Zhao, Zhi Qiao, Cao Xiao, Lucas Glass, and Jimeng Sun. Pyhealth: A python library for health predictive models. *arXiv preprint arXiv:2101.04209*, 2021.

[91] Cheng Zheng, Bo Zong, Wei Cheng, Dongjin Song, Jingchao Ni, Wenchao Yu, Haifeng Chen, and Wei Wang. Robust graph representation learning via neural sparsification. In *International Conference on Machine Learning*, pages 11458–11468. PMLR, 2020.

A Broader Impact and Limitations

Border Impacts. The GPT-Aug framework offers significant extensibility across diverse applications due to its versatile core methodologies, notably in Computational Biology, Computer Vision, and sequence data.

In computational biology, it enhances drug discovery and protein structure prediction by generating biologically plausible augmentations for protein graphs, leveraging domain knowledge about amino acid sequences and protein interactions [29, 60, 33]. This addresses the challenge of relying on vast, high-quality datasets [51].

For histopathology analysis, GPT-Aug collaborates well with typical multiple-instance learning methods and can integrate the biological and clinical information described in the clinical reports with the LLM components [35, 27, 53].

In recommendation systems, GPT-Aug can facilitate graph-based methods as well as collaborative filtering (CF) methods which mitigate the bias inherent from the noisy user-iterm interaction. The use of LLM can effectively mine the contextual information from item descriptions to provide more accurate recommendations to users [34, 89, 28, 54, 78, 44].

Limitations. Since our method operates on latent knowledge graphs, it is difficult to generate KGs and perform instruction fine-tuning online in general scenarios due to the limitations of computational resources. However, under sufficient computational power, the LLM can be updated simultaneously during GNN training via instruction fine-tuning (e.g., after every backpropagation step) such that the generated KGs can be closer to the domain context, which is a promising extension in future works.

B Additional Dataset Information

We present additional information on the datasets used in the experiments. Tables 10 and 8 present the summary information on generic graph datasets and the MIMIC-III dataset.

- Cora Dataset: The Cora dataset includes 2,708 scientific publications across seven classes, forming a citation network with 5,429 links. Each publication is represented by a binary word vector indicating the presence or absence of 1,433 unique words.
- Protein-Protein Interaction (PPI) Dataset: The PPI dataset consists of graphs representing interactions between proteins in various human tissues. Nodes reflect gene expressions, and edges denote protein interactions.

Table 8: Summary of the generic graph benchmark datasets.

	PPI	Actor	Cora	Citeseer
Task	Inductive	Inductive	Transductive	Transductive
Nodes	56,944	7,600	2,708	3,327
Edges	818,716	33,391	5,429	4,732
Features	50	932	1,433	3,703
Classes	121	5	7	6
Training Nodes	44,906	3,648	140	120
Validation Nodes	6,514	2,432	500	500
Testing Nodes	5,524	1,520	1,000	1,000

Table 9: Summary of the OGBN datasets

Datasets	**Scale**	**# Node**	**# Edges**	**# Class**
OGBN-products	Large	2,449,029	61,859,140	47
OGBN-arxiv	Large	169,343	1,166,243	40

Table 10: Summary of the MIMIC-III dataset.

Node Type	Count	Avg. # Visits Per Entity	Task	# Obs.
Patients	46,520	—	Mortality	9,718
Visits	58,976	—	Readmission	9,718
Diagnoses	6,984	11.04	LOS	44,407
Prescriptions	4,204	70.40	Drug Recomm.	14,142
Procedures	2,032	1.55		

C Evaluation Metrics

We provide detailed definitions of the evaluation metrics. For multi-class and multi-label classification tasks, the weighted averaging method is adopted for some metrics.

- Classification metrics:
  - Accuracy: the fraction of correct predictions to the total number of ground truth labels.
  - F-1 score: The F-1 score for each class is defined as

$$\text{F-1 score} = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}$$

    where 'recall' is the fraction of correct predictions to the total number of ground truths in each class and precision is the fraction of correct predictions to the total number of predictions in each class.
  - AUC: the area under the receiver operating curve (ROC) which is the plot of the true positive rate (TPR/Recall) against the false positive rate (FPR).
  - AUPR: the area under the precision-recall curve.
  - Jaccard index: measures the similarity between the true binary labels and the predicted binary labels by the ratio of the size of the intersection of the true positive labels and the predicted positive labels to the size of the union of the true positive (TP) labels and the predicted positive labels including TP, false postive (FP) and false negative (FN),

$$\text{Jaccard} = \frac{\text{TP}}{\text{TP} + \text{FP} + \text{FN}}.$$

D Additional Information of Related Methods

We provide supplementary information on the baseline methods and related works employed in our study. All baseline models were trained for 50 epochs with the option of early stopping. We choose the model at the epoch where it reaches the best performance in terms of AUROC. The following is a summary of the EHR analysis included and compared:

