applied sciences

MDPI

Article

Prompt-Based Graph Convolution Adversarial Meta-Learning for Few-Shot Text Classification

Ruwei Gong [1], Xizhong Qin [1,2,*] and Wensheng Ran [3]

[1] College of Information Science and Engineering, Xinjiang University, Urumqi 830049, China; 709919420@stu.xju.edu.cn
[2] Xinjiang Key Laboratory of Signal Detection and Processing, Urumqi 830049, China
[3] Xinjiang Uygur Autonomous Region Product Quality Supervision and Inspection Institute, Urumqi 830049, China; rws@xju.edu.cn
* Correspondence: qinxz@xju.edu.cn

Abstract: Deep learning techniques have demonstrated significant advancements in the task of text classification. Regrettably, the majority of these techniques necessitate a substantial corpus of annotated data to achieve optimal performance. Meta-learning has yielded intriguing outcomes in few-shot learning tasks, showcasing its potential in advancing the field. However, the current meta-learning methodologies are susceptible to overfitting due to the mismatch between a small number of samples and the complexity of the model. To mitigate this concern, we propose a Prompt-based Graph Convolutional Adversarial (PGCA) meta-learning framework, aiming to improve the adaptability of complex models in a few-shot scenario. Firstly, leveraging prompt learning, we generate embedding representations that bridge the downstream tasks. Then, we design a meta-knowledge extractor based on a graph convolutional neural network (GCN) to capture inter-class dependencies through instance-level interactions. We also integrate the adversarial network architecture into a meta-learning framework to extend sample diversity through adversarial training and improve the ability of the model to adapt to new tasks. Specifically, we mitigate the impact of extreme samples by introducing external knowledge to construct a list of class prototype extensions. Finally, we conduct a series of experiments on four public datasets to demonstrate the effectiveness of our proposed method.

Keywords: meta-learning; prompt learning; graph convolutional neural network; adversarial network

Citation: Gong, R.; Qin, X.; Ran, W. Prompt-Based Graph Convolution Adversarial Meta-Learning for Few-Shot Text Classification. *Appl. Sci.* **2023**, *13*, 9093. https://doi.org/10.3390/app13169093

Academic Editors: Duy-Tai Dinh and Uday Kiran Rage

Received: 2 June 2023
Revised: 31 July 2023
Accepted: 8 August 2023
Published: 9 August 2023

1. Introduction

The text classification problem refers to the task of determining the most suitable label from a given list of potential classes for a given unlabeled text. The task has been extensively studied and applied in many practical scenarios [1–5] such as social media, news websites, literature classification, etc. Deep-learning-based supervised text classification methods, including CNN, RNN, Transformers [6–9], and more, have demonstrated their superiority and achieved remarkable achievements.

But, in order to learn numerous model parameters, deep learning-based text categorization techniques need a lot of labelled data. In the real world, obtaining large-scale labeled data is time-consuming and expensive. Deep-learning-based algorithms have trouble learning the semantic space when there are few data available, which leads to subpar text categorization outcomes. As a result, few-shot text classification becomes a challenging task.

In recent times, prompt tuning [10,11] has emerged as a pragmatic strategy for addressing the challenges associated with few-shot learning. This process entails converting the original problem into a cloze test, thereby establishing a connection between pre-training and subsequent tasks in a manner that promotes cohesive integration. By doing so, prompt tuning addresses the overfitting challenge in few-shot learning by leveraging embeddings to transform samples into a more adaptable distribution space. This process encourages

the pre-training model to retrieve pertinent information from the pre-training phase that is specifically relevant to the target task. In general, the downstream task has to introduce additional parameters to adapt to the corresponding task needs. However, the downstream task model has difficulty learning the knowledge needed for the task quickly with a few samples. Meta-learning is a standard means of addressing this issue.

Through several meta-training tasks, meta-learning tries to enhance a model's capacity for learning, enabling the model to swiftly adapt to new tasks with a little amount of training data. Numerous meta-learning-based techniques [12–16] have proven to be superior in tackling challenges involving few-shot text categorization. However, the existing meta-learning methods suffer from the problem of overfitting caused by the mismatch of a few samples on the complex model. Ref. [17] first proposed the introduction of adversarial domain adaptive networks to enhance the generalization capability of the meta-learning framework.

Meanwhile, the majority of the few-shot text categorization techniques now in use [15] primarily emphasize information transmission between samples belonging to the same category, neglecting the relationships between different categories. Graph neural networks (GNNs) can effectively provide richer semantic representations by capturing the relationships between nodes through inter-instance information transfer and aggregation over graph structures, which is well suited for small-sample learning.

In this paper, we propose a Prompt-based Graph Convolutional Adversarial (PGCA) meta-learning network framework. First, we leverage prompt learning and pre-trained language models to obtain text embeddings adapted to downstream tasks. Then, we construct a GCN-based meta-knowledge extractor, allowing feature interactions among a few samples to capture the relationship information between nodes and fully extract the internal meta-knowledge of the text, generating higher-quality text embedding representations for the text classification task. In addition, we incorporate the adversarial network into the meta-learning framework, where the generator and discriminator are represented by GCN and feed-forward neural networks, respectively. We extend the text embedding feature space through adversarial training and enhance sample diversity. In addition, we introduce external knowledge to obtain a class prototype extension label list for each class. These class prototype representations are embedded into the feature space, considering label names and related word information, which we believe are more reliable. This approach effectively avoids extreme samples that may arise when obtaining class prototypes from support instances. The following are the primary contributions of our work:

- We propose a Prompt-based Graph Convolutional Adversarial (PGCA) meta-learning network framework to address the overfitting issue in few-shot text classification.
- We design a GCN-based meta-knowledge extractor to fully use the limited knowledge by obtaining node-relationship information through inter-instance interactions. We also integrate adversarial networks into the meta-learning framework to extend the sample space through adversarial training and improve the model's generalization ability.
- Our tests on four openly accessible datasets show that our strategy outperforms a number of other competing categorization strategies.

