electronics

Article

Multi-Channel Graph Convolutional Networks for Graphs with Inconsistent Structures and Features

Xinglong Chang [1,2], Jianrong Wang [1,3], Rui Wang [3], Tao Wang [3], Yingkui Wang [4,*] and Weihao Li [5]

[1] School of New Media and Communication, Tianjin University, Tianjin 300350, China; changxinglong@tju.edu.cn (X.C.); wjr@tju.edu.cn (J.W.)
[2] Qijia Youdao Network Technology (Beijing) Co., Ltd., Beijing 100012, China
[3] College of Intelligence and Computing, Tianjin University, Tianjin 300350, China; wr1895@tju.edu.cn (R.W.); wt2019216113@tju.edu.cn (T.W.)
[4] Department of Computer Science and Technology, Tianjin Renai College, Tianjin 301636, China
[5] Data61-CSIRO, Black Mountain Laboratories, Canberra, ACT 2601, Australia; weihao.li@data61.csiro.au
* Correspondence: ykwang@tju.edu.cn

Abstract: Graph convolutional networks (GCNs) have attracted increasing attention in various fields due to their significant capacity to process graph-structured data. Typically, the GCN model and its variants heavily rely on the transmission of node features across the graph structure, which implicitly assumes that the graph structure and node features are consistent, i.e., they carry related information. However, in many real-world networks, node features may unexpectedly mismatch with the structural information. Existing GCNs fail to generalize to inconsistent scenarios and are even outperformed by models that ignore the graph structure or node features. To address this problem, we investigate how to extract representations from both the graph structure and node features. Consequently, we propose the multi-channel graph convolutional network (MCGCN) for graphs with inconsistent structures and features. Specifically, the MCGCN encodes the graph structure and node features using two specific convolution channels to extract two separate specific representations. Additionally, two joint convolution channels are constructed to extract the common information shared by the graph structure and node features. Finally, an attention mechanism is utilized to adaptively learn the importance weights of these channels under the guidance of the node classification task. In this way, our model can handle both consistent and inconsistent scenarios. Extensive experiments on both synthetic and real-world datasets for node classification and recommendation tasks show that our methods, MCGCN-A and MCGCN-I, achieve the best performance on seven out of eight datasets and the second-best performance on the remaining dataset. For simpler graph structures or tasks where the overhead of multiple convolution channels is not justified, traditional single-channel GCN models might be more efficient.

Keywords: graph convolutional networks; network analysis; graph representation learning

check for updates

Citation: Chang, X.; Wang, J.; Wang, R.; Wang, T.; Wang, Y.; Li, W. Multi-Channel Graph Convolutional Networks for Graphs with Inconsistent Structures and Features. *Electronics* **2024**, *13*, 607. https://doi.org/10.3390/electronics13030607

Academic Editor: Christos J. Bouras

Received: 1 January 2024
Revised: 29 January 2024
Accepted: 30 January 2024
Published: 1 February 2024

1. Introduction

Network data are ubiquitous in the real world, such as social networks [1,2], citation networks [3,4], communication networks [5,6], and biological networks [7,8]. Such data often comprise two sources of information: the underlying network structure and the node features. For example, in a citation network, each node symbolizes a paper, characterized by keywords, whereas the edges denote citation links between papers. Consequently, an effective graph learning algorithm should seamlessly integrate both types of information [9,10].

Graph convolutional networks (GCNs) [11] are extensively studied for their robust representation learning capabilities and have been applied to various network analysis tasks, such as node classification, link prediction, community detection, and recommendation systems [12–14]. Typically, GCNs employ a feature-passing mechanism over the structure of the input graph, integrating information from both the network structure and node features. Specifically, each layer of the GCN updates node representations by aggregating those of their neighboring nodes, which is supervised by partially available node labels.

Recently, a variety of GCN layers have been proposed. For example, GraphSAGE [15] samples a fixed size of neighborhoods for each node, proposes a sum/max/LSTM pooling aggregator, and adopts a concatenation operation for the update. GAT [16] applies an attention mechanism to assign weights to neighbors. GIN [17] generates a provably maximally powerful graph neural network under the Weisfeiler–Lehman test [18].

Existing GCN-based methods [11,15,16,19] suffer from serious drawbacks. To fuse structural information and node features, they heavily rely on the process of passing features across the graph structure, which is roughly equivalent to Laplacian smoothing on node features [20]. Essentially, they assume consistency between structural and feature information, both contributing positively to node label prediction. However, in numerous real-world networks, these two sources of information often possess distinct characteristics, leading to a prevalent mismatch between them [21].

Such a mismatch (inconsistency) between structures and features can arise from two scenarios: (1) Either the topological structure or the node features frequently carry substantial noise, which can negatively impact node predictions. Using social networks like Twitter as an example, the real social relationship reflects the community structure more directly than the diverse and noisy user-generated features. (2) Adversarial attacks can specifically target either the network structure or node features [22]. These attacks involve actions like adding/removing edges or injecting random noise into node attributes. Both scenarios result in a mismatch between these two information sources. When such a mismatch occurs, existing methods struggle to derive effective representations from both the graph structure and node features. These methods exhibit inferior performance compared to models that solely leverage either the network structure (e.g., DeepWalk [23]) or node features (e.g., MLP [24]), as supported by our motivating observations.

Recent methods, such as H2GCN [25], Geom-GCN [26], CPGNN [27], GGCN [28], and GPR-GNN [29], target graphs exhibiting varying degrees of heterophily or low homophily, where connected nodes may have differing class labels and features. However, these methods continue to presume that both high-order structures and node features are reliable and consistent. In situations where the graph structure is initially completely inconsistent with the node features (e.g., random graph structure), incorporating a high-order structure fails to capture useful representations for downstream tasks [30]. As a result, these models struggle to generalize in scenarios with inconsistent structures and features.

Therefore, our research focuses on devising methods to derive representations from both the graph structure and node features, particularly under circumstances where these elements exhibit inconsistencies. In this work, we propose a novel approach called the multi-channel graph convolutional network (MCGCN) for graph data with inconsistent structures and node features. The core idea is to extract distinct representations from both the structural aspects and node features of the graphs. Additionally, we aim to capture the shared insights arising from combinations of structures and features. Meanwhile, we utilize an attention mechanism to adaptively incorporate these representations under the guidance of semi-supervised node classification tasks.

Specifically, we first generate a feature structure from the original node features and extract topological features based on the inherent network topology. Subsequently, we propagate the original features across the feature structure and the topological features across the original structure, employing two distinct convolution channels to extract two specific representations of the features and structure, respectively. This approach aims to identify inconsistencies within the data.

Furthermore, we propagate the original features across the original structure and the topological features across the feature structure using two joint convolution channels, aiming to effectively fuse information in a consistent scenario.

Ultimately, we leverage an attention mechanism to adaptively learn the importance weights of these representations, extracting the most useful information for downstream tasks [31,32]. Our method excels at extracting useful representations that significantly contribute to node prediction, even in cases of mismatched network structures and node features.

To summarize, our work makes the following contributions:

- We study the mismatch (inconsistency) between structures and node features and present two motivating examples, highlighting the limitations of GCNs in fusing inconsistent structures and node features.
- We propose a multi-channel graph convolutional network for graphs characterized by inconsistent structures and features. Our method extracts representations from both the structure and feature spaces, along with their combinations, and adaptively fuses the most useful information from these representations through an attention mechanism.
- Extensive results on both synthetic and real-world datasets for node classification tasks show that the proposed method outperforms existing start-of-the-art methods on graphs with inconsistent structures and features and also delivers competitive performance on graphs with consistent structures and features.

2. Related Works

2.1. Graph Convolutional Networks

A graph convolutional network (GCN) [11] is a kind of graph neural network that aims to extend traditional deep learning methods to graph-structured data [33,34]. GCNs simplify spectral-based graph convolutional networks [35,36] by limiting the filtering operation on the first-order neighborhood. GCNs update the representation of the target node by combining its own representation and the aggregated representations from its direct neighborhood.