- Dipole [39]: adopts bidirectional recurrent neural networks and attention mechanism to learn medical code representation and provide predictions.
- KAME [40]: a generalized version of GRAM [4] adding attention mechanism to graph representation learning to provide interpretative diagnoses.
- SparcNet [31]: an algorithm that can classify seizures and other seizure-like events with expert-level reliability by analyzing electroencephalograms (EEGs).
- GRU [43]: a vanilla Gated recurrent unit model for visits sequence modelling.
- Transformer [61]: it leverages the idea of self-attention, which allows the model to selectively focus on different parts of the input sequence when generating an output.
- GRAM [4]: the first work models EHR with a knowledge graph and uses recurrent neural networks to learn the medical code representations and predict the future visit information.
- StageNet [15]: using a stage-aware LSTM to conduct clinical predictive tasks while learning patient disease progression stage change in an unsupervised manner.

- Concare [42]: it considers personal characteristics during clinical visits and uses cross-head decorrelation to capture inter-dependencies among dynamic features and static baseline information for predicting patients' clinical outcomes given EHRs.
- Adacare [41]: it can capture the long and short-term variations of biomarkers as clinical features, model the correlation between clinical features to enhance the ones which strongly indicate health status, and provide qualitative interpretability while maintaining a state-of-the-art performance in terms of prediction accuracy.
- Dr. Agent [14]: mimics clinical second opinions using two reinforcement learning agents and learn patient embeddings with the agents.
- GRASP [85]: applies GNN to cluster patients using their latent features and identifies similar patients based on latent clusters.
- GraphCare [30]: integrates external open-world knowledge graphs (KGs) into the patient-specific KGs with large language models.

For drug recommendation, we also included the following additional competitors.

- MICRON [74]: a sophisticated personalized drug recommendation system that incorporates patient-specific genetic and molecular information. It utilizes multi-omics data to identify optimal drug combinations, especially for complex diseases, thereby increasing treatment specificity based on patients' molecular characteristics.
- Safedrug [75]: a drug-drug-interaction-controllable (DDI-controllable) drug recommendation model that leverages drugs' molecule structures and models DDIs explicitly. It uses a global message passing neural network (MPNN) module and a local bipartite learning module to fully encode the connectivity and functionality of drug molecules.
- MoleRec [78]: a novel molecular substructure-aware encoding method that employs a hierarchical architecture to model inter-substructure interactions and the impact of individual substructures on a patient's health condition.

E Implementation Details

Temperature annealing. We are aware of the vanishing classification loss in practice. Therefore, we alleviate this issue by annealing the temperature over the training epochs with the schedule $\tau = \max(0.5, \exp(rp))$, where p is the training epoch and $r = 0.01$.

Downsampling for mortality task. We are aware that the samples in the mortality prediction task are heavily imbalanced (i.e., most of the samples are not dead). We therefore perform downsampling during training to balance the samples.

Configurations. The proposed framework is implemented in Python with the *Pytorch* library on a server equipped with four NVIDIA GeForce RTX 3090 GPUs. We use the *dgl* library to perform graph-related operations, and *pyhealth* [90] to benchmark SOTA methods and perform EHR-related operations. We integrate the gpt-4-0125-preview API [45], serving as a frozen large language model. The dropout ratio of each dropout layer is set as 0.1. All models are trained with 1000 epochs with early stopping. We use the Adam optimizer to optimize the model with a learning rate of 5×10^{-5} and a weight decay of 1×10^{-5}.

F Compute Amount Analysis

Time Complexity Analysis. Since we generate the KGs offline using the OpenAI API of gpt-4-0125-preview (our method works under a black-box setting), this process only needs to be performed once for each dataset. The additional complexity arises from the dynamic merging process, which needs to be repeated at each optimization step. However, the time complexity of this step is trivial compared to the forward passing of GNNs. Therefore, it only increases the overall time complexity on a minor level.

Table 11: Analysis of time complexity of training time on the ogbn-arxiv dataset.

