Robust Data-centric Graph Structure Learning for Text Classification

Jun Zhuang
Boise State University
Boise, ID, USA
junzhuang@boisestate.edu

ABSTRACT

Over the past decades, text classification underwent remarkable evolution across diverse domains. Despite these advancements, most existing model-centric methods in text classification cannot generalize well on class-imbalanced datasets that contain high-similarity textual information. Instead of developing new model architectures, data-centric approaches enhance the performance by manipulating the data structure. In this study, we aim to investigate robust data-centric approaches that can help text classification in our collected dataset, the metadata of survey papers about Large Language Models (LLMs). In the experiments, we explore four paradigms and observe that leveraging arXiv's co-category information on graphs can help robustly classify the text data over the other three paradigms, conventional machine-learning algorithms, pre-trained language models' fine-tuning, and zero-shot / few-shot classifications using LLMs.

CCS CONCEPTS

• **Computing methodologies** → *Information extraction.*

KEYWORDS

Data-centric AI; Graph neural networks; Text classification

ACM Reference Format:
Jun Zhuang. 2024. Robust Data-centric Graph Structure Learning for Text Classification. In *Companion Proceedings of the ACM Web Conference 2024 (WWW '24 Companion), May 13–17, 2024, Singapore, Singapore.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3589335.3651915

1 INTRODUCTION

Text classification, as a fundamental task in natural language processing (NLP), has undergone significant evolution over the past few decades in many application fields, such as context understanding [20, 21, 42], content debiasing [61, 62], spam detection [2], and taxonomy generation [25]. Conventional methods transform the text via sparse feature representation, e.g., bag-of-words model [53]. Recently, deep-learning-based approaches, such as long short-term memory (LSTM) [15], have been widely applied to better learn text representations. Subsequent improvements [22, 51] attempt to capture the long-range dependencies of words for textual understanding. Most of these methods improve performances from the angle of model architecture but couldn't generalize well on specific types of text data, such as class-imbalanced data [33] or high-similarity data [31], which are commonly seen in daily lives.

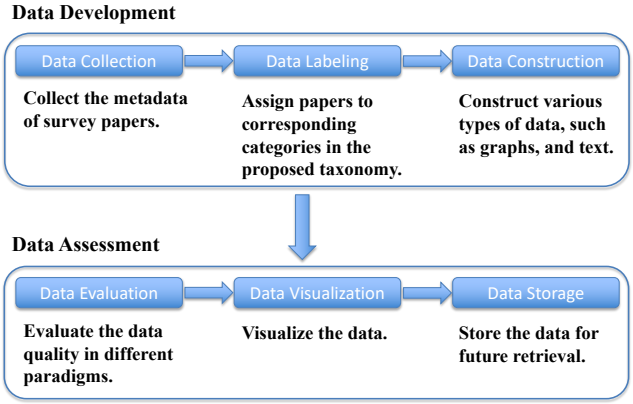


Figure 1: The overall process of our data-centric approaches. The arrow points to the next step in the workflow.

Recent studies reveal that data-centric approaches could be a potential solution to enhance text classification performance [3, 16]. Compared to the model-centric approaches, which aim to design a well-generalized model on the given datasets, data-centric approaches usually optimize the model's outputs by manipulating the dataset [52]. In this study, we aim to investigate robust data-centric approaches that can help improve text classification performance on class-imbalance datasets that contain similar textual information. To illustrate our data-centric approaches, we present the overall process in Figure 1. Our process is mainly divided into two stages, data development and data assessment. In the data development stage, we initially collected the metadata of Large Language Models (LLMs)' survey papers until November 30^{th}, 2023, and then assigned each paper to the corresponding category in our new proposed taxonomy. In our collected dataset, on the one hand, the distribution of each category is not uniform, which leads to a substantial class imbalance issue. On the other hand, authors usually use similar terminologies to describe LLMs in the title and the abstract of these survey papers. Such a textual similarity introduces significant difficulties in text classifications. To embrace these two challenges, we conduct investigations into various types of data, such as attributed graphs and text data. In the data assessment stage, we first evaluate which types of data can yield superior classifications in four paradigms, conventional machine learning algorithms, graph structure learning, fine-tuning the pre-trained language models, and

WWW '24 Companion, May 13–17, 2024, Singapore, Singapore

ACM ISBN 979-8-4007-0172-6/24/05.
https://doi.org/10.1145/3589335.3651915

zero-shot/few-shot classifications using LLMs. Our evaluations reveal that leveraging the graph structure information of co-category graphs can help better classifications over the other three paradigms. After evaluating the data, we visualize various graph structures to illustrate the effectiveness of graph structure learning on co-category graphs. Last, we store our datasets for future retrieval. [1]

Overall, our primary contributions can be summarized as follows:

- We first investigate data-centric approaches that can help text classification on class-imbalance datasets that contain similar textual information.
- We first collect the metadata of 112 literature reviews about Large Language Models (LLMs) and propose a new taxonomy for these papers.
- Extensive experiments indicate that graph structure learning on co-category graphs can robustly classify the text data and substantially outperform the other three paradigms.

2 RELATED WORK

2.1 Data-centric Artificial Intelligence (AI)

The success of AI models is inseparable from a large amount of high-quality annotated data [32, 54]. Compared to improving AI models, an increasing number of research works are dedicated to developing frameworks, commonly named Data-centric AI approaches, that can iteratively improve the data quality for AI systems [52]. Most related papers can be divided into two categories, automatic approaches and collaborative approaches [52]. The automatic approaches aim to automate the process of data manipulation, whereas the collaborative approaches involve human collaboration. Within the former category, the majority of works are classified based on the types of approaches, such as programming-based methods [26, 28], learning-based [18, 43], and pipeline-based methods [12, 38]. In the latter category, most works are assigned based on the extent of human involvement, such as full collaboration [27] or partial collaboration [4].

2.2 Graph Structure Learning

Graph Neural Networks (GNNs) have been widely used for graph structure learning [7, 8, 19, 46, 49, 55, 56, 58, 59]. Bruna et al. [6] first extend convolution operations on graphs using both spatial methods and spectral methods. To improve the efficiency of the eigendecomposition of the graph Laplacian matrix, Defferrard et al. [10] approximate spectral filters by using K-order Chebyshev polynomial. Kipf et al. [23] simplify graph convolutions to a first-order polynomial while achieving state-of-the-art performance for semi-supervised learning. Hamilton et al. [13] propose an inductive-learning approach that aggregates node features from corresponding fixed-size local neighbors. These GNNs have been proven to achieve extraordinary performance in graph structure learning.

2.3 Text Classification

Text Classification has been widely studied in recent years [24, 47, 48, 60]. In the late 20th century, machine-learning models were initially developed to classify text data [39]. Since 2017, Transformer kicked off the era of large language models and has achieved a huge breakthrough in text understanding [45]. On the one hand, by harnessing the power of Transformer [45], BERT [22] can better learn the bidirectional representations, significantly enhancing the performance across a wide range of contextual understanding [35–37]. Subsequent improvements, such as RoBERTa [34], DistilBERT [41], and Albert [29], made substantial contributions to this direction. On the other hand, inspired by the Transformer [45], researchers at OpenAI, introduced a series of Generative Pre-Training (GPT) models, such as GPT-1 [40], that integrate unsupervised pre-training with supervised fine-tuning. With iterative enhancements, GPT-3 achieved human-level classification performance on several NLP benchmarks [5]. GPT-4 extended the capabilities to multi-modal learning and obtained remarkable advancements, leading the development of large language models [1]. Besides employing language models, Yao et al. [51] first explore leveraging graph neural networks in text classification, which sparked enthusiasm for better understanding textual information via graph structure learning [17].