The field has seen a variety of methodological advancements aimed at improving the flexibility, scalability, and performance of GCNs. This includes the development of spatial-based approaches like GraphSAGE [15] and GIN [17], which allow for inductive learning on graphs [37–39]. Attention mechanisms were introduced into GCNs with the Graph Attention Network (GAT) [40], enabling the model to focus on important parts of the graph. Further, Rong et al. proposed the DropEdge technique to alleviate over-smoothing issues prevalent in deep GCNs [41].

2.2. Multi-Channel Graph Convolutional Networks

Drawing inspiration from multi-channel strategies in conventional CNNs, researchers have explored similar concepts in the realm of GCNs. These works involve using multiple types of convolutions or incorporating various aspects of graph data (like different types of relationships or features) into separate channels. One of the pioneering works in this area is the multi-channel spectral GCN [42–44], which extends spectral graph convolutions by incorporating multiple spectral filters. This method allows different channels to capture various frequency components of graph signals, leading to richer representations.

However, in scenarios where the graph structure is misaligned with node features, existing methods prove ineffective in capturing useful representations for subsequent tasks. Consequently, models operating under these conditions face challenges in generalizing effectively when confronted with discrepancies between structures and features.

In this work, we design two specific convolution channels to extract representations from the structure and feature spaces. Another two joint convolution channels are designed to fuse and capture the common information shared by both the structure and node features. Finally, an attention mechanism is employed to adaptively learn the importance weights of these representations. By adopting this approach, the problem of mismatch between network structure and features can be effectively resolved.

3. Preliminaries

We first present the notations and problem definition and then introduce graph convolutional networks (GCNs).

3.1. Problem Definition

Definition 1. ***Attributed Network***. *Given an undirected, unweighted, and attributed network, $G = (V, E, X)$, where $V = \{v_1, v_2, \ldots, v_n\}$ is a set of n nodes, $E = \{e_{ij}\} \subseteq V \times V$ is a set of edges, and $X \in \mathbb{R}^{n \times m}$ is a matrix representing node features, where m denotes the feature dimension. The i-th row of X corresponds to the feature of node v_i. The topological structure of G is depicted by an adjacency matrix $A = [a_{ij}] \in \mathbb{R}^{n \times n}$, where $a_{ij} = 1$ indicates a connection between nodes v_i and v_j, and $a_{ij} = 0$ otherwise.*

Definition 2. ***Semi-Supervised Node Classification Task***. *In the attributed network, $G = (V, E, X)$, each node is categorized into one of C classes. We have labels for a subset V_L of nodes, containing u nodes, where $u \ll n$. For each node $v_i \in V_L$, a label $y_i \in C$ is assigned. The objective of the node classification task is to predict the labels of the remaining nodes in $V \backslash V_L$.*

3.2. Notations of Graph Convolutional Networks

Given a network, $G = (V, E, X)$, where A is the adjacency matrix and D is the degree matrix with $D_{ii} = \sum_j A_{ij}$. By introducing self-loop and normalizing the adjacency matrix, we have $\tilde{A} = \tilde{D}^{(-1/2)} \tilde{A} \tilde{D}^{(-1/2)}$, where $\tilde{A} = A + I$ (identity matrix) and $\tilde{D}_{ii} = \sum_j \tilde{A}_{ij}$. Then, the classic two-layer GCN can be defined as:

$$Z = f(X, A) = \sigma(\tilde{A}\sigma(\tilde{A}XW^{(0)})W^{(1)}), \tag{1}$$

where $W^{(0)}$ and $W^{(1)}$ are learnable weight matrices, σ denotes non-linear activation functions such as ReLU, and Z is the final output representation for downstream tasks. Although GCNs show proficiency in various network analysis tasks such as node classification [12,45], they fail to extract useful and effective representations from networks with inconsistent structures and features, as discussed in this paper.

4. Motivating Observations

In this section, we generate two sets of synthetic graphs to evaluate the adaptability of classical GCNs in learning from features and topological structures, especially under different inconsistency levels. A graph with 900 nodes is generated, and these nodes are categorized into three groups, with each group containing 300 nodes. For nodes sharing the same category, node features are generated following a Gaussian distribution, whereas their topological structure is created using the Stochastic Block Model (SBM) [46]. As a result, node labels are correlated with both the node features and topological structure. The initial phase of our experiment involves a synthetic graph with ideal features and an ideal structure, a condition termed the "perfect setting". We then incrementally add inconsistency. The methodology used to create these synthetic graphs and the approach used to integrate the inconsistencies are detailed in Section 6.2.3. For a comparative study, we select three benchmark models: Multi-Layer Perceptron (MLP) [24] as a feature-based model, DeepWalk [23] for structure-based analysis, and a GCN, which is a standard graph neural network model that utilizes both the graph structure and feature data, assuming their dependability and association with node labels.

4.1. Setting One: Structure Inconsistency

In this part of the experiment, we maintain constant and ideal node features that are correlated with their labels but introduce inconsistency into the topological structure. We generate a series of 11 graphs with increasing levels of structure inconsistency. The process begins with a perfectly consistent structure (i.e., structure inconsistency is 0). In the final graph of the series, the structure inconsistency reaches its maximum (equal to 1), resulting in a completely random topological structure. The results are shown in Figure 1a. Our observations reveal that all models perform efficiently under ideal conditions. However, as structural inconsistency is introduced, the MLP's performance remains stable, given its independence from structural information. On the other hand, DeepWalk's performance significantly worsens as structural inconsistency increases, eventually equating to the accuracy of random assignments. This decline is attributed to DeepWalk's inability to extract meaningful insights from a random and label-unrelated topology. Similarly, the performance of the GCN also diminishes, underscoring its limitations in gleaning valuable information from unreliable topological structures.

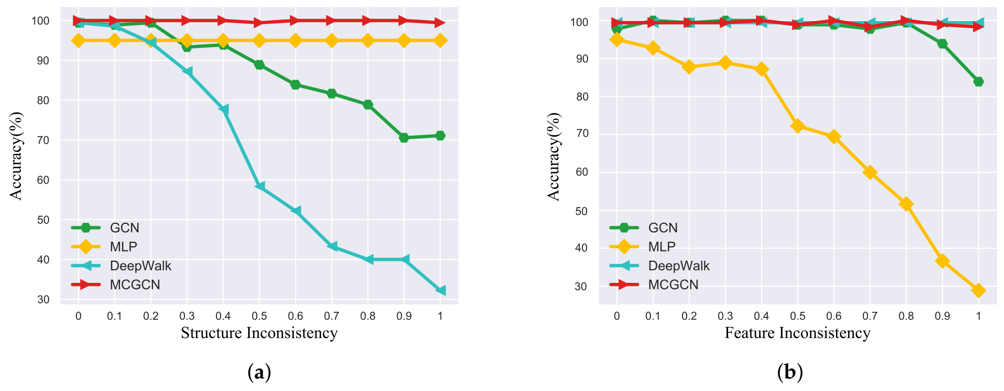

Figure 1. Classification accuracy on synthetic graphs. The x-axis represents the structure/feature inconsistency levels.

4.2. Setting Two: Feature Inconsistency

In this part, we maintain a constant and ideal structure, and feature inconsistency is added gradually. Similarly, 11 graphs with increasing feature inconsistency levels are generated. The series starts with an ideal situation where feature inconsistency is nonexistent (equal to 0). In the final graph of this sequence, feature inconsistency reaches its peak (equal to 1), characterized by all nodes sharing the same Gaussian distribution, thereby making the features uncorrelated with the node labels. The results are illustrated in Figure 1b. The results indicate that under ideal conditions, all models perform efficiently. DeepWalk demonstrates robust performance across various feature inconsistency levels. The effectiveness of the MLP diminishes swiftly, eventually aligning with the accuracy of random label assignment. The performance of the GCN also reduces as feature inconsistency increases, confirming its limitations in extracting valuable information from unreliable feature data.