Method	CGA (Ours) on GAT	TAPE	GraphGPT-stage-2	GraphGPT-stage-1
Training Time (min)	89	192	224	1325

Table 11 above shows the quantitative analysis of the training time complexity on the ogbn-arxiv dataset.

Efficiency through Single Query and Reuse. our prompting paradigm avoids manual prompt customization for adaptations to different datasets, thereby reducing human labor costs. Our method necessitates only a single query to the LLM, with KGs and significant concept nodes stored for subsequent reuse. Our query process can be efficiently completed in 37.6 seconds in average for the large-scale ogbn-arxiv dataset. This approach not only enhances efficiency but also reduces the number of API calls, thereby saving the cost of commercial LLMs. Additionally, we have provided the responses from the LLMs gained in our experiments for the public use.

G Additional Details on Healthcare Tasks

We include detailed descriptions of the healthcare tasks performed on the EHR datasets.

Mortality Prediction Mortality prediction aims to predict the mortality label of the subsequent visit for each sample. Formally, the function $f : (x_1, x_2, \ldots, x_{t-1}) \rightarrow y[x_t]$ is defined, where $y[x_t] \in \{0, 1\}$ is a binary label indicating the patient's survival status recorded in visit x_t.

Readmission Prediction The task of readmission prediction focuses on whether a patient will be readmitted within δ days. Formally, $f : (x_1, x_2, \ldots, x_{t-1}) \rightarrow y[\delta(x_t) - \delta(x_{t-1})]$, where $\delta(x_t)$ represents the encounter time of visit x_t, so that $y[\delta(x_t) - \delta(x_{t-1})]$ is 1 if $\delta(x_t) - \delta(x_{t-1}) \leq \delta$ and 0 otherwise. For our EHR study, we set $\delta = 15$ days.

Length-Of-Stay Prediction Length-Of-Stay (LOS) prediction predicts the length of ICU stay for each visit. Formally, $f : (x_1, x_2, \ldots, x_t) \rightarrow y[x_t]$, where $y[x_t] \in \mathbb{R}^{1\times C}$ is a one-hot vector indicating its class among C classes.

Drug Recommendation This task predicts medication labels for each visit. Formally, $f : (x_1, x_2, \ldots, x_t) \rightarrow y[x_t]$, where $y[x_t] \in \mathbb{R}^{1\times|d|}$ is a multi-hot vector where $|d|$ denotes the number of all drug types.

H Additional Experiment Results

We present additional experiment results with more metrics for comparisons. Table 12 and 13 present the performance on length of stay prediction and drug recommendations with more evaluation metrics.

I Additional Details on GNN Architectures

Graph Neural Network (GNN). A GNN, denoted as $\mathcal{M}$, operates on $\mathcal{G}$ and takes its feature space $\mathcal{X}$ to perform prediction by message passing. The message-passing mechanism of GNNs can be presented as

$$h_v^{(l+1)} = \text{UPDATE}^{(l)}\left(h_v^{(l)}, \text{AGG}^{(l)}\{h_u^{(l)} : u \in \mathcal{N}(v)\}\right),$$

where $h_v^{(l)}$ is the feature vector of node v at the l-th layer, $\mathcal{N}(v)$ denotes the set of neighbors of v, and UPDATE(•) and AGG(•) are functions for the update and aggregation steps respectively.

- Graph Convolutional Network (GCN): operates on graphs using a spectral approach for convolution, aggregating neighbor node information to update node features.

Table 12: Performance (in %) of our method on the drug recommendation task on the MIMIC-III dataset. Standard deviations are shown in brackets.

	Drug Recommendation			
Model	**AUPRC**	**AUROC**	**F1-score**	**Jaccard**
GRU	77.0(0.1)	94.4(0.0)	62.3(0.3)	47.8(0.3)
Transformer	76.1(0.1)	94.2(0.0)	62.1(0.4)	47.1(0.4)
RETAIN	77.1(0.1)	94.4(0.0)	63.7(0.2)	48.8(0.2)
GRAM	76.7(0.1)	94.2(0.1)	62.9(0.3)	47.9(0.3)
Deepr	74.3(0.1)	93.7(0.0)	60.3(0.4)	44.7(0.3)
StageNet	74.4(0.1)	93.0(0.1)	61.4(0.3)	45.8(0.4)
SafeDrug	68.1(0.3)	91.0(0.1)	46.7(0.4)	31.7(0.3)
MICRON	77.4(0.0)	94.6(0.1)	63.2(0.4)	48.3(0.4)
GAMENet	76.4(0.0)	94.2(0.1)	62.1(0.1)	47.2(0.4)
MoleRec	69.8(0.1)	92.0(0.1)	58.1(0.1)	43.1(0.3)
GraphCare	78.5(0.2)	94.8(0.1)	64.4(0.3)	49.2(0.4)
Ours	**81.8(0.1)**	**97.1 (0.2)**	**66.1 (0.2)**	**49.4 (0.8)**

Table 13: Performance (in %) of our method on prediction of the length of stay on the MIMIC-III datasets. Standard Deviations are shown in brackets.