3 METHODOLOGY

In this section, we introduce our data-centric approaches in two stages, data development and data assessment. In the former stage, we introduce the process of data collection, data labeling, and data construction. For the latter stage, we mainly explain the evaluation of graph structure learning.

3.1 Data Development

In the data development stage, we divide the process into data collection, data labeling, and data construction. We introduce each step in detail in this section.

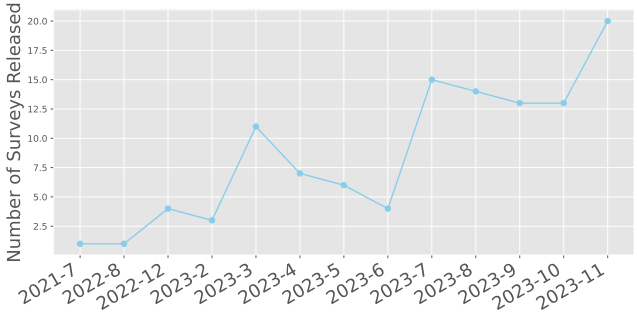


Figure 2: Trends of survey papers on large language models. We focus on the trends of the first released date.

3.1.1 Data Collection. In recent years, large language models attracted more and more attention. Related survey papers have also been continuously emerging in 2023. As shown in Figure 2, the trend has been increasing, with significant growth in March, July, and November of 2023. We scraped the metadata of survey papers about large language models from the arXiv website and further manually supplemented the dataset from Google Scholar. We updated the dataset weekly until November 30, 2023, and collected 112 survey papers in this study.

To better understand the collected papers, we present the word frequency in Figure 3 to show which words have been frequently

[1] Dataset and source codes: https://github.com/junzhuang-code/DCGSL

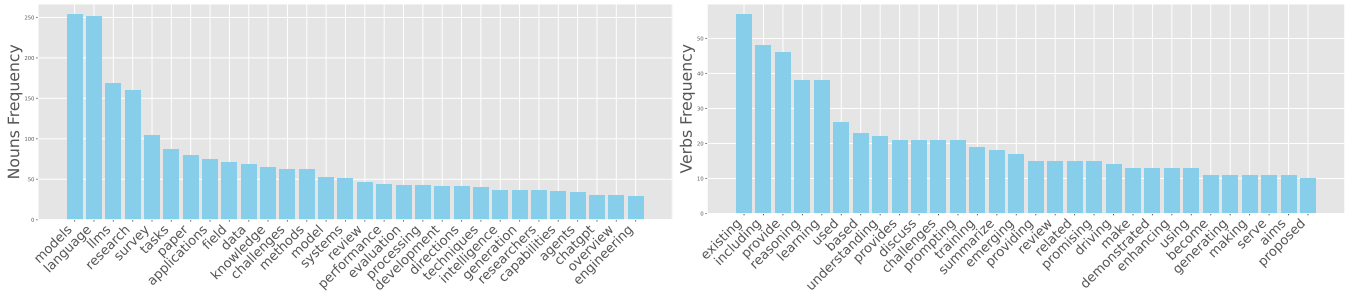


Figure 3: The 30 most frequently occurring noun (left) and verb (right) keywords in the abstract.

used in summary (abstract). These distributions suggest that the abstracts of these papers contain many similar terms, which increases the difficulty of text classification. Thus, the above observation motivates us to explore other methods, such as leveraging the graph structure information, to classify the papers.

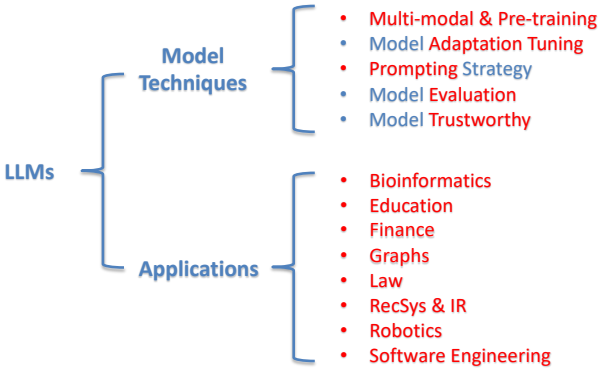


Figure 4: The mind map of survey papers about large language models. Besides "Comprehensive" and "Others" that are not included in the mind map, we highlight thirteen categories in our proposed taxonomy. The total number of classes in the labels is fifteen.

3.1.2 Data Labeling. After collecting data, we further designed a new taxonomy and assigned each paper to the corresponding class. One benefit of providing the taxonomy is that a taxonomy can help newcomers understand the hierarchy of concepts. The mind map of the proposed taxonomy is presented in Figure 4. We highlight thirteen classes in the mind map. The total classes in the labels are fifteen, including "Comprehensive" and "Others" (Not presented in the mind map). To better understand the distribution of the classes, we present the class distribution in Figure 5. The distribution indicates that the class is extremely imbalanced, introducing a challenge to this classification task.

Note that we prefer to propose a new taxonomy instead of using the arXiv categories since the arXiv categories cannot reflect the concept hierarchy for LLMs. To illustrate this point, we present the distribution of survey papers across different arXiv categories in Figure 6. Top-2 frequent categories are "cs.CL" (Computation and Language), and "cs.AI" (Artificial Intelligence), which means that

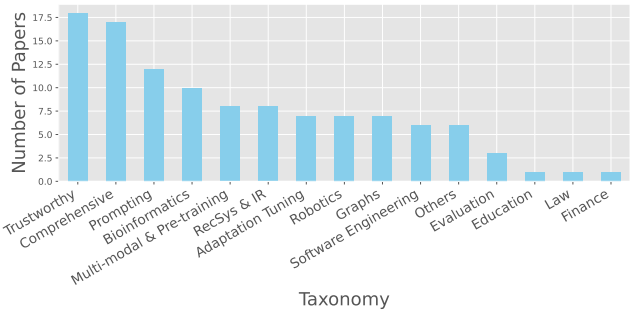


Figure 5: Distribution of classes in the proposed taxonomy.

most authors choose these two categories for their works. However, these two categories don't reflect the model techniques. So, it is essential to propose a new taxonomy in this study.

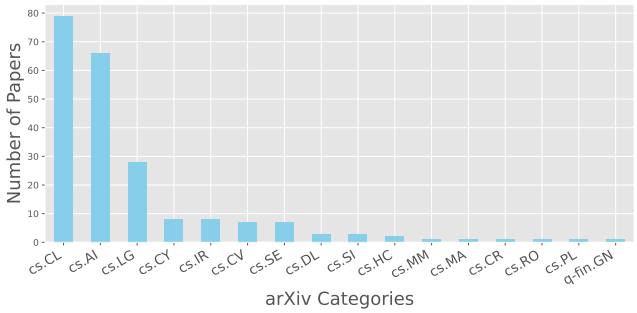


Figure 6: Distribution of arXiv categories in our dataset.