4.3. Motivation

The outcomes of these experiments reveal a critical aspect of classical GCNs: their inherent method of combining information from both features and topological structures without discerning the reliability of each. This approach results in GCNs being less effective in extracting useful information from data where either the structure or features are inconsistent. Intriguingly, models that solely rely on either a topological structure or node features have been shown to outperform GCNs in such scenarios. This limitation of GCNs has been a key motivator in our research, leading us to develop and introduce specific and joint convolution methods as the primary contribution of this work.

5. Methodology

In this section, we provide a brief overview of our approach, followed by a detailed introduction of the main components of our model.

5.1. Overview

In order to enhance information fusion and extract optimal representations from inconsistent structures and features, we introduce the MCGCN, a multi-channel graph convolutional network. The framework of the proposed approach is illustrated in Figure 2. We separately process the network structure and features, deriving two distinct representations from the structure and feature spaces to effectively address inconsistency. Initially, feature structures are generated from the original node features, and structural features are initialized based on the network topology. Subsequently, the MCGCN propagates the original node features and structural features using two specific convolution channels over the features and the network structure, respectively. Simultaneously, the original and structural features are propagated through their respective structures via two joint convolution channels, efficiently extracting shared information and enabling generalization in both consistent and partially inconsistent scenarios. Lastly, recognizing the uncertain contribution of each representation to label prediction before training, we employ an attention mechanism to adaptively learn their importance weights. This approach adaptively ensures the extraction of pertinent information for the classification task, even in scenarios where the structure and features exhibit inconsistency.

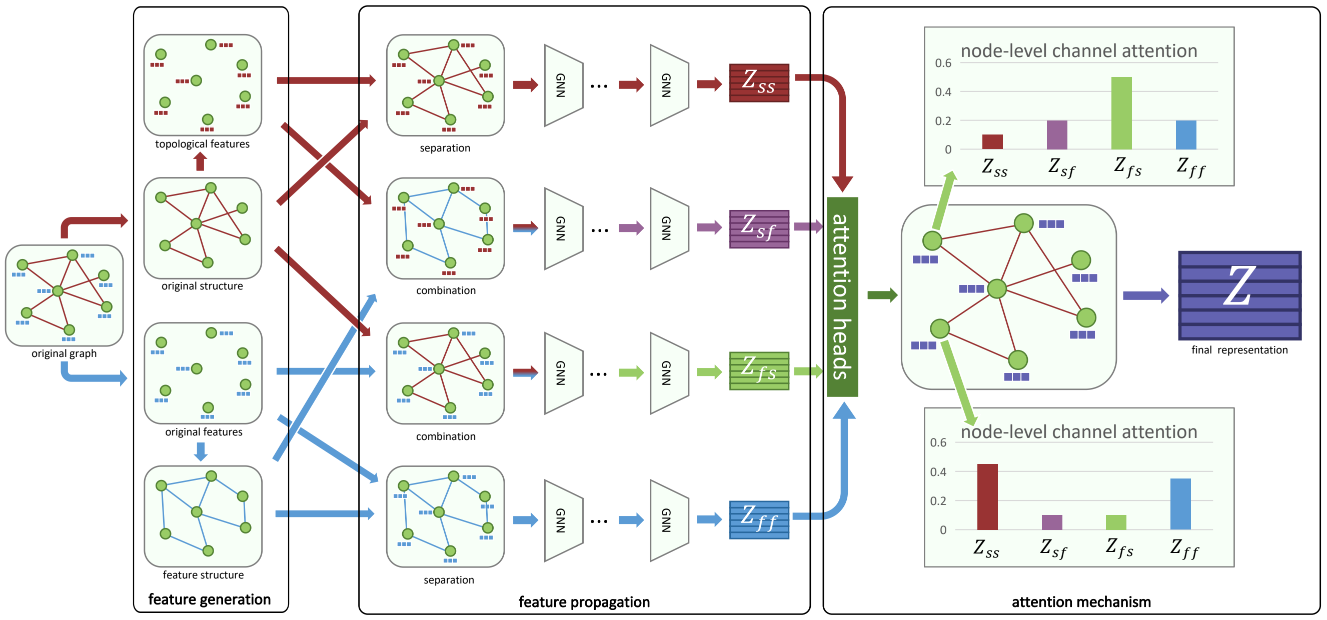

Figure 2. The framework of the MCGCN. We first generate topological features from the original structure and create a feature structure from the original features. Then, we design four convolution channels to capture different information: two specific convolution channels to separately capture feature and structure information, and two joint convolution channels to effectively extract the common information shared by the structure and features. Finally, we employ an attention mechanism to adaptively learn the importance weights of these representations.

5.2. Specific Convolution Channels

To enhance the ability to manage inconsistent information, we employ two specific convolution channels to extract distinct representations from both the structure and features. Initially, we use a traditional MLP to capture representations from node features. However, this method treats each node independently, disregarding any concealed relationships among them. To address this limitation and extract more potent feature representations that account for the hidden node relationships, we leverage a specific graph convolution channel. This channel allows us to grasp the underlying connections between nodes, leading to more effective representations in the feature space. In order to carry out convolution operations within the feature space, we first generate a feature structure based on the original node features. Although there exist various methods for creating a feature structure, in this context, we choose an intuitive approach, *k*-nearest neighbor (kNN), to construct the feature structure, denoted as $A_f \in \mathbb{R}^{n\times n}$.

Specifically, we first calculate the similarity matrix $S \in \mathbb{R}^{n \times n}$ using cosine similarity to assess the similarity between any two nodes.

$$S_{ij} = \frac{x_i x_j}{|x_i||x_j|}, \tag{2}$$

where $x_i \in \mathbb{R}^m$ represents the original features of node v_i. Based on the similarity matrix S, we choose the top k similar nodes for each node to establish edges and, finally, derive the adjacency matrix of the feature structure A_f. Then, with the feature structure A_f and original node features X, we use a two-layer graph convolutional network to extract specific representation from the feature space, each layer of which is expressed as:

$$Z_{ff}^{(l)} = \sigma(\tilde{D}_f^{-1/2} \tilde{A}_f \tilde{D}_f^{-1/2} Z_{ff}^{(l-1)} W_{ff}^{(l)}), \tag{3}$$

where $\tilde{A}_f = A_f + I_f$ (I_f is the identity matrix) and $\tilde{D}_f$ denotes the degree matrix of $\tilde{A}_f$ with $(\tilde{D}_f)_{ii} = \sum_j (\tilde{A}_f)_{ij}$. $W_{ff}^{(l)}$ denotes the weight matrix of the l-th layer. $Z_{ff}^{(0)} = X$. σ is an activation function. We use $\mathrm{Relu}(t) = \max(0, t)$ in the first layer and $\mathrm{linear}(t) = t$ in the second layer. In this way, we can obtain the final feature representation Z_{ff} ($Z_{ff} = Z_{ff}^{(2)}$) that captures the underlying structure of nodes in the feature space.