	Prediction of Length of Stay MIMIC-III		
Model	**Accuracy**	**AUROC**	**F1**
GRU	42.14 (0.6)	80.23 (0.2)	27.36 (0.7)
Transformer	41.68 (0.7)	79.30 (0.8)	27.52 (0.8)
Deepr	39.31 (1.2)	78.02 (0.4)	25.09 (1.3)
GRAM	40.00 (0.0)	78.00 (0.0)	34.00 (0.0)
Concare	42.04 (0.6)	80.27 (0.3)	25.44 (1.3)
Dr. Agent	41.40 (0.5)	79.45 (0.6)	27.55 (0.3)
AdaCare	40.7 (0.8)	78.73 (0.4)	26.26 (0.8)
StageNet	40.18 (0.7)	77.94 (0.2)	26.63 (1.2)
GRASP	40.66 (0.3)	78.97 (0.4)	22.80 (0.8)
GraphCare	43.20 (0.4)	81.40 (0.3)	37.50 (0.2)
GPT-Aug	46.28 (1.0)	85.68 (0.1)	38.67 (0.6)

- Graph Attention Network (GAT): utilizes attention mechanisms to weigh neighbor contributions, employing a self-attention mechanism for attention weight calculation and neighbor feature aggregation.
- Graph Isomorphism Network (GIN): aggregates neighbor features using a learnable function, maintaining invariance to neighbor ordering in both directed and undirected graphs.
- GraphSAGE [17]: a pioneer sampling and aggregating algorithm for inductive graph representation learning.

J Additional Ablation Studies

The Influence of Number of GNN Layers. We evaluate the performance of our method with different numbers of GNN layers, as summarized in Table 14. We observe that in general a better performance is obtained when the number of layers is small. The performance slightly deteriorates as the number of layers increases more than two layers, indicating the potential over-smoothing problem. Other experiments on relatively fine-grained hyperparameters, such as the dropout rate, number of hidden dimensions, and number of attention heads for GAT, are presented in the appendix.

Dropout Ratios. Since graph learning is difficult to optimize and easy to lead to overfitting, we adopt dropout as the default regularizer for all benchmark methods. We further study the effects of

Table 14: Performance in terms of accuracy (%) of our framework on node classification with different numbers of layers L, using GCN and GAT.

	GCN			GAT		
L	**Cora**	**Actor**	**Citeseer**	**Cora**	**Actor**	**Citeseer**
1	81.40	31.91	71.77	83.30	32.21	72.70
2	81.50	32.41	73.10	83.60	29.21	73.10
3	82.90	31.45	70.45	82.10	28.49	72.10
4	80.50	30.54	70.04	81.70	28.20	71.70

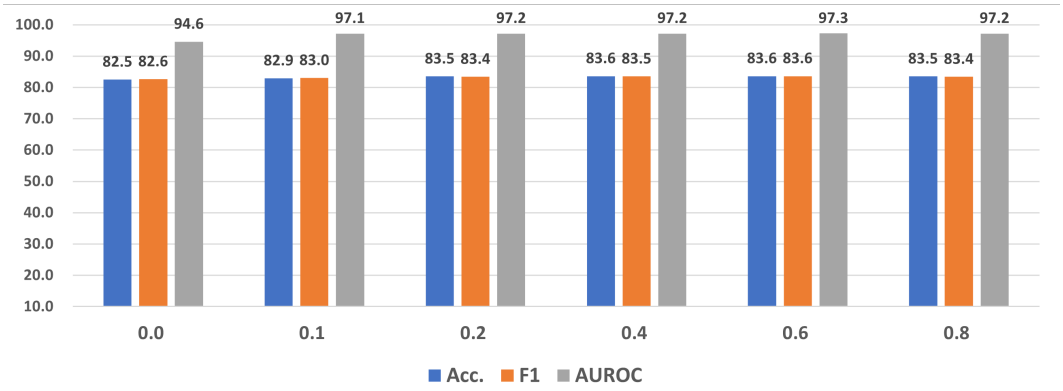

Figure 6: Performance of our method on Cora node classifications with respect to different dropout ratios, with GAT as the GNN architecture.

different dropout rates, Figure 6 presents the results. We observe that our method is in general robust to changes in dropout rates while being optimized when the dropout rate is 0.6. However, a large dropout rate would lead to over-sparsification of neural network weights and important features being dropped, hindering the predictive performance.