Overall, we present the data description in Table 1.

3.1.3 Data Construction. In the previous section, we explain the motivation for exploring graph structure learning. The goal of constructing attributed graphs is to utilize the graph structure information to classify survey papers into corresponding categories in the proposed taxonomy. Before constructing the graphs, We first define the attributed graphs as follows.

Definition 1. *An attributed graph $\mathcal{G}$ denotes a graph structure that represents topological connections $\mathcal{E}$ among a set of vertices $\mathcal{V}$ associated with attributes. The topological relationship among*

Table 1: Descriptions of data attributes in our dataset.

Attributes	Descriptions
Taxonomy	proposed categories
Title	paper title
Authors	lists of author's name
Release Date	first released date
Links	links of papers
Paper ID	arXiv's paper ID
Categories	arXiv's categories
Summary	abstract of papers

vertices in $\mathcal{G}(\mathcal{V}, \mathcal{E})$ can be represented by a symmetric adjacency matrix $\mathbf{A} \in \mathbb{R}^{N \times N}$, where N is the number of vertices. Each vertex contains an attribute (feature) vector. All feature vectors constitute a feature matrix $\mathbf{X} \in \mathbb{R}^{N \times d}$, where d is the number of features for each vertex. Therefore, the matrix representation of $\mathcal{G}(\mathcal{V}, \mathcal{E})$ can be defined as $\mathcal{G}(\mathbf{A}, \mathbf{X})$.

Based on Definition 1, we start by creating the term frequency-inverse document frequency (TF-IDF) feature matrices for both title and summary columns, where the term frequency denotes the word frequency in the document, and inverse document frequency denotes the log-scaled inverse fraction of the number of documents containing the word. TF-IDF matrix is commonly used for text classification tasks because it helps capture the distinctive words that can indicate specific classes. After establishing the TF-IDF matrices, we apply one-hot encoding on the arXiv's categories and then combine three matrices along the feature dimension to build the feature matrix $\mathbf{X}$.

To leverage the topological information among vertices, we proceed to construct the graph structures to connect the attribute vectors. In this study, we are interested in three types of graphs, text graph, co-author graph, and co-category graph.

To enhance the text classification, Yao et al. [51] initially verified that long-distance lexical relationships can be effectively represented in a text graph. Thus, in the work, we follow the same settings as TextGCN [51] to build text graphs. Note that in the text graph, the aforementioned feature matrix remains unutilized as only paper vertices contain attribute vectors. To retain consistency, all entries in the feature matrix are uniformly set to unity. Correspondingly, solely paper vertices are endowed with labels, while all word vertices are uniformly assigned a new class, which remains untouched throughout both the training and testing phases.

Exploring the co-relationship among vertices is a common practice in graph structure learning [13]. In our dataset, two attributes, "Authors" and "Categories", can be utilized to explore such co-relationships as these attributes exhibit inherent connections among survey papers. Thus, we build co-author graphs and co-category graphs using these two attributes. In the co-author graph, we introduce an edge connecting two vertices (papers) if they share at least one common author. In the co-category graph, an edge is added between two vertices with at least one common category. In these two types of graphs, each vertex is assigned one class (taxonomy) as the label. Note that in this study all edges are undirected.

Besides constructing graphs, we compare the performances on text data, which includes both the title and abstract of survey papers.

3.2 Data Assessment

In this section, we mainly introduce how we evaluate the classification performances on our constructed attributed graphs. Moreover, we provide additional evaluation for the other three paradigms in the experiment section. After evaluating the data, we further visualize the graphs and store the datasets during the process.

3.2.1 Graph Structure Learning in Text Classification. Given the well-built attributed graphs $\mathcal{G}(\mathbf{A}, \mathbf{X})$, we aim to investigate whether data-centric graph structure learning using graph neural networks (GNNs) can help text classification. Before feeding the matrix representation, $\mathbf{A}$ and $\mathbf{X}$, of the attributed graphs $\mathcal{G}$ into GNNs, we first preprocess the adjacency matrix $\mathbf{A}$ as follows:

$$\hat{\mathbf{A}} = \tilde{\mathbf{D}}^{-\frac{1}{2}} \tilde{\mathbf{A}} \tilde{\mathbf{D}}^{-\frac{1}{2}}, \tag{1}$$

where $\tilde{\mathbf{A}} = \mathbf{A} + I_N$, $\tilde{\mathbf{D}} = \mathbf{D} + I_N$. I_N is an identity matrix. $\mathbf{D}_{i,i} = \sum_j \mathbf{A}_{i,j}$ is a diagonal degree matrix.

After preprocessing, we utilize GNNs to learn node representation. Note that in this study, a node in graphs could represent a word or a document. By doing so, we could transform the text classification tasks into the node classification tasks. This transformation underscores the versatility of GNNs in handling diverse tasks. Within these tasks, the layer-wise message-passing mechanism of GNNs serves as a foundation to capture intricate relationships in graph-structured data. For general expression, we formulate the layer-wise message-passing mechanism of GNNs as follows:

$$f_{\mathbf{W}^{(l)}}\left(\hat{\mathbf{A}}, \mathbf{H}^{(l)}\right) = \sigma\left(\hat{\mathbf{A}}\mathbf{H}^{(l)}\mathbf{W}^{(l)}\right), \tag{2}$$

where $\mathbf{H}^{(l)}$ is a node hidden representation in the l-th layer. The dimension of $\mathbf{H}^{(l)}$ in the input layer, middle layer, and output layer is the number of features d, hidden units h, and classes K, respectively. $\mathbf{H}^{(0)} = \mathbf{X}$. $\mathbf{W}^{(l)}$ is the weight matrix in the l-th layer. σ denotes a non-linear activation function, such as ReLU.

In general node classification tasks, a GNN is trained with ground-truth labels $\mathbf{Y} \in \mathbb{R}^{N \times 1}$. In this study, we build the ground-truth labels based on our proposed taxonomy. To simplify the problem, each paper is assigned one primary category as the label, even if the paper sometimes may belong to more than one category. During training, we optimize GNNs with cross-entropy as follows.

$$\mathcal{L}_{ce} = -\frac{1}{N_{tr}} \sum_{i=1}^{N_{tr}} \sum_{k=1}^{K} y_{i,k} log(\hat{y}_{i,k}), \tag{3}$$

where $y_{i,k}$ denotes a ground-truth label of the i-th node in the k-th class; $\hat{y}_{i,k}$ denotes a predicted label of the i-th node in the k-th class; N_{tr} denotes the number of train nodes; K denotes the number of classes. The i-th predicted label $\hat{y}_i$ is computed by choosing the maximum probability of the corresponding categorical distribution.

In brief, we formalize the problem that we aim to solve via data-centric graph structure learning in this study as follows.