Next, to extract specific representations from the topological space, we employ another dedicated convolutional channel designed to capture the inherent relationships between nodes within the topology space. Specifically, we first generate the initial topological features from the original network structure. There are numerous methods available for generating these topological features. Notably, the network's adjacency matrix, denoted as A, can directly encapsulate the structural information of nodes. Therefore, we initially consider the original adjacency matrix A as the initial topological features X_s, i.e., $X_s = A$. Leveraging both the topological features and the original network structure, we subsequently employ a two-layer graph convolutional network (GCN) to distill structural representations from the structural space, each layer of which is expressed as:

$$Z_{ss}^{(l)} = \sigma(\tilde{D}^{-1/2} \tilde{A} \tilde{D}^{-1/2} Z_{ss}^{(l-1)} W_{ss}^{(l)}), \tag{4}$$

where $\tilde{A} = A + I$ and $\tilde{D}$ denotes the degree matrix of $\tilde{A}$. $W_{ss}^{(l)}$ denotes the weight matrix of the l-th layer. $Z_{ss}^{(0)} = X_s$. We also use Relu and $\mathrm{linear}(t) = t$ in the first and second layers, respectively. Moreover, as shown in (4), we can also set the initial structure features $Z_{ss}^{(0)} = X_s = I$, as used in SAT [47]. Finally, we obtain the final structure representation $Z_{ss}^{(2)}$ that captures the underlying relationship of nodes in the structure space.

With these two specific convolution channels, we can extract specific representations from both the feature and topology spaces. In this way, our model can adaptively choose useful information for graphs with inconsistent structures and features according to specific downstream tasks.

5.3. Joint Convolution Channels

Many real-world networks exhibit diverse degrees of alignment between their partial network structures and node features. To ensure that our model is capable of handling both consistent and inconsistent scenarios, we further design two joint convolutional channels. These channels effectively merge the network structure and node feature information, enhancing flexibility in classification tasks to identify more pertinent information.

Specifically, our method propagates the original features across the network structure and topological features through the feature structure using two joint convolution channels. It is important to highlight that this approach is distinct from traditional GCN-based methods that only propagate features across the network structure. The new joint convolution channels effectively fuse these two distinct types of information more efficiently.

First, based on the original network structure and node features, we use a two-layer graph convolutional network, each layer of which is expressed as:

$$Z_{fs}^{(l)} = \sigma(\tilde{D}^{-1/2}\tilde{A}\tilde{D}^{-1/2}Z_{fs}^{(l-1)}W_{fs}^{(l)}), \tag{5}$$

where $W_{fs}^{(l)}$ denotes the weight matrix of the l-th layer and $Z_{fs}^{(0)} = X$.

Second, based on the feature structure and topological features, we use a two-layer graph convolutional network, each layer of which is expressed as:

$$Z_{sf}^{(l)} = \sigma(\tilde{D}_f^{-1/2}\tilde{A}_f\tilde{D}_f^{-1/2}Z_{sf}^{(l-1)}W_{sf}^{(l)}), \tag{6}$$

where $W_{sf}^{(l)}$ denotes the weight matrix of the l-th layer and $Z_{sf}^{(0)} = X_s$. Finally, we obtain two final joint representations $Z_{fs}^{(2)}$ and ($Z_{sf}^{(2)}$). With these two joint convolution channels, we can extract the common information shared by both the structure and features.

5.4. Attention Mechanism

Following the utilization of two distinct convolution channels and two combined convolution channels, we can successfully acquire the structural representation Z_{ss}, feature representation Z_{ff}, and two joint representations Z_{fs} and Z_{sf}. Nonetheless, predicting which representation correlates most with the classification task can be challenging, particularly in cases where the structure and features do not match. To address this, we employ an attention mechanism, which adaptively combines these representations using varying degrees of importance weights, enabling the automatic extraction of valuable insights. Their importance weights $(\alpha_{ss}, \alpha_{ff}, \alpha_{fs}, \alpha_{sf})$ are defined as follows:

$$(\alpha_{ss}, \alpha_{ff}, \alpha_{fs}, \alpha_{sf}) = \text{attention}(Z_{ss}, Z_{ff}, Z_{fs}, Z_{sf}), \tag{7}$$

where $\alpha_{ss}, \alpha_{ff}, \alpha_{fs}, \alpha_{sf} \in \mathbb{R}^{n\times 1}$ represent the attention weights of the representations Z_{ss}, Z_{ff}, Z_{fs}, $Z_{sf} \in \mathbb{R}^{n\times d}$, respectively. Specifically, we take the structure representation of node v_i, $z_{ss}^i \in \mathbb{R}^{1\times d}$ as an example. To obtain the attention value of the structure representation, we first transform the representation through linear transformation, followed by a nonlinear activation function tanh. Then, we apply a shared attention vector $q \in \mathbb{R}^{d'\times 1}$ to obtain the attention value as follows:

$$\beta_{ss}^i = q^T \cdot \tanh(W_a \cdot (z_{ss}^i)^T + b_1), \tag{8}$$

where $W_a \in \mathbb{R}^{d'\times d}$ represents the learnable weight matrix and $b_1 \in \mathbb{R}^{d'\times 1}$ denotes the bias vector. Similarly, we can obtain the attention values of $\beta_{ff}^i, \beta_{fs}^i, \beta_{sf}^i$. We can obtain the final importance weights by normalizing the attention values using a softmax function [48]:

$$\alpha_{ss}^i = \text{softmax}(\beta_{ss}^i) = \frac{\exp(\beta_{ss}^i)}{\exp(\beta_{ss}^i + \beta_{ff}^i + \beta_{fs}^i + \beta_{sf}^i)}. \tag{9}$$

The above equation implies that a larger α_{ss}^i value increases the importance of the structure representation. Similarly, we have:

$$\begin{aligned} \alpha_{ff}^i &= \text{softmax}(\beta_{ff}^i) \\ \alpha_{fs}^i &= \text{softmax}(\beta_{fs}^i) \\ \alpha_{sf}^i &= \text{softmax}(\beta_{sf}^i). \end{aligned} \tag{10}$$

Based on the importance weights, we can obtain the final representation of node v_i as follows:

$$z^i = \alpha^i_{ss} z^i_{ss} + \alpha^i_{ff} z^i_{ff} + \alpha^i_{fs} z^i_{fs} + \alpha^i_{sf} z^i_{sf}. \tag{11}$$

The final representation for all n nodes is expressed in matrix form as follows:

$$Z = \alpha_{SS} Z_{ss} + \alpha_{FF} Z_{ff} + \alpha_{FS} Z_{fs} + \alpha_{SF} Z_{sf}, \tag{12}$$

where $\alpha_{SS} = diag(\alpha_{ss})$, $\alpha_{FF} = diag(\alpha_{ff})$, $\alpha_{FS} = diag(\alpha_{fs})$ and $\alpha_{SF} = diag(\alpha_{sf})$.

5.5. Optimization Objective

Based on the final representation Z obtained using (12), we then apply a linear transformation and softmax function to obtain the predicted soft label assignment matrix $\hat{Y} \in \mathbb{R}^{n \times C}$ as follows:

$$\hat{Y} = \mathrm{softmax}(W_c Z + b_2), \tag{13}$$

where $\hat{Y}_{ic}$ represents the probability of node v_i belonging to class c. Then, for the semi-supervised node classification task, we can minimize the cross-entropy over all labeled nodes between the ground truth and the prediction:

$$L = \sum_{l \in V_L} Y_l \ln \hat{Y}_l, \tag{14}$$

where Y represents the label indicator matrix, i.e., $Y_{ic} = 1$ if node v_i belongs to class c, or $Y_{ic} = 0$ otherwise.

6. Experiments

In this section, we first give the experimental settings and then compare our proposed approach (MCGCN) with some state-of-the-art methods on two network analysis tasks, including transductive node classification and visualization. Next, we give attention to the analysis of synthetic datasets to validate whether the MCGCN can learn interpretable importance weights.