Number of hidden dimensions. We benchmark our method with respect to different hidden dimensions. Figure 7 presents the results of this study. We observe our method is overall robust to different numbers of hidden dimensions. In general, a larger number of hidden dimensions leads to better classification performance.

Number of attention heads. We benchmark the performance of our method with respect to different attention heads, as summarized in Figure 8. We observe that the performance is overall improving with the number of heads increases, while a larger number of heads (e.g., 32) would lead to a heavier memory burden under the current hardware settings.

K Details of Prompting Designs

We present additional examples of our prompting designs, including the ones used in KG generations for generic graph datasets (Cora, Citeseer, PPI, Actor).

Prompt Template for Cora: Specifically, for the Cora dataset of scientific publications, we follow the structure mentioned before and thus design an effective prompt as follows.

```
Given a prompt (a node from Cora dataset with Neural_Networks as label
), generate an extensive array of associated connections based on your
 domain knowledge, which shuold be helpful for the "node
classification task." [Replaced with the description of the specific
downstream task(s).]
Note that the updates should based on the provided node label and
important indices and backed up by your knowledge, being reasonable
and useful. It should not be unnecessary or nonsense.
```

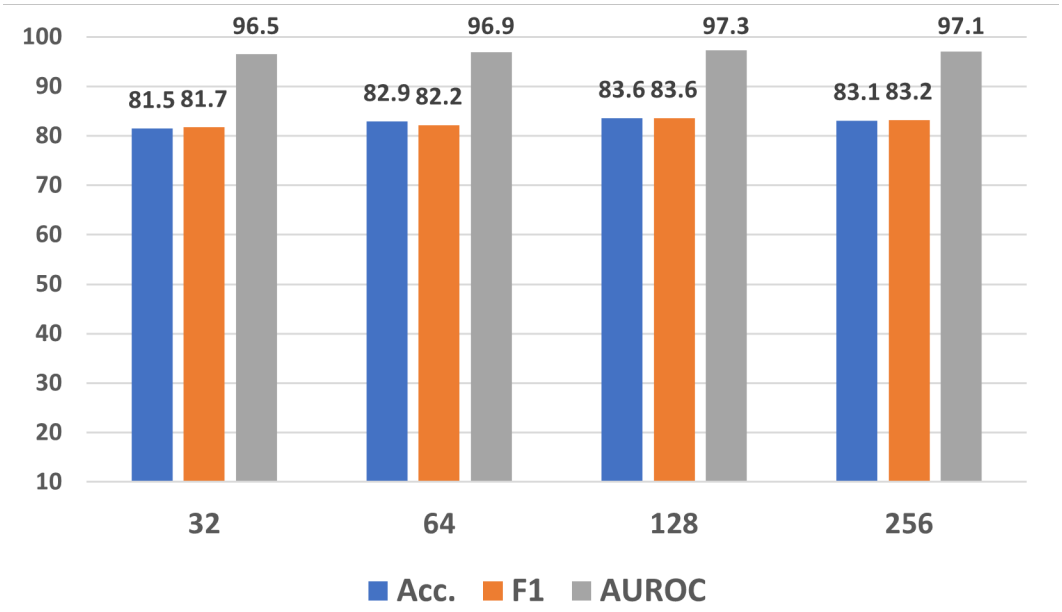

Figure 7: Performance of our method on Cora node classifications with respect to different numbers of hidden dimensions, with GAT as the GNN architecture.

```
"The Cora dataset consists of 2708 scientific publications classified
into one of seven classes.The citation network consists of 5429 links.
 Each publication in the dataset is described by a 0/1-valued word
vector indicating the absence/presence of the corresponding word from
the dictionary. The dictionary consists of 1433 unique words." [
Replaced with the description of the specific dataset.]
Format each association as [ENTITY 1, RELATIONSHIP, ENTITY 2],
ensuring the sequence reflects the direction of the relationship. Both
 ENTITY 1 and ENTITY 2 are to be nouns. Elements within [ENTITY 1,
RELATIONSHIP, ENTITY 2] must be definitive and succinct.
{example}
prompt: f"Node {term} in Cora with label {mode} and important indices
{non_zero_indices}"
updates:
```