Problem 1. *After constructing an attributed graph $\mathcal{G}(\hat{\mathbf{A}}, \mathbf{X})$ and ground-truth labels $\mathbf{Y}$, we train a graph neural network (GNN) on the train data and evaluate the classification performance on the test data. Our goal is to design a data-centric method that can help robustly classify the text data.*

Table 2: Evaluation of data-centric graph structure learning on three types of attributed graphs. We also conduct an ablation study on the graph structure of co-category graphs. Rm denotes "Removed".

	GCN		GraphSAGE		GIN		TAGCN	
	Accuracy	Weighted-F1	Accuracy	Weighted-F1	Accuracy	Weighted-F1	Accuracy	Weighted-F1
Text	25.45 (3.64)	18.11 (2.39)	21.82 (12.33)	14.54 (9.07)	16.36 (2.23)	10.76 (2.08)	4.64 (1.40)	3.90 (1.28)
Co-author	25.22 (8.43)	26.59 (10.20)	33.91 (6.96)	34.09 (7.97)	26.96 (11.47)	26.04 (11.30)	35.65 (10.07)	32.54 (10.56)
Co-category (All)	77.39 (7.48)	74.73 (9.81)	75.65 (3.48)	75.24 (3.07)	69.57 (7.08)	**69.32 (9.42)**	72.17 (5.90)	69.65 (5.51)
Co-category (Rm cs.CL)	75.65 (9.37)	73.39 (10.21)	75.65 (8.06)	73.85 (9.08)	67.39 (9.12)	65.37 (8.47)	66.96 (8.06)	63.76 (9.50)
Co-category (Rm cs.AI)	**80.00 (8.95)**	**77.93 (10.11)**	**80.87 (13.07)**	**78.77 (14.34)**	**70.87 (8.06)**	67.54 (6.10)	**73.91 (9.53)**	**71.80 (8.27)**
Co-category (Rm cs.CL, cs.AI)	29.57 (5.07)	28.11 (5.28)	32.17 (3.48)	27.07 (3.52)	19.13 (10.51)	15.85 (8.16)	41.74 (13.07)	40.03 (14.02)
Co-category (Rm cs.IR)	77.39 (6.39)	75.49 (7.33)	73.91 (3.89)	73.21 (4.43)	**69.57 (7.08)**	**69.32 (9.42)**	**72.17 (5.90)**	**69.65 (5.51)**
Co-category (Rm cs.RO)	77.39 (4.26)	75.15 (5.88)	**75.65 (4.43)**	74.00 (5.02)	**69.57 (7.08)**	**69.32 (9.42)**	**72.17 (5.90)**	**69.65 (5.51)**
Co-category (Rm cs.SE)	75.65 (8.06)	73.46 (8.16)	**75.65 (2.13)**	**74.82 (2.28)**	68.70 (9.28)	64.06 (10.62)	71.30 (10.14)	67.98 (8.37)
Co-category (Rm cs.IR, cs.RO)	76.52 (4.43)	73.99 (6.42)	73.04 (5.07)	70.89 (5.88)	**69.57 (7.08)**	**69.32 (9.42)**	**72.17 (5.90)**	**69.65 (5.51)**
Co-category (Rm cs.IR, cs.SE)	77.39 (6.39)	75.49 (7.33)	74.78 (5.07)	73.53 (5.59)	68.70 (9.28)	64.06 (10.62)	71.30 (10.14)	67.98 (8.37)
Co-category (Rm cs.RO, cs.SE)	**78.26 (4.76)**	**76.39 (6.85)**	74.78 (5.07)	72.53 (6.10)	68.70 (9.28)	64.06 (10.62)	71.30 (10.14)	67.98 (8.37)
Co-category (Rm cs.IR, cs.RO, cs.SE)	**78.26 (4.76)**	**76.39 (6.85)**	**75.65 (5.90)**	74.26 (7.02)	68.70 (9.28)	64.06 (10.62)	71.30 (10.14)	67.98 (8.37)

4 EXPERIMENTS

In this section, we verify the effectiveness and robustness of graph structure learning for text classification in our dataset. We further examine its superior performance over the other three paradigms.

4.1 Experimental Settings

Table 3: Statistics of graph datasets. $|\mathcal{V}|$, $|\mathcal{E}|$, $|F|$, and $|C|$ denote the number of nodes, edges, features, and classes, respectively.

Dataset	$\lvert\mathcal{V}\rvert$	$\lvert\mathcal{E}\rvert$	$\lvert F\rvert$	$\lvert C\rvert$
Text	737	95,987	737	16
Co-author	112	204	3,065	15
Co-category (All)	112	4,904	3,065	15

We investigate three types of attributed graphs, text graphs, co-author graphs, and co-category graphs, for graph structure learning and present their statistics in Table 3. Note that the text graph consists of paper vertices and word vertices, and thus contains 16 classes because all word vertices are labeled as a new class, which is not touched during the training or testing phase. In the comparative analysis, we examine the classic machine-learning algorithms on the above feature matrix and evaluate the language models on the text data, which contains both title and summary.

To validate our methods, we split the train, validation, and test data as 60%, 20%, and 20%. After the split, we're aware that different splits will highly affect the performance on such a small dataset. So, we ran the experiments five times using random seed IDs from 0 to 4 and reported the mean values with corresponding standard deviations, mean (std).

We evaluate the classification performance by accuracy and weighted f1 score. Accuracy is a common metric on classification tasks, whereas the weighted f1 score provides a balanced measure of the class-imbalanced dataset.

4.2 Data-centric Graph Structure Learning Can Help Text Classification.

We investigate whether leveraging the graph structure information can robustly help classify the text data in our dataset. In this experiment, we build graph structures based on the text data (including the title and summary) and the relationship of co-author and co-category. After building the graphs, we examine various graph structures on four classic graph neural networks, GCN [23], GraphSAGE [13], GIN [49], and TAGCN [11].

According to the results in Table 2, four GNNs fail to learn graph representation on both the text graph and the co-author graph. For the text graph, we argue that the degradation of GNNs may be caused by excessively similar words in the summary of survey papers. When constructing the text graph, these word vertices connect with many paper vertices, resulting in the paper vertices being less distinguishable. For the co-author graph, we conjecture that it is challenging to categorize papers solely based on the sparse co-authorship in this dataset.

On the contrary, four GNNs can achieve great performance (evaluated by both accuracy and weighted F1 score) in most co-category graphs. We conducted an ablation study to examine various graph structures of co-category graphs. First, according to Figure 6, most papers are assigned as "cs.CL" and "cs.AI" in the arXiv categories. Thus, we study how the categories, "cs.CL" and "cs.AI", affect the performance by muting these two categories in a combinatorial manner. In Table 2, we observe that GNNs can maintain a comparable performance after removing either "cs.CL" or "cs.AI". However, the performance dramatically drops after removing both categories. This is possible since most node connections are significantly sparsified after these two categories are removed. In other words, even though both "cs.CL" and "cs.AI" do not directly map to the existing classes, either one can help connect the nodes and further strengthen the message-passing mechanism in GNNs, allowing GNNs to learn better node representation.