6.1. Experimental Setting

6.1.1. Datasets

Eight publicly available real-world datasets were utilized for evaluation. The Cornell, Texas, and Wisconsin datasets consist of web pages as nodes, hyperlinks as edges, and page categories (student, project, course, staff, and faculty) as node labels. These datasets were obtained from the respective web pages of Cornell University, the University of Texas, and the University of Wisconsin. The textual content of the web pages is typically processed and converted into feature vectors, which serve as input attributes for each node in the learning process. They are benchmarks for evaluating the performance of various graph-based machine learning algorithms, especially GCNs and other GNN models. The Film dataset is an actor co-occurrence network [49]. where nodes are actors, edges represent co-occurrence on the same Wikipedia page, and labels categorize actors based on their Wikipedia content. The Chameleon and Squirrel datasets are Wikipedia networks [50] with web pages as nodes, mutual links as edges, and labels categorized by the average monthly traffic of each page. The Cora and Citeseer datasets are citation networks [51,52], where nodes are academic papers, edges are citations, and node labels represent academic topics. Paper contents are processed as node features. Table 1 summarizes the statistics of these datasets.

Table 1. The statistics of the datasets. # denotes the number of objects.

Dataset	Texas	Wisconsin	Cornell	Squirrel	Chameleon	Film	Cora	Citeseer
# Nodes	183	251	183	5201	2277	7600	2708	3327
#Edges	309	499	295	217,073	36,101	33,544	5429	4732
#Features	1703	1703	1703	2089	2325	931	1433	3703
#Classes	5	5	5	5	5	5	7	6

6.1.2. Baselines

We compared two variants of the MCGCN with six state-of-the-art methods (DeepWalk, MLP, GCN, GAT, H2GCN [25], GPRGNN [29]): (1) MCGCN-A, which initializes topological features with an adjacency matrix, and (2) MCGCN-I, which initializes topological features with an identity matrix.

6.1.3. Parameter Setting

Following the Geom-GCN approach [26], for each dataset, we performed ten random separations, allocating 38%, 52%, and 10% of the total number of nodes for training, validation, and testing, respectively. All the parameters of the baselines were initialized as suggested in their respective papers. In our methods, MCGCN-A and MCGCN-I, we employed two-layer GCNs across four convolution channels. For Citeseer, we maintained a consistent hidden layer dimension of 512 and an output layer dimension of 32. ReLU was used as the activation function for all GCN modules. For the feature structure, we constructed a kNN graph using $k \in \{2, \ldots, 9\}$. Our models were developed using PyTorch deep learning tools and optimized with the Adam optimizer. The learning rate was set to 0.001, and we assessed model performance using classification accuracy.

6.2. Results and Analysis

6.2.1. Node Classification

Table 2 demonstrates that our methods, MCGCN-A and MCGCN-I, achieved the best performance on seven out of the eight datasets and the second-best performance on the remaining dataset.

Table 2. Mean accuracies for transductive node classification task (%). The best result is indicated in bold and the second-best result is underlined.

	Texas	Wisconsin	Cornell	Squirrel	Chameleon	Film	Cora	Citeseer
DeepWalk	49.19 ± 0.38	53.51 ± 1.10	44.12 ± 0.52	32.37 ± 0.95	42.61 ± 0.42	23.74 ± 0.56	76.08 ± 0.63	53.59 ± 0.63
MLP	77.30 ± 0.55	**83.01 ± 1.02**	77.98 ± 0.83	34.39 ± 0.43	45.47 ± 0.37	32.78 ± 0.52	72.30 ± 0.88	70.17 ± 0.62
GCN	52.16 ± 1.04	55.88 ± 0.97	52.70 ± 0.71	37.96 ± 1.13	60.03 ± 0.74	27.92 ± 0.51	85.21 ± 0.53	73.68 ± 0.47
GAT	58.38 ± 0.48	54.41 ± 0.94	54.32 ± 0.38	30.03 ± 1.28	59.93 ± 0.69	28.15 ± 0.92	85.34 ± 0.73	73.92 ± 0.43
H2GCN	77.57 ± 0.87	81.72 ± 0.74	77.81 ± 0.69	40.14 ± 0.47	59.64 ± 1.02	31.63 ± 0.49	85.27 ± 0.32	$\underline{74.42 \pm 0.49}$
GPRGNN	77.83 ± 0.43	81.96 ± 0.96	77.93 ± 0.98	41.81 ± 0.89	$\underline{60.09 \pm 0.73}$	$\underline{33.25 \pm 0.47}$	85.79 ± 0.36	73.37 ± 0.90
MCGCN-A	**78.46 ± 0.39**	82.39 ± 0.91	**78.65 ± 0.59**	$\underline{41.74 \pm 1.28}$	59.91 ± 0.47	**33.46 ± 0.62**	**86.32 ± 0.60**	74.04 ± 0.67
MCGCN-I	$\underline{78.39 \pm 0.47}$	$\underline{82.55 \pm 0.42}$	$\underline{78.21 \pm 0.38}$	**42.21 ± 1.79**	**61.64 ± 0.35**	33.17 ± 0.58	$\underline{86.28 \pm 0.74}$	**74.51 ± 0.34**

Our models demonstrated a noticeable improvement over the standard DeepWalk, GCN, and GAT across all datasets. This is primarily due to the MCGCN's multi-channel approach, which allows it to better capture the complex interplay between node features and the graph structure. DeepWalk and GCN, while robust in simpler graph environments, fell short in scenarios where the graph structure was inconsistent or features were complex. Compared to GAT, the MCGCN exhibited superior performance, particularly in environments with inconsistent graph structures. The MCGCN outperformed GAT, which employs attention mechanisms at the node level. This suggests that the MCGCN's method for handling inconsistency through multiple channels is more effective than the attention-based strategies used in GAT. The H2GCN achieved the second-best result on the Citeseer dataset. It addressed the limitations of GNNs in handling heterophily or low-homophily settings. However, the real-world complexity of heterophily might not have been fully captured in the models and datasets used. GPRGNN, which utilizes a novel GPRGNN architecture that adaptively learns GPR weights, achieved the second-best result on both the Chameleon and Film datasets. This adaptability is crucial for optimizing the use of both node features and topological information, regardless of the extent to which node labels exhibit homophily or heterophily.

The MCGCN's strength lies in its ability to handle inconsistency in both graph structures and node features, making it particularly suitable for complex networks found in domains like social network analysis or bioinformatics. However, its complex architecture may not yield significant improvements in more uniform or consistent graph datasets. In such scenarios, the additional computational complexity of the MCGCN might not translate into proportionate performance gains, especially when compared to simpler models like the GCN or GAT, which are optimized for such environments.

In conclusion, the MCGCN represents a significant advancement in graph convolutional networks, especially for challenging datasets with structural and feature inconsistencies. Its multi-channel approach offers a novel and effective solution for these complex scenarios, though its complexity may not always be necessary for more straightforward graph analysis tasks.

6.2.2. Visualization

To demonstrate our approach's superior performance, we employed t-SNE [53], which can project the learned node representations into a two-dimensional space, to visualize the representations in the Cora dataset as an example. Figure 3 displays the visualization results for the MLP, DeepWalk, GCN, and MCGCN-A, with each color denoting a categorical label. As shown, the results for the MLP, DeepWalk, and GCN are less satisfactory, since the borders between different segmented groups are unclear, and some points of different classes are mixed with each other. In contrast, the MCGCN forms more discernible clusters.

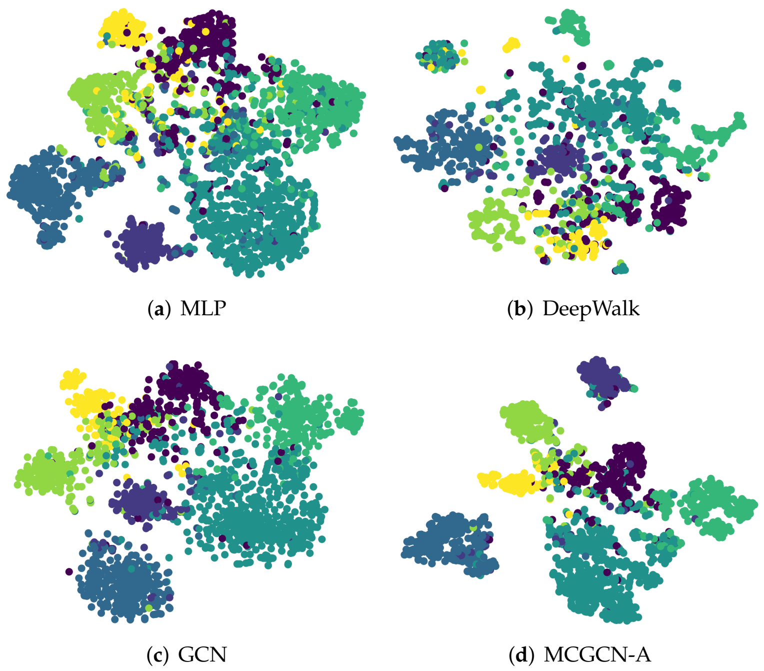

Figure 3. Visualization results on the Cora dataset. Each color denotes a categorical label (i.e., academic topic).