Prompt Example for Citeseer (a scientific publications citation dataset) on the Granularity Level s_0 with IFT: Specifically, for the Citeseer dataset of citation networks of scientific publications, we provide a concrete example of prompts following the prompting strategy discussed before.

```
Given the The Citeseer dataset,which consists of 3312 scientific
publications classified into one of six classes. The citation network
consists of 4732 links. Each publication in the dataset is described
by a 0/1-valued word vector indicating the absence/presence of the
corresponding word from the dictionary. The dictionary consists of
3703 unique words.
Please generate 100 important concepts that related to the whole
dataset and all subtypes(Agents, Artificial Intelligence, Database,
Human Computer Interaction, Machine Learning and Information Retrieval
.), which are crucial for downstream task like node classification.
Each concept should be a single term or a short phrase that
encapsulates an important idea, technique, or subject within these
domains.
Make sure the concepts are relevant to the whole dataset and could
improve the downstream tasks' performance.
```

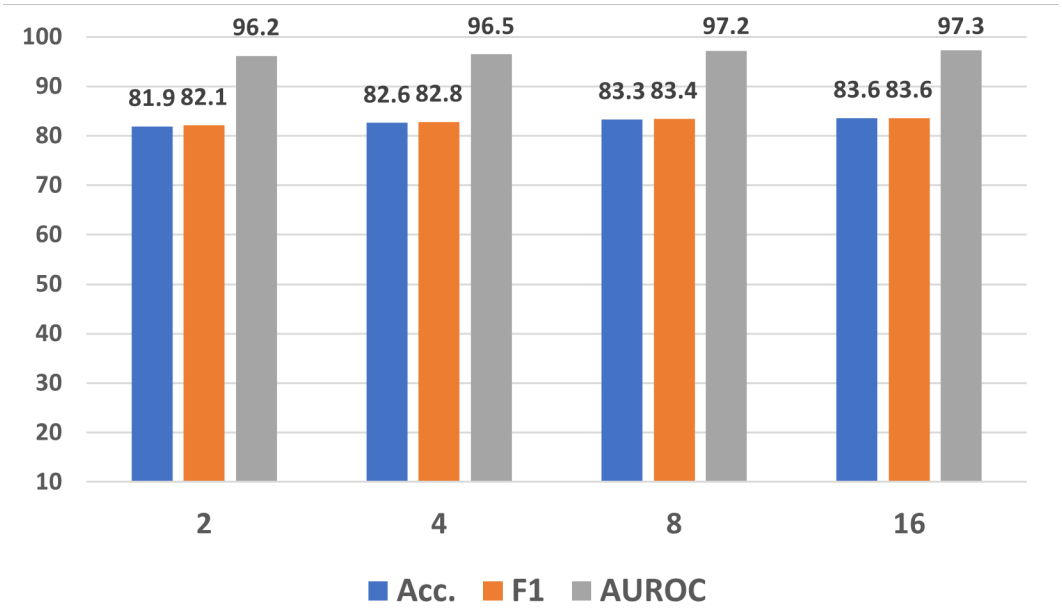

Figure 8: Performance of our method on Cora node classifications with respect to different heads of GAT.

The following prompt are for the instruction fine-tuning process for concept pruning:

```
Among the prior 100 concepts, select 30 most important concepts from
the list. The importance of a concept is based on your knowledge and
inference on how it will help improve the node classification task on
Citeseer dataset. If you think a concept is important, please keep it.
 Otherwise, please remove it.
```

The following prompt is for the KG generation given the pruned concepts output from the LLM using the previous prompts.

```
You are now a professional scientific publication researcher. Given
the list of 30 important concepts augmented with the Citeseer datset,
extrapolate 100 relationships of it and provide a list of triples.
{context_descriptions}
The relationships should be helpful for downstream node classification
 task.
Each update should be exactly in format of [ENTITY 1, RELATIONSHIP,
ENTITY 2]. The relationship is directed, so the order matters.
Both ENTITY 1 and ENTITY 2 should be selected from the list of 30
important concepts.
The selection of entities and the construction of relationship should
be supported by your professional domain knowledge, and do make sense.
 It should not be nonsense or randomly constructed.
Consideting the complexity of the task, you can iterate each one of
the concept in the list and consider them as 'ENTITY1', and provide
the name of 'ENTITY2', which are also from the list, to fulfill the
requirement stated before.
```

where the “context_descriptions" in {} is prompted into the LLM in the previous prompt.