We visualize the co-category graphs in Figure 7. The visualization indicates that most nodes, such as the nodes labeled with

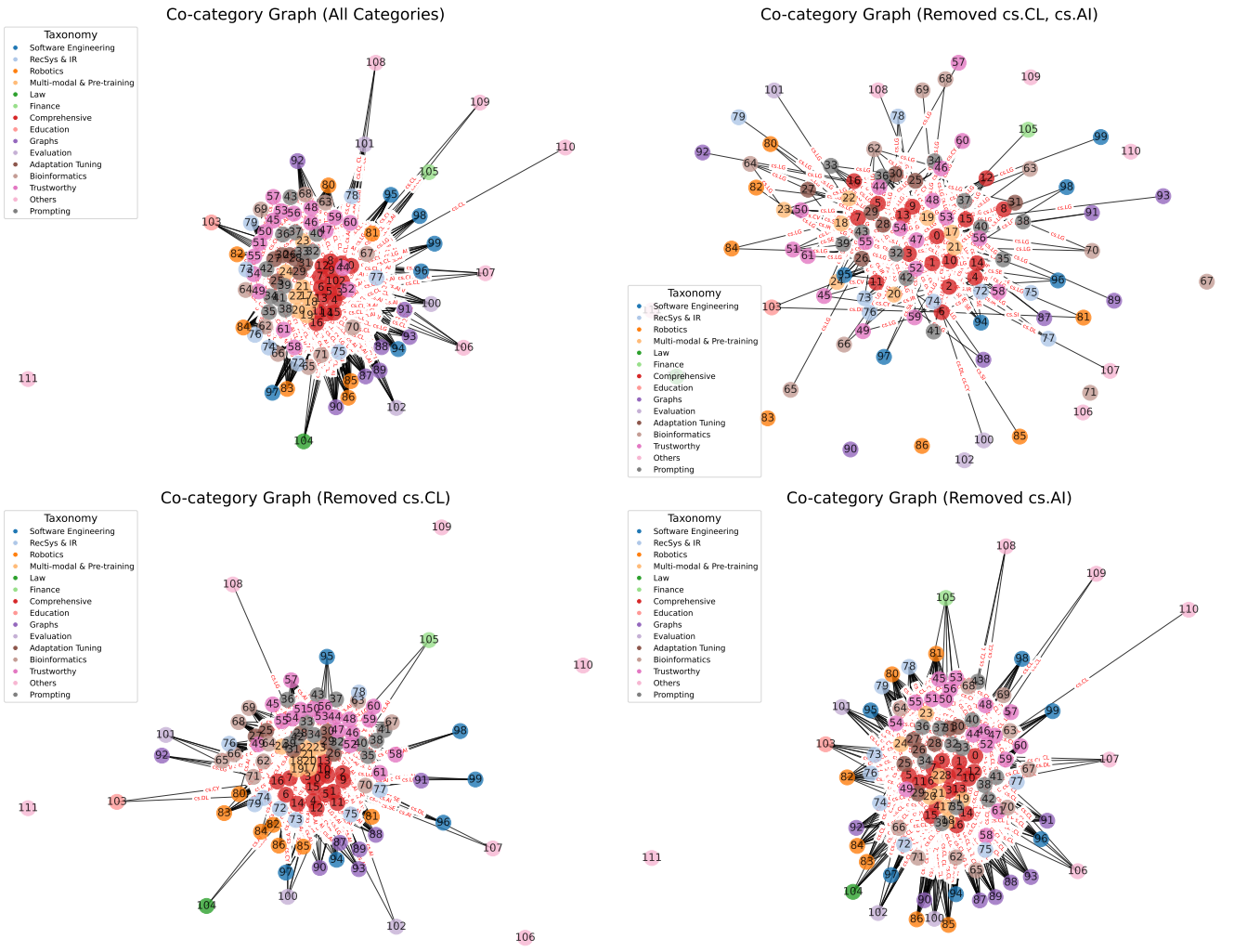


Figure 7: Visualization of co-category graphs. We visualize the graphs by muting the categories.

red or pink color, are clustered well even if we remove the category either "cs.CL" or "cs.AI". However, after removing these two categories simultaneously, we observe that node classifications gradually become disordered and many nodes are then isolated. This visualization helps us intuitively understand the effectiveness of graph structure learning.

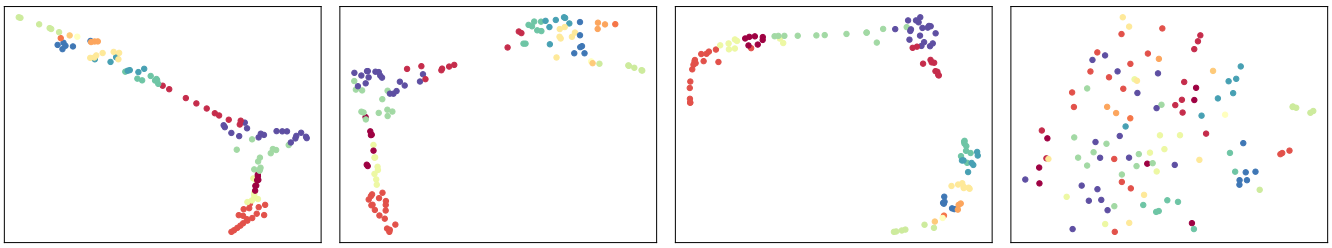
We also visualize GCNs' hidden representation on the above co-category graphs in Figure 8, which shows that nodes are well-classified in the hidden space even if either the category "cs.CL" or "cs.AI" is removed. However, the distribution of nodes tends to become chaotic when these two categories are removed simultaneously. The visualization verifies experimental results in Table 2.

To further assess the robustness of graph structure learning, we conducted another ablation study to examine how the categories, "cs.IR", "cs.SE", and "cs.RO", affect the classification performance as their names are similar to that of some classes in our proposed taxonomy. Note that our proposed taxonomy is not based on the arXiv categories. According to Table 2, the performances are well-maintained no matter which category is removed. We argue that results are reasonable since the removals only drop a small number of edges and don't break the topological connections in the graph.

Besides examining various graph structures, we compare the performance of graph structure learning under different noise ratios (nr) in the train labels. Even though it's expected that the classification accuracy decreases as the noise ratio increases, the results in Figure 9 indicate that learning through co-category graphs can achieve robust performance across different noise ratios and stably outperform the other two graph structures. Overall, the above experiments verify the robustness of graph structure learning on co-category graphs.

4.3 Comparative Analysis

After verifying the effectiveness of graph structure learning, we further investigate the performance of several classic models in



(a) All Categories (b) Removed cs.CL (c) Removed cs.AI (d) Removed cs.CL, cs.AI

Figure 8: Visualization of GCNs' hidden representation on co-category graphs in 2-dimension via t-SNE. Each dot represents one node and is labeled with one color.