6.2.3. Synthetic Experiments

In this section, we elaborate on the generation of the synthetic datasets described in Section 4, which are utilized in the subsequent attention analysis.

Each synthetic graph comprised 900 nodes, and each node had a 50-dimensional feature vector. These 900 nodes were divided into three groups of 300, with each group receiving a distinct label. Although the three Gaussian distributions for these groups shared a common covariance matrix, their centers were initially distanced from each other. The topological structure was created using the SBM model [46], where each group was treated as a community. Intra-community edges were formed with a probability of 0.03, while initially, there were no inter-community edges. The node labels correlated with both the node features and the topological structure. Starting from this ideal scenario, we gradually added inconsistency.

In scenarios with structural inconsistencies, we maintained perfect and fixed node features while setting the intra-community edge probability at 0.03. We began with a synthetic graph with a 0 probability for inter-community edges. Inconsistency was introduced by incrementally increasing the inter-community edge probability by 0.003 for each new graph. In total, eleven graphs were generated, culminating in a final graph where the inter-community edge probability reached 0.03, matching the intra-community probability.

In settings with feature inconsistencies, we began with an ideal scenario, where three distinct node groups were formed using three Gaussian distributions with centers distanced from each other. We assumed that three randomly generated Gaussian distribution centers were f_1, f_2, and f_3, with a mean of $f_{mean} = (f_1 + f_2 + f_3)/3$. We generated eleven graphs, each with centers defined as $f_k[i] = i \times f_k + (1 - i) \times f_{mean}$, where $k = 1, 2, 3$. When $i = 1$, the three centers were in their original positions, distant from each other. We then decreased i by 0.1 for each subsequent graph, causing each center to gradually shift toward f_{mean}. In the final graph of the series, with $i = 0$, all three Gaussian distributions converged at the same center.

The results, presented in Figure 1, indicate that the MCGCN consistently performed well in both groups of settings. This demonstrates the MCGCN's effectiveness across various inconsistency levels, aligning with our expectations.

6.2.4. Attention Analysis

To further validate the MCGCN's ability to yield interpretable results, we analyzed the distributions and trends of its attention values. Given the lack of a definitive measure for inconsistency in real-world datasets, we analyzed the MCGCN's attention values on the synthetic datasets described earlier. This analysis helps illustrate how attention values shift with varying degrees of inconsistency in structures or features. Figure 4 shows the trends of the attention values on four representations (i.e., structure representation, feature representation, and two joint representations). Initially, when both the features and structure are informative, attention is distributed uniformly across all representations, as depicted in Figure 4a. However, as structural inconsistency intensifies, particularly approaching 1, the attention predominantly shifts toward feature representation. The results validate that the MCGCN can extract information that is correlated with the classification task and filter the redundant information contained in an unreliable structure. A similar pattern is observed in Figure 4b, where increasing feature inconsistency leads to a gradual shift of attention toward structure representation. These trends, illustrated in Figure 4, indicate that the MCGCN can adaptively learn the effective representation from an inconsistent structure and features and further demonstrate that our proposed MCGCN can obtain more interpretable results.

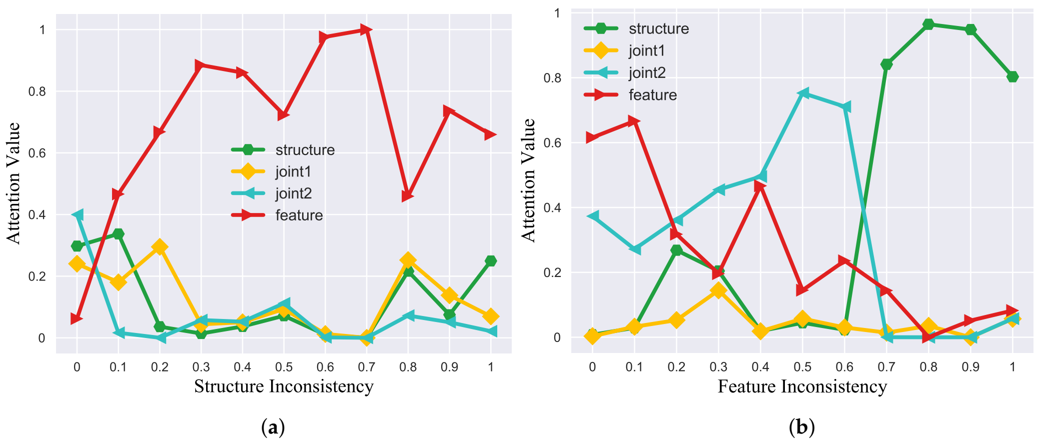

Figure 4. Attention analysis of the MCGCN on synthetic datasets. The x-axis represents the feature/structure inconsistency levels.

6.3. *Case Study on Recommendation Task*

6.3.1. Datasets

To evaluate the performance of the MCGCN in the context of recommendations, we constructed a dataset focused on person–job fit, sourced from the renowned human resources platform Xinrenxinshi.com, as outlined in Table 3. The dataset allocation included 80% for training, with the remaining 20% equally divided between validation and testing, each constituting 10%. BERT was utilized to initialize the representations for each person or job node, deriving these from their respective textual descriptions.

Table 3. Statistics of the Xinrenxinshi dataset. # denotes the number of objects.

#Person	#Job	#Interaction	#Recommend	#Interview	#Offer
9719	2035	15,101	1735	952	159

6.3.2. Baselines

For the job recommendation task, we employed GCN, GAT, AGC [54], and HAN [55] as baseline methods. These models are recognized as state of the art in this domain. Notably, HAN introduces a unique technique through its use of hierarchical attention networks. This method enables a deeper comprehension of the inherent hierarchical structures present in documents.

6.3.3. Metrics

The AUC (Area Under the Receiver Operating Characteristic Curve) metric is of paramount importance in assessing the performance of recommendation tasks. Furthermore, we employed four other widely acknowledged metrics—accuracy, recall, and F1 score—to comprehensively evaluate effectiveness.

6.3.4. Results and Analysis

Table 4 displays the job recommendation results on the Xinrenxinshi dataset, with our method exhibiting outstanding performance. Our approach surpassed all competing methods in terms of accuracy and F1 score while achieving the second-best results in terms of the AUC. In terms of recall, AGC and GAT marginally outperformed our model, with differences of 3.79% and 0.45%. Overall, as illustrated in Table 4, our model demonstrated either superior or comparable effectiveness in relation to other baseline models across most of the evaluated metrics.

Table 4. A comparison of the job recommendation results on the Xinrenxinshi dataset. The best result is indicated in bold and the second-best result is underlined.