Prompt Example for PPI (a protein-protein interaction dataset): Specifically, for the PPI dataset of protein-protein interaction in the Computational Biology field, we provide a concrete example of prompts following the prompting strategy discussed before.

```
Given the The PPI dataset,which is protein-protein interaction network
 that contains physical interactions between proteins that are
experimentally documented in humans, such as metabolic enzyme-coupled
interactions and signaling interactions. Nodes represent human
proteins and edges represent physical interaction between proteins in
a human cell. It contains 24 graphs. The average number of nodes per
graph is 2372. Each node has 50 features and 121 labels. 20 graphs for
 training, 2 for validation and 2 for testing.
Please generate 200 important concepts that related to the whole
Protein-protein Interactions dataset and all its subtypes, which are
crucial for downstream task like node classification.
Each concept should be a single term or a short phrase that
encapsulates an important idea, technique, or subject within these
domains.
Make sure the concepts are relevant to the whole dataset and could
improve the downstream tasks' performance.
```

The following prompt are for the instruction fine-tuning process for concept pruning:

```
Among the prior 200 concepts, select the 100 most important concepts
from the list. The importance of a concept is based on your knowledge
and inference of how it will help improve the node classification task
 on the Actor dataset. If you think a concept is important, please
keep it. Otherwise, please remove it.
```

The following prompt is for the KG generation given the pruned concepts output from the LLM using the previous prompts.

```
You are now a professional computational biology researcher. Given the
 list of 100 important concepts augmented with the Protein-Protein
Interaction(PPI) datset, extrapolate 300 relationships between the
concept nodes and provide a list of triples.
{context_descriptions}
The relationships should be helpful for downstream node classification
 task on PPI dataset.
Each update should be exactly in format of [ENTITY 1, RELATIONSHIP,
ENTITY 2]. The relationship is directed, so the order matters.
Both ENTITY 1 and ENTITY 2 should be selected from the list of 200
important concepts in the txt file uploaded.
The selection of entities and the construction of relationship should
be supported by your professional domain knowledge, and do make sense.
 It should not be nonsense or randomly constructed.
```

where the "context_descriptions" in {} is prompted into the LLM in the previous prompt.

Prompt Example for Actor (a dataset of film professionals relationships) with IFT: Specifically, for the Actor dataset of connections among film professionals, we provide a concrete example of prompts following the prompting strategy discussed before.

```
Given the Actor dataset, which is crawled from Wikipedia under the
category of "English-language films". In total, there are Nodes: 7600,
 Edges: 33391, Number of Classes: 5. The relationship types include:
film-director, film-actor, film-writer, and other relationships
between actors, directors, and writers. The first three types of
relationships are extracted from the "infobox" on the films' Wiki
pages. All the other types of people relationships are created as
follows: if one person (including actors, directors, and writers)
appears on another people's page, then a directed relationship is
created between them.
Please generate 100 important concepts of 'film/director/actor/writer'
 that are related to the whole dataset and all 5 subtypes, which are
crucial for downstream tasks like node classification.
```

```
Each concept should be a single term or a short phrase that
encapsulates an important idea, technique, or subject within these
domains.
Make sure the concepts are relevant to the whole dataset and could
improve the downstream tasks' performance.
```

The following prompt are for the instruction fine-tuning process for concept pruning:

```
Among the prior 100 concepts, select the 30 most important concepts
from the list. The importance of a concept is based on your knowledge
and inference of how it will help improve the node classification task
 on the Actor dataset. If you think a concept is important, please
keep it. Otherwise, please remove it.
```

The following prompt is for the KG generation given the pruned concepts output from the LLM using the previous prompts.

```
You are now a professional film industry researcher. Given the list of
 30 important concepts augmented with the Actor dataset, extrapolate
100 relationships between the concept nodes and provide a list of
triples. {context_descriptions}
The relationships should be helpful for the downstream node
classification task. Each update should be exactly in the format of [
ENTITY 1, RELATIONSHIP, ENTITY 2]. The relationship is directed, so
the order matters.
Both ENTITY 1 and ENTITY 2 should be selected from the list of 30
important concepts in the txt file uploaded.
The selection of entities and the construction of relationships should
 be supported by your professional domain knowledge, and do make sense
. It should not be nonsense or randomly constructed.
```

where the "context_descriptions" in {} is prompted into the LLM in the previous prompt.