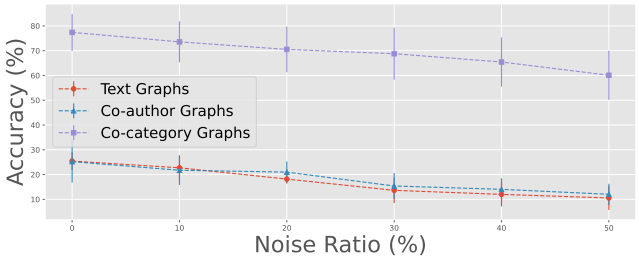


Figure 9: Comparison of three types of graph structures under different noise ratios (nr).

three different paradigms on the classification tasks. Specifically, we first employ classic machine learning algorithms on the feature matrix (without leveraging the topological relationships). Second, we fine-tune the pre-trained language models on the text data for the downstream classification tasks. Third, we evaluate the zero-shot / few-shot classification capabilities of large language models.

Table 4: Evaluation of classic machine learning algorithms on the feature matrix. We denote NB, SVM, RF, and GB as Naïve Bayes Classifiers, Support Vector Machines, Random Forest, and Gradient Boosting, respectively.

Algorithms	Accuracy	Weighted-F1
NB	**39.13 (8.70)**	**36.82 (9.49)**
SVM	21.74 (7.28)	14.92 (7.66)
RF	20.87 (4.26)	13.54 (3.29)
GB	33.91 (7.48)	32.36 (8.15)

We first examine four classic machine-learning algorithms, Naïve Bayes Classifiers (NB), Support Vector Machines (SVM), Random Forest (RF), and Gradient Boosting (GB), and present the results of the first paradigm in Table 4. The results indicate that these machine-learning algorithms cannot perform well on this task.

Second, we examine whether fine-tuning the pre-trained language models on the text data can help achieve better classification. The results in Table 5 indicate that medium-size language models, such as DistilBERT [41], can achieve better performance on smaller text data. However, the performance may dramatically drop when

Table 5: Evaluation of fine-tuning the pre-trained language models on the text data.

Language Models	Model Size	Accuracy	Weighted-F1
BERT [22]	109.49M	43.48 (11.67)	41.50 (13.50)
RoBERTa [34]	124.66M	40.87 (14.18)	39.68 (19.51)
DistilBERT [41]	66.97M	**55.65 (9.28)**	**53.55 (11.25)**
XLNet [50]	117.32M	30.43 (20.76)	25.72 (23.00)
Electra [9]	109.49M	33.04 (5.22)	31.72 (4.87)
Albert [29]	11.70M	11.30 (8.06)	6.56 (7.50)
BART [30]	140.02M	48.70 (3.25)	47.77 (7.58)
DeBERTa [14]	139.20M	28.70 (11.20)	25.48 (13.21)
Llama2 [44]	6.61B	14.49 (8.93)	4.72 (4.43)

the model size is too small, such as Albert [29]. We argue that fine-tuning larger pre-trained language models, such as Llama2 [44], on smaller text data may cause overfitting issues, which leads to worse performance on larger models.

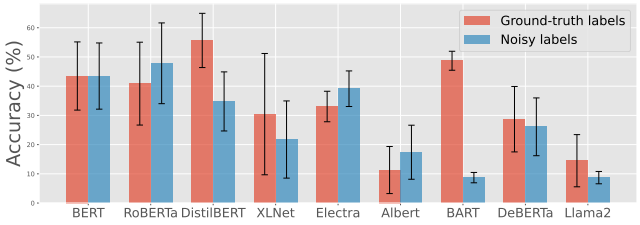


Figure 10: Comparison of fine-tuning the language models using ground-truth labels and noisy labels.

We further investigate whether leveraging noisy labels can help fine-tuning. Our previous experiments confirmed that graph structure learning can outperform fine-tuning in this classification task. Thus, we first generate noisy labels by GCN and then fine-tune the pre-trained language models with the noisy labels. The results in Figure 10 indicate that for some models, the performance achieved through training with noisy labels can surpass that of training with ground-truth labels. One possible reason is that training the model using noisy labels with a low noise ratio can be equivalent to a kind of regularization, improving the classification performance [57].

Table 6: Evaluation of zero-shot and few-shot classification capabilities of three large language models, Claude, ChatGPT 3.5, and ChatGPT 4.

	Accuracy	Weighted-F1
Claude w.o. hints	11.61 (0.90)	12.61 (0.17)
Claude w. hints	10.27 (2.23)	12.79 (1.52)
ChatGPT 3.5 w.o. hints	47.39 (3.38)	43.19 (4.57)
ChatGPT 3.5 w. hints	**53.56 (2.98)**	**53.14 (3.02)**
ChatGPT 4 w.o. hints	29.76 (5.89)	26.87 (7.86)
ChatGPT 4 w. hints	33.04 (4.55)	27.76 (6.32)

Third, we evaluate the zero-shot and few-shot classification capabilities of three large language models, Claude, ChatGPT 3.5, and ChatGPT 4. We ran the experiments five times and presented the mean value with the corresponding standard deviation in Table 6. Among the large language models, ChatGPT 3.5 outperforms the other two models given that all models have not seen the data before (zero-shot). We further provide some hints to the models before classification (few-shot). For example, we release the keywords of the class "Trustworthy" to the models before classification. In this setting, both ChatGPT 3.5 and ChatGPT 4 can achieve higher accuracy and a weighted F1 score after obtaining some hints. Overall, all three LLMs cannot outperform graph structure learning, which reveals that in this task, LLMs still have room to improve.

4.4 Limitation

The experimental results in this study have demonstrated the effectiveness of leveraging graph structure information to classify the survey papers. However, constructing a graph structure may encounter certain constraints. For instance, we build co-category graphs based on the arXiv categories. When papers come from distinct fields, such as biology, physics, and computer science, the graph structure may be very sparse, weakening the effectiveness of graph structure learning.

4.5 Future Directions

In the future, our primary motivation extended from this study is to tailor GPT-based applications to assist readers in understanding survey papers more effectively. On the other hand, our collected datasets can potentially contribute to node alignment tasks, which involve the alignment of nodes in one or more graphs, such as co-category graphs and co-author graphs in this study.

5 CONCLUSION

In this study, we aim to investigate data-centric approaches that can help text classification on class-imbalance datasets that contain similar textual information. To build such a dataset, we collected the metadata of 112 LLMs' survey papers. In the experiments, we conduct a comparative analysis across four paradigms and demonstrate that graph structure learning outperforms conventional machine-learning algorithms, pre-trained language models' fine-tuning, and zero-shot / few-shot classifications using LLMs. Within graph structure learning, we explore three types of attributed graphs, text graph, co-author graph, and co-category graph, and observe that leveraging arXiv's co-category information can help robustly classify text data in our dataset.

A APPENDIX

In the appendix, we present the hyper-parameters and settings for the models in this study, and the hardware and software.

A.1 Hyper-parameters and Settings

Graph Structure Learning. We examine the effectiveness of graph structure learning on four classic two-layer GNNs, GCN [23], GraphSAGE [13], GIN [49], and TAGCN [11], with 200 hidden units and a ReLU activation function. GNNs are trained by the Adam optimizer with a learning rate, 1×10^{-2} for both co-author graphs and co-category graphs and 2×10^{-2} for text graphs, and converged within 500 training epochs on all datasets. The dropout rate is 0.5. We chose "gcn" and "mean" aggregators for GraphSAGE and GIN, and fixed the number of filters as 3 for TAGCN.