Method	AUC	Accuracy	Recall	F1 Score
GCN	0.7789	<u>0.7347</u>	0.9155	0.7753
GAT	0.8176	0.7300	**0.9765**	<u>0.7834</u>
AGC	**0.8909**	0.6596	<u>0.9437</u>	0.7349
HAN	0.6465	0.5266	0.7230	0.5693
Ours	<u>0.8448</u>	**0.7483**	0.9395	**0.7891**

7. Conclusions and Future Work

In this paper, we propose the MCGCN, a novel multi-channel graph convolutional network for graphs with inconsistent structures and features. The proposed approach includes four main convolution channels and an attention mechanism. To handle inconsistent information, two specific convolution channels are designed to extract representations from the structure and feature spaces. Additionally, two joint convolution channels are developed to fuse and capture the common information shared by both the structure and node features, enabling the handling of both consistent and inconsistent scenarios. An attention mechanism is employed to adaptively learn the importance weights of these representations, focusing on the information most relevant to node label prediction. Comprehensive experiments on synthetic and real-world datasets demonstrate the proposed approach's superior performance compared to several existing state-of-the-art methods.

The MCGCN is particularly well suited for applications involving complex network structures with variable or inconsistent data, such as social networks, biological networks, and recommendation systems. Its ability to handle inconsistency and extract meaningful representations from complex graphs makes it a valuable tool in these areas. However, for simpler graph structures or tasks where the overhead of multiple convolution channels is not justified, traditional single-channel GCN models might be more efficient.

Incorporating the proposed approach into real-world applications demands careful consideration of several practical aspects. (1) Scalability: As the approach is designed for graph-based data, which can be extensive and complex in real-world scenarios, it is vital to ensure that the model scales efficiently. This might involve using distributed computing techniques to manage large datasets and optimizing the model's architecture to handle an increasing number of nodes and edges without significantly compromising performance. (2) Deployment challenges: Integrating this model into existing systems involves challenges such as ensuring compatibility with current infrastructures, minimal downtime, and maintaining data integrity and security during the integration process. Continuous monitoring and maintenance will be required to ensure the model's performance remains optimal over time. (3) Computational resources: The proposed approach requires significant computational resources for both training and inference. The training phase might require high-performance GPUs or even TPUs to manage the computational load, especially for larger datasets. For inference, especially in real-time applications, the computational demands can be substantial. It is crucial to balance the computational costs with the performance gains provided by the model. In summary, the successful implementation of the proposed approach in real-world applications requires a well-thought-out strategy that addresses scalability, deployment, and computational resources.

In future research, we plan to explore alternative approaches for initializing topological features and generating feature structures. Furthermore, we aim to enhance our method with a data-driven, self-supervised learning approach, making it applicable to graphs without available label information.

Author Contributions: Conceptualization, X.C. and Y.W.; methodology, J.W.; software (https://github.com/wyking36/AGC, accessed on 1 January 2024), R.W. and T.W.; validation, T.W. and W.L.; writing—original draft preparation, X.C.; writing—review and editing, J.W.; visualization, R.W.; supervision, W.L. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the National Natural Science Foundation of China (No. 62276187 and No. U2031142).

Data Availability Statement: The data presented in this study are available on request from the corresponding author.

Conflicts of Interest: Author Xinglong Chang was employed by the company Qijia Youdao Network Technology (Beijing) Co., Ltd. The remaining authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

References

1. Mitchell, J.C. Social networks. *Annu. Rev. Anthropol.* **1974**, *3*, 279–299. [CrossRef]
2. Milroy, L.; Llamas, C. Social networks. In *The Handbook of Language Variation and Change*; Wiley-Blackwell: Oxford, UK, 2013; pp. 407–427.
3. Radicchi, F.; Fortunato, S.; Vespignani, A. Citation networks. In *Models of Science Dynamics: Encounters between Complexity Theory and Information Sciences*; Springer: Berlin/Heidelberg, Germany, 2011; pp. 233–257.
4. Greenberg, S.A. How citation distortions create unfounded authority: Analysis of a citation network. *BMJ* **2009**, *339*, b2680. [CrossRef] [PubMed]
5. Shaw, M.E. Communication networks. In *Advances in Experimental Social Psychology*; Elsevier: Amsterdam, The Netherlands, 1964; Volume 1, pp. 111–147.
6. Monge, P.R.; Contractor, N.S. *Theories of Communication Networks*; Oxford University Press: New York, NY, USA, 2003.
7. Alm, E.; Arkin, A.P. Biological networks. *Curr. Opin. Struct. Biol.* **2003**, *13*, 193–202. [CrossRef] [PubMed]
8. Girvan, M.; Newman, M.E. Community structure in social and biological networks. *Proc. Natl. Acad. Sci. USA* **2002**, *99*, 7821–7826. [CrossRef] [PubMed]
9. Grover, A.; Leskovec, J. node2vec: Scalable feature learning for networks. In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, 13–17 August 2016; pp. 855–864.
10. Chen, Y.; Wu, L.; Zaki, M. Iterative deep graph learning for graph neural networks: Better and robust node embeddings. *Adv. Neural Inf. Process. Syst.* **2020**, *33*, 19314–19326.
11. Kipf, T.N.; Welling, M. Semi-supervised classification with graph convolutional networks. In Proceedings of the ICLR, Toulon, France, 24–26 April 2017.
12. Tang, J.; Aggarwal, C.C.; Liu, H. Node classification in signed social networks. In Proceedings of the ICDM, Barcelona, Spain, 12–15 December 2016; pp. 54–62.
13. Gao, S.; Denoyer, L.; Gallinari, P. Temporal link prediction by integrating content and structure information. In Proceedings of the CIKM, Scotland, UK, 24–28 October 2011; pp. 1169–1174.
14. Yu, X.; Han, J. Personalized entity recommendation: A heterogeneous information network approach. In Proceedings of the WSDM, New York City, NY, USA, 24–28 February 2014; pp. 283–292.
15. Hamilton, W.L.; Ying, Z.; Leskovec, J. Inductive representation learning on large graphs. In Proceedings of the NIPS, Long Beach, CA, USA, 12–24 December 2017; pp. 1024–1034.
16. Velickovic, P.; Cucurull, G.; Casanova, A.; Romero, A.; Liò, P.; Bengio, Y. Graph attention networks. In Proceedings of the ICLR, Vancouver, BC, Canada, 30 April–3 May 2018.
17. Xu, K.; Hu, W.; Leskovec, J.; Jegelka, S. How Powerful are Graph Neural Networks? In Proceedings of the ICLR, New Orleans, LA, USA, 6–9 May 2019.
18. Shervashidze, N.; Schweitzer, P.; van Leeuwen, E.J.; Mehlhorn, K.; Borgwardt, K.M. Weisfeiler-Lehman Graph Kernels. *J. Mach. Learn. Res.* **2011**, *12*, 2539–2561.
19. Wu, F.; Souza, A.; Zhang, T.; Fifty, C.; Yu, T.; Weinberger, K. Simplifying Graph Convolutional Networks. In Proceedings of the ICML, Long Beach, CA, USA, 10–15 June 2019; pp. 6861–6871.
20. Li, Q.; Han, Z.; Wu, X. Deeper Insights Into Graph Convolutional Networks for Semi-Supervised Learning. In Proceedings of the AAAI, New Orleans, LA, USA, 2–7 February 2018; pp. 3538–3545.
21. Qin, M.; Jin, D.; Lei, K.; Gabrys, B.; Musial-Gabrys, K. Adaptive community detection incorporating topology and content in social networks. *Knowl. Based Syst.* **2018**, *161*, 342–356. [CrossRef]
22. Zügner, D.; Akbarnejad, A.; Günnemann, S. Adversarial attacks on neural networks for graph data. In Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, New York, NY, USA, 19–23 August 2018; pp. 2847–2856.