L Differences Between Our Work and GraphCare

We highlight a closely related work to ours — GraphCare [30]. Although both works borrow the knowledge from LLMs to graph learning domain, GraphCare can be viewed as a special case of our work where patients are modelled as a personalized graph. GraphCare distinctively requests clinical reports and enriched contextual information, making it difficult to generalize to scenarios when such information is scarce or not available. Our method focuses on graph data augmentation in general, where our designed prompting strategy is applicable to almost all graph representation learning scenarios. Our method also involved sparsity control designs such as granularity-aware prompting and IFT concept pruning, which are absent in the work of GraphCare. Empirical comparisons on the EHR dataset also validate that our proposed graph data augmentation method outperforms their design.

M Extended Summary of Graph Data Augmentation Works

Modality-oriented Augmentation: GDA includes structure/feature/label-oriented techniques based on the different types of information modalities present in graphs. Structure-oriented GDA adopted edge perturbation [63], graph diffusion [59, 91, 48, 46], graph sampling [18, 48], node dropping and insertion [16], etc. Feature-oriented techniques focus on feature corruption [11, 63, 77], masking [81], rewriting [67, 76], and mixing [64]. Label-oriented GDA directly enriches the expensive labeled data mainly by two ways: Pseudo-labeling and Label Mixing [86]. While these methods may be helpful for certain tasks, their granularity levels are inflexible and semantic awareness is lost.

Graph Data Augmentation for Low-resource Graph Learning: Graph self-supervised learning explores generative modeling and contrastive learning. Techniques like denoising link reconstruction

[22] and GPT-GNN [23] utilize edge perturbation and feature masking for data augmentation, aiming to reconstruct augmented graph features. Graph contrastive learning, exemplified by Deep Graph Infomax [63] and GCC [48], employs feature shuffling and graph sampling to generate contrasting graph samples.

Semi-supervised GDA enhances DGL by using unlabeled data. Key methods include self-training [80], co-training [1], imbalanced training [88], consistency training [72], which aligns node representations between original and augmented graphs, and graph data interpolation [86], creating synthetic examples through feature and label mixing.

Towards Reliable Augmentation: However, most existing techniques apply augmentations uniformly without considering the underlying graph semantics. This can introduce undesirable artifacts by distorting important graph structures and losing interpretability. Recent works have begun to train graphs in latent space [83] or incorporate context and enhanced knowledge when perturbing graphs [65, 30], but generating reliable graphs conditioned on context remains an open challenge.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: Yes we summarized the key contributions in the abstract matching the major challenges in GDA. And the relevant experiments are performed to justify the claims.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We mention that our method os currently limited to offline KG generation in the appendix.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory Assumptions and Proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [NA]

Justification: We do not provide theoretical analysis in this paper.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental Result Reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: Experiment settings, datasets and implementation details are fully disclosed in the main text and the supplementary. We also include the workflow in Figure 2 and Algorithm 1.

Guidelines:

- The answer NA means that the paper does not include experiments.
- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: Codes are anonymously available at `https://anonymous.4open.science/r/GPT-Aug`. Data are all open-sourced.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental Setting/Details**

Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

Answer: [Yes]

Justification: All experiments details are provided in Section 5.1 & 5.2, and additional implementation details are provided in Appendix C.1.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment Statistical Significance**

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [Yes]

Justification: Error bars were reported for major experiments.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).

- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments Compute Resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: The proposed framework is implemented in Python with the *Pytorch* library on a server equipped with four NVIDIA GeForce RTX 3090 GPUs with 32GB memory. Compute amount analysis including time complexity analysis is presented in the Appendix F.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code Of Ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: Yes, we have carefully reviewd the NeurIPS Code of Ethics. Our research conforms the NeurIPS Code of Ethics in every respect.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader Impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: We have discussed the broader impact of the work performed. The discussion can be found in the appendix.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.

- If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: We believe that this paper does not pose such risk. Thus, this question is not applicable for our study.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: We have properly credited every creators of original owners of assets and explicitly mentioned the corresponding licenses in the paper. [TODO: licence of MIMIC3?]

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.

- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New Assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [NA]

    Justification: Our paper does not release new assets. Thus this question is not applicable for our study.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and Research with Human Subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: Our paper does not involve crowdsourcing nor research with human subjects. Thus this question is not applicable for our study.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional Review Board (IRB) Approvals or Equivalent for Research with Human Subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: Our paper does not involve crowdsourcing nor research with human subjects. Thus this question is not applicable for our study.

    Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.