Table 7: The optimal parameters of random forest (RF) and gradient boosting (GB) for each data split.

	Parameters	Values
RF	Max depth	20, 20, 20, 10, 10
	Min samples leaf	1, 1, 2, 2, 1
	Min samples split	10, 5, 10, 5, 10
	Number of estimators	150, 250, 250, 200, 200
GB	Max depth	3, 4, 5, 3, 4
	Min samples leaf	2, 2, 2, 2, 4
	Min samples split	3, 5, 5, 5, 3
	Number of estimators	50, 50, 50, 50, 50

Machine-learning Algorithms. We apply grid search methods with four-fold cross-validation to tune the machine-learning algorithms and report the optimal parameters for each data split (random seed ID is from 0 to 4) as follows. For Naïve Bayes Classifiers, we choose the α as 1.0, 0.5, 0.7, 0.3, and 0.5, respectively. For Support Vector Machines, we choose the regularization parameter C as 0.1 and the linear kernel for all splits. For Random Forest and Gradient Boosting, we present their optimal parameters for each data split in Table 7.

Pre-trained Language Models' Fine-tuning. We fine-tune pre-trained language models using the Adam optimizer with a 1×10^{-4} learning rate. We chose the batch size of 8 for Llama2 and fixed the batch size of 16 for the rest of the models.

A.2 Hardware and Software

All experiments are conducted on the server with the following configurations:

- Operating System: Ubuntu 22.04.3 LTS
- CPU: Intel Xeon w5-3433 @ 4.20 GHz
- GPU: NVIDIA RTX A6000 48GB
- Software: Python 3.11, PyTorch 2.1, HuggingFace 4.31, dgl 1.1.2+cu118.

REFERENCES

[1] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. 2023. GPT-4 Technical Report. *arXiv preprint arXiv:2303.08774* (2023).

[2] Hadeer Ahmed, Issa Traore, and Sherif Saad. 2018. Detecting opinion spams and fake news using text classification. *Security and Privacy* 1, 1 (2018), e9.

[3] Samar Bashath, Nadeesha Perera, Shailesh Tripathi, Kalifa Manjang, Matthias Dehmer, and Frank Emmert Streib. 2022. A data-centric review of deep transfer learning with applications to text data. *Information Sciences* 585 (2022), 498–528.

[4] Alex Bogatu, Alvaro AA Fernandes, Norman W Paton, and Nikolaos Konstantinou. 2020. Dataset discovery in data lakes. In *2020 IEEE 36th International Conference on Data Engineering (ICDE)*. IEEE, 709–720.

[5] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems* 33 (2020), 1877–1901.

[6] Joan Bruna, Wojciech Zaremba, Arthur Szlam, and Yann LeCun. 2013. Spectral networks and locally connected networks on graphs. *arXiv preprint arXiv:1312.6203* (2013).

[7] Hanning Chen, Ali Zakeri, Fei Wen, Hamza Errahmouni Barkam, and Mohsen Imani. 2023. HyperGRAF: Hyperdimensional Graph-Based Reasoning Acceleration on FPGA. In *2023 33rd International Conference on Field-Programmable Logic and Applications (FPL)*. IEEE, 34–41.

[8] Ziheng Chen, Fabrizio Silvestri, Jia Wang, Yongfeng Zhang, Zhenhua Huang, Hongshik Ahn, and Gabriele Tolomei. 2022. Grease: Generate factual and counterfactual explanations for gnn-based recommendations. *arXiv preprint arXiv:2208.04222* (2022).

[9] Kevin Clark, Minh-Thang Luong, Quoc V Le, and Christopher D Manning. 2019. ELECTRA: Pre-training Text Encoders as Discriminators Rather Than Generators. In *International Conference on Learning Representations*.

[10] Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. 2016. Convolutional neural networks on graphs with fast localized spectral filtering. In *Advances in neural information processing systems*. 3844–3852.

[11] Jian Du, Shanghang Zhang, Guanhang Wu, José MF Moura, and Soummya Kar. 2017. Topology adaptive graph convolutional networks. *arXiv preprint arXiv:1710.10370* (2017).

[12] Matthias Feurer, Aaron Klein, Katharina Eggensperger, Jost Springenberg, Manuel Blum, and Frank Hutter. 2015. Efficient and robust automated machine learning. *Advances in neural information processing systems* 28 (2015).

[13] Will Hamilton, Zhitao Ying, and Jure Leskovec. 2017. Inductive representation learning on large graphs. In *Advances in neural information processing systems*. 1024–1034.

[14] Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2020. Deberta: Decoding-enhanced bert with disentangled attention. *arXiv preprint arXiv:2006.03654* (2020).

[15] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* 9, 8 (1997), 1735–1780.

[16] Chengyue Huang, Anindita Bandyopadhyay, Weiguo Fan, Aaron Miller, and Stephanie Gilbertson-White. 2023. Mental toll on working women during the COVID-19 pandemic: An exploratory study using Reddit data. *PloS one* 18, 1 (2023), e0280049.

[17] Lianzhe Huang, Dehong Ma, Sujian Li, Xiaodong Zhang, and Houfeng Wang. 2019. Text level graph neural network for text classification. *arXiv preprint arXiv:1910.02356* (2019).

[18] Wei Jin, Xiaorui Liu, Xiangyu Zhao, Yao Ma, Neil Shah, and Jiliang Tang. 2021. Automated self-supervised learning for graphs. *arXiv preprint arXiv:2106.05470* (2021).

[19] Wei Jin, Yao Ma, Xiaorui Liu, Xianfeng Tang, Suhang Wang, and Jiliang Tang. 2020. Graph structure learning for robust graph neural networks. In *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining*. 66–74.

[20] Xin Jin, Sunil Manandhar, Kaushal Kafle, Zhiqiang Lin, and Adwait Nadkarni. 2022. Understanding iot security from a market-scale perspective. In *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security*. 1615–1629.

[21] Xin Jin and Yuchen Wang. 2023. Understand legal documents with contextualized large language models. *arXiv preprint arXiv:2303.12135* (2023).

[22] Jacob Devlin Ming-Wei Chang Kenton and Lee Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of naacL-HLT*, Vol. 1. 2.

[23] Thomas N Kipf and Max Welling. 2016. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907* (2016).

[24] Kamran Kowsari, Kiana Jafari Meimandi, Mojtaba Heidarysafa, Sanjana Mendu, Laura Barnes, and Donald Brown. 2019. Text classification algorithms: A survey. *Information* 10, 4 (2019), 150.

[25] Raghu Krishnapuram and Krishna Kummamuru. 2003. Automatic taxonomy generation: Issues and possibilities. In *International Fuzzy Systems Association World Congress*. Springer, 52–63.

[26] Arun Kumar, Jeffrey Naughton, Jignesh M Patel, and Xiaojin Zhu. 2016. To join or not to join? thinking twice about joins before feature selection. In *Proceedings of the 2016 International Conference on Management of Data*. 19–34.

[27] Mucahid Kutlu, Tyler McDonnell, Tamer Elsayed, and Matthew Lease. 2020. Annotator rationales for labeling tasks in crowdsourcing. *Journal of Artificial Intelligence Research* 69 (2020), 143–189.