23. Perozzi, B.; Al-Rfou, R.; Skiena, S. DeepWalk: Online learning of social representations. In Proceedings of the KDD, New York, NY, USA, 24–27 August 2014; pp. 701–710.
24. Pal, S.K.; Mitra, S. Multilayer perceptron, fuzzy sets, and classification. *IEEE Trans. Neural Netw.* **1992**, *3*, 683–697. [CrossRef]
25. Zhu, J.; Yan, Y.; Zhao, L.; Heimann, M.; Akoglu, L.; Koutra, D. Beyond Homophily in Graph Neural Networks: Current Limitations and Effective Designs. In Proceedings of the NIPS, Virtual-only Conference, 6–12 December 2020.
26. Pei, H.; Wei, B.; Chang, K.C.; Lei, Y.; Yang, B. Geom-GCN: Geometric Graph Convolutional Networks. In Proceedings of the ICLR, Virtual-only Conference, 26 April–1 May 2020.
27. Zhu, J.; Rossi, R.A.; Rao, A.B.; Mai, T.; Lipka, N.; Ahmed, N.K.; Koutra, D. Graph Neural Networks with Heterophily. In Proceedings of the AAAI, Virtual-only Conference, 2–9 February 2021.
28. Yan, Y.; Hashemi, M.; Swersky, K.; Yang, Y.; Koutra, D. Two Sides of the Same Coin: Heterophily and Oversmoothing in Graph Convolutional Neural Networks. *arXiv* **2021**, arXiv:2102.06462.
29. Chien, E.; Peng, J.; Li, P.; Milenkovic, O. Adaptive Universal Generalized PageRank Graph Neural Network. In Proceedings of the ICLR, Virtual-only Conference, 3–7 May 2021.
30. Wang, K.; Zhu, Y.; Liu, H.; Zang, T.; Wang, C. Learning Aspect-Aware High-Order Representations from Ratings and Reviews for Recommendation. *ACM Trans. Knowl. Discov. Data* **2023**, *17*, 1–22. [CrossRef]
31. Guo, M.H.; Xu, T.X.; Liu, J.J.; Liu, Z.N.; Jiang, P.T.; Mu, T.J.; Zhang, S.H.; Martin, R.R.; Cheng, M.M.; Hu, S.M. Attention mechanisms in computer vision: A survey. *Comput. Vis. Media* **2022**, *8*, 331–368. [CrossRef]
32. Niu, Z.; Zhong, G.; Yu, H. A review on the attention mechanism of deep learning. *Neurocomputing* **2021**, *452*, 48–62. [CrossRef]
33. Scarselli, F.; Gori, M.; Tsoi, A.C.; Hagenbuchner, M.; Monfardini, G. The Graph Neural Network Model. *IEEE Trans. Neural Netw.* **2009**, *20*, 61–80. [CrossRef]
34. Wu, Z.; Pan, S.; Chen, F.; Long, G.; Zhang, C.; Yu, P.S. A Comprehensive Survey on Graph Neural Networks. *IEEE Trans. Neural Netw. Learn. Syst.* **2021**, *32*, 4–24. [CrossRef]
35. Defferrard, M.; Bresson, X.; Vandergheynst, P. Convolutional Neural Networks on Graphs with Fast Localized Spectral Filtering. In Proceedings of the NIPS, Barcelona, Spain, 5–10 December 2016; pp. 3837–3845.
36. Shuman, D.I.; Narang, S.K.; Frossard, P.; Ortega, A.; Vandergheynst, P. The Emerging Field of Signal Processing on Graphs: Extending High-Dimensional Data Analysis to Networks and Other Irregular Domains. *IEEE Signal Process. Mag.* **2013**, *30*, 83–98. [CrossRef]
37. Ge, L.; Li, S.; Wang, Y.; Chang, F.; Wu, K. Global spatial-temporal graph convolutional network for urban traffic speed prediction. *Appl. Sci.* **2020**, *10*, 1509. [CrossRef]
38. Chen, Z.; Li, S.; Yang, B.; Li, Q.; Liu, H. Multi-scale spatial temporal graph convolutional network for skeleton-based action recognition. In Proceedings of the AAAI, Virtual-only Conference, 2–9 February 2021; Volume 35, pp. 1113–1122.
39. Shan, X.; Cao, J.; Huo, S.; Chen, L.; Sarrigiannis, P.G.; Zhao, Y. Spatial–temporal graph convolutional network for Alzheimer classification based on brain functional connectivity imaging of electroencephalogram. *Hum. Brain Mapp.* **2022**, *43*, 5194–5209. [CrossRef]
40. Veličković, P.; Cucurull, G.; Casanova, A.; Romero, A.; Lio, P.; Bengio, Y. Graph attention networks. *arXiv* **2017**, arXiv:1710.10903.
41. Rong, Y.; Huang, W.; Xu, T.; Huang, J. Dropedge: Towards deep graph convolutional networks on node classification. *arXiv* **2019**, arXiv:1907.10903.
42. Wang, X.; Zhu, M.; Bo, D.; Cui, P.; Shi, C.; Pei, J. Am-gcn: Adaptive multi-channel graph convolutional networks. In Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, New York, NY, USA, 6–10 July 2020; pp. 1243–1253.
43. Zhang, H.; Tian, Q.; Han, Y. Multi channel spectrum prediction algorithm based on GCN and LSTM. In Proceedings of the 2022 IEEE 96th Vehicular Technology Conference (VTC2022-Fall), London, UK, 26–29 September 2022; pp. 1–5.
44. Zhai, R.; Zhang, L.; Wang, Y.; Song, Y.; Yu, J. A multi-channel attention graph convolutional neural network for node classification. *J. Supercomput.* **2023**, *79*, 3561–3579. [CrossRef]
45. Hu, F.; Zhu, Y.; Wu, S.; Wang, L.; Tan, T. Hierarchical Graph Convolutional Networks for Semi-supervised Node Classification. In Proceedings of the IJCAI, Macao, China, 10–16 August 2019; pp. 4532–4539.
46. Karrer, B.; Newman, M.E.J. Stochastic blockmodels and community structure in networks. *Phys. Rev.* **2010**, *83*, 016107. [CrossRef] [PubMed]
47. Chen, X.; Chen, S.; Yao, J.; Zheng, H.; Zhang, Y.; Tsang, I.W. Learning on attribute-missing graphs. *IEEE Trans. Pattern Anal. Mach. Intell.* **2020**, *44*, 740–757. [CrossRef]
48. Liu, W.; Wen, Y.; Yu, Z.; Yang, M. Large-margin softmax loss for convolutional neural networks. In Proceedings of the 33rd International Conference on International Conference on Machine Learning, New York, NY, USA, 19–24 June 2016; Volume 48, pp. 507–516.
49. Tang, J.; Sun, J.; Wang, C.; Yang, Z. Social influence analysis in large-scale networks. In Proceedings of the KDD, Paris, France, 28 June–1 July 2009; pp. 807–816.
50. Rozemberczki, B.; Allen, C.; Sarkar, R. Multi-Scale attributed node embedding. *J. Complex Netw.* **2021**, *9*, cnab014. [CrossRef]
51. Sen, P.; Namata, G.; Bilgic, M.; Getoor, L.; Gallagher, B.; Eliassi-Rad, T. Collective Classification in Network Data. *AI Mag.* **2008**, *29*, 93–106. [CrossRef]

52. Namata, G.M.; London, B.; Getoor, L.; Huang, B. Query-driven Active Surveying for Collective Classification. In Proceedings of the Workshop on Mining and Learning with Graphs (MLG), Edinburgh, UK, 1 July 2012.
53. Hinton, G.E. Visualizing High-Dimensional Data Using t-SNE. *Vigiliae Christ.* **2008**, *9*, 2579–2605.
54. Chang, X.; Wang, J.; Guo, R.; Wang, Y.; Li, W. Asymmetric Graph Contrastive Learning. *Mathematics* **2023**, *11*, 4505. [CrossRef]
55. Yang, Z.; Yang, D.; Dyer, C.; He, X.; Smola, A.; Hovy, E. Hierarchical attention networks for document classification. In Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, San Diego, CA, USA, 12–17 June 2016; pp. 1480–1489.