[28] Kwei-Herng Lai, Daochen Zha, Junjie Xu, Yue Zhao, Guanchu Wang, and Xia Hu. 2021. Revisiting time series outlier detection: Definitions and benchmarks. In *Thirty-fifth conference on neural information processing systems datasets and benchmarks track (round 1)*.

[29] Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2019. ALBERT: A Lite BERT for Self-supervised Learning of Language Representations. In *International Conference on Learning Representations*.

[30] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. 7871–7880.

[31] Baoli Li and Liping Han. 2013. Distance weighted cosine similarity measure for text classification. In *Intelligent Data Engineering and Automated Learning–IDEAL 2013: 14th International Conference, IDEAL 2013, Hefei, China, October 20-23, 2013. Proceedings 14*. Springer, 611–618.

[32] Haoyang Liu, Maheep Chaudhary, and Haohan Wang. 2023. Towards trustworthy and aligned machine learning: A data-centric survey with causality perspectives. *arXiv preprint arXiv:2307.16851* (2023).

[33] Ying Liu, Han Tong Loh, and Aixin Sun. 2009. Imbalanced text classification: A term weighting approach. *Expert systems with Applications* 36, 1 (2009), 690–701.

[34] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692* (2019).

[35] Weimin Lyu, Xinyu Dong, Rachel Wong, Songzhu Zheng, Kayley Abell-Hart, Fusheng Wang, and Chao Chen. 2022. A multimodal transformer: Fusing clinical notes with structured EHR data for interpretable in-hospital mortality prediction. In *AMIA Annual Symposium Proceedings*, Vol. 2022. American Medical Informatics Association, 719.

[36] Weimin Lyu, Songzhu Zheng, Tengfei Ma, and Chao Chen. 2022. A Study of the Attention Abnormality in Trojaned BERTs. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. 4727–4741.

[37] Weimin Lyu, Songzhu Zheng, Lu Pang, Haibin Ling, and Chao Chen. 2023. Attention-Enhancing Backdoor Attacks Against BERT-based Models. In *Findings of the Association for Computational Linguistics: EMNLP 2023*. 10672–10690.

[38] Dhruv Mahajan, Ross Girshick, Vignesh Ramanathan, Kaiming He, Manohar Paluri, Yixuan Li, Ashwin Bharambe, and Laurens Van Der Maaten. 2018. Exploring the limits of weakly supervised pretraining. In *Proceedings of the European conference on computer vision (ECCV)*. 181–196.

[39] Andrew McCallum, Kamal Nigam, et al. 1998. A comparison of event models for naive bayes text classification. In *AAAI-98 workshop on learning for text categorization*, Vol. 752. Madison, WI, 41–48.

[40] Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, et al. 2018. Improving language understanding by generative pre-training. (2018).

[41] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108* (2019).

[42] Saurabh Srivastava, Chengyue Huang, Weiguo Fan, and Ziyu Yao. 2023. Instance Needs More Care: Rewriting Prompts for Instances Yields Better Zero-Shot Performance. *arXiv preprint arXiv:2310.02107* (2023).

[43] Qiaoyu Tan, Guoxian Yu, Carlotta Domeniconi, Jun Wang, and Zili Zhang. 2018. Incomplete multi-view weak-label learning.. In *Ijcai*. 2703–2709.

[44] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288* (2023).

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

[46] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph Attention Networks. In *International Conference on Learning Representations*.

[47] Jun Wu, Xuesong Ye, and Yanyuet Man. 2023. Bottrinet: A unified and efficient embedding for social bots detection via metric learning. In *2023 11th International Symposium on Digital Forensics and Security (ISDFS)*. IEEE, 1–6.

[48] Jun Wu, Xuesong Ye, Chengjie Mou, and Weinan Dai. 2023. Fineehr: Refine clinical note representations to improve mortality prediction. In *2023 11th International Symposium on Digital Forensics and Security (ISDFS)*. IEEE, 1–6.

[49] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. 2018. How powerful are graph neural networks? *arXiv preprint arXiv:1810.00826* (2018).

[50] Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. XLNet: Generalized Autoregressive Pretraining for Language Understanding. *Advances in Neural Information Processing Systems* 32 (2019).

[51] Liang Yao, Chengsheng Mao, and Yuan Luo. 2019. Graph convolutional networks for text classification. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 33. 7370–7377.

[52] Daochen Zha, Zaid Pervaiz Bhat, Kwei-Herng Lai, Fan Yang, and Xia Hu. 2023. Data-centric ai: Perspectives and challenges. In *Proceedings of the 2023 SIAM International Conference on Data Mining (SDM)*. SIAM, 945–948.

[53] Yin Zhang, Rong Jin, and Zhi-Hua Zhou. 2010. Understanding bag-of-words model: a statistical framework. *International journal of machine learning and cybernetics* 1 (2010), 43–52.

[54] Tong Zhao, Wei Jin, Yozen Liu, Yingheng Wang, Gang Liu, Stephan Günnemann, Neil Shah, and Meng Jiang. 2022. Graph data augmentation for graph machine learning: A survey. *arXiv preprint arXiv:2202.08871* (2022).

[55] Jun Zhuang and Mohammad Al Hasan. 2022. Defending Graph Convolutional Networks against Dynamic Graph Perturbations via Bayesian Self-Supervision. *Proceedings of the AAAI Conference on Artificial Intelligence* 36, 4 (Jun. 2022), 4405–4413. https://doi.org/10.1609/aaai.v36i4.20362

[56] Jun Zhuang and Mohammad Al Hasan. 2022. Deperturbation of Online Social Networks via Bayesian Label Transition. In *Proceedings of the 2022 SIAM International Conference on Data Mining (SDM)*. SIAM, 603–611.

[57] Jun Zhuang and Mohammad Al Hasan. 2022. Robust Node Classification on Graphs: Jointly from Bayesian Label Transition and Topology-based Label Propagation. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*. 2795–2805.

[58] Jun Zhuang and Mohammad Al Hasan. 2022. How Does Bayesian Noisy Self-Supervision Defend Graph Convolutional Networks? *Neural Processing Letters* (2022), 1–22.

[59] Jun Zhuang and Mohammad Al Hasan. 2023. Robust Node Representation Learning via Graph Variational Diffusion Networks. *arXiv preprint arXiv:2312.10903* (2023).

[60] Henry Zou and Cornelia Caragea. 2023. JointMatch: A Unified Approach for Diverse and Collaborative Pseudo-Labeling to Semi-Supervised Text Classification. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*. 7290–7301.

[61] Henry Zou, Yue Zhou, Weizhi Zhang, and Cornelia Caragea. 2023. DeCrisisMB: Debiased Semi-Supervised Learning for Crisis Tweet Classification via Memory Bank. In *Findings of the Association for Computational Linguistics: EMNLP 2023*. 6104–6115.

[62] Henry Peng Zou, Yue Zhou, Cornelia Caragea, and Doina Caragea. 2023. Crisismatch: Semi-supervised few-shot learning for fine-grained disaster tweet classification. *arXiv preprint arXiv:2310.14627* (2023).