The remainder of the essay is structured as follows. The work on small sample text categorization is presented in Section 2. We go into great depth about our suggested PGCA approach in Section 3. The specifics of our experiments are described in Section 4, along with an analysis of the results. Section 5 draws the summary.

2. Related Work

2.1. Meta-Learning

Meta-learning is the process of improving a model's capacity for learning via training it on a variety of meta-training tasks, allowing it to swiftly pick up new information and adapt to new tasks with a little amount of practice. Existing meta-learning methods can be primarily classified into two categories:

(1) Optimization-based meta-learning methods aim to learn a well-initialized generalized model that allows the model to converge and adapt quickly to new tasks with only a few training samples. This type of meta-learning approach is typified by MAML [14], which learns a generic initialized model parameter that allows the model to converge to optimal performance with a small number of iterations of training when faced with a new task [12,18]. To enhance the ability of models to adapt to new tasks, ref. [17] first proposed using adversarial networks to enhance the adaptability of meta-learning architectures. Ref. [19] mitigated the overfitting of meta-learning through gradient similarity using an adaptive meta-learner.

(2) Metric-based meta-learning techniques seek to boost meta-learning performance by teaching participants a metric function or distance metric that may be used to compare the similarity of various tasks. Prototype networks [20] use the idea of clustering to project support sets into a metric space, obtain vector means based on the Euclidean distance metric as class prototypes, and calculate the distance to each prototype for test samples to achieve classification. However, the prototype network is susceptible to extreme samples. Ref. [21] introduced tagged word information to construct class prototypes, reducing the influence of extreme samples.

2.2. Graph Neural Network

Processing graph-structured data is carried out using Graph Neural Networks (GNN) [22,23]. GNNs have seen considerable application recently in a variety of disciplines, including physics [24,25] and computer vision [26,27].

Much of the work applying graph neural networks to the field of NLP has shown their superiority. Ref. [28] suggested a GNN model using message forwarding to spread labelled data from labelled examples to unlabeled query instances. TextGCN [29] built a single-text graph corpus and learned graph convolutional neural networks for text classification. Through a heterogeneous graph attention network, HGAT [30] provides a flexible heterogeneous information network to integrate extra information about classes and learn the significance of various neighboring nodes and various node kinds to the current node. By pre-training pair representations and aggregating data from nearby instance node representations, the Frog-GNN [31] presents a multi-view aggregation-based graph neural network may build graphs.

In order to capture information about the links between instances and provide a richer feature representation for the classification problem, this article interacts with data between instances using graph convolutional neural networks.

2.3. Prompt Tuning

The performance of prompt-based fine-tuning of previously trained language models has been demonstrated in few-shot learning. In order to aid language models in understanding a task, ref. [32] introduced pattern formation training, in which input examples are reconstructed as ideal fill-in-the-blank sentences. Ref. [33] proposed prefix adaptation to guide downstream language models by upstream prefixes [11]. Explored prompt fine-tuning, proposing efficient learning soft prompt mechanisms that adjust frozen language models to perform specific downstream tasks [21]. Prompt tuning and meta-learning combined, where base learners pick up task-specific tag words while meta-learners tweak task-neutral templates and encoders.

2.4. Inspiration

Inspired by the work of [21], we combined meta-learning with prompt fine-tuning to fine-tune the pre-trained language model in an external loop of meta-learning. We also focused on the relational information between different classes of samples, so we designed a GCN-based feature extractor to capture the relational information between instances using the inter-node interaction of the GCN, which we refer to as meta-knowledge. We were also inspired by [17] to integrate adversarial networks into the meta-learning framework

to extend the feature space of samples and enhance the ability of the model to adapt to new tasks.

3. Methodology

3.1. Problem Formulation

In this paper, we built a few-shot text classification task in the meta-learning framework [34]. Specifically, we gave labeled instances from a set of classes C_{train}. We aimed to learn a classification algorithm on C_{train} and then make predictions in new classes with only a few samples. These new classes belong to a set of classes C_{test} that do not intersect with C_{train}.

We took a sample of N-way K-shot tasks from C_{train} and assessed them similarly on C_{test} to replicate the few-shot scenario. We sampled N classes from C_{train} in the meta-training phase to create a meta-task. In meta-learning, each meta-task consists of a support set S and a query set Q. During the inner loop of meta-learning, we employed the support set to train the model parameters θ and the query set to test the model's performance. We sampled K examples for the support set S for each of these N classes and T instances for the query set Q, i.e., $S = \{(\boldsymbol{X}_i, \boldsymbol{y}_i)\}_{i=1}^{N \times K}$ and $Q = \left\{\left(\boldsymbol{X}_j, \boldsymbol{y}_j\right)\right\}_{j=1}^{N \times T}$. In our text classification model, the initial parameters are denoted as θ_0. A set of model parameters $\hat{\theta}$ is obtained in a meta-training task by feeding the model the support set S and optimising the internal loop. Subsequently, the parameters $\hat{\theta}$ of the model are employed on the query set Q, and the parameters θ_0 are updated by minimizing the loss incurred on the query set Q. During the meta-testing phase, we utilized a rigorous episodic-based approach to assess the model's ability to quickly adapt to new class C_{test}.

3.2. PGCA

The general structure of the PGCA network we propose is shown in Figure 1. Four major modules make up our model: an encoder with templates, a GCN-based meta-knowledge extractor, a domain discriminator, and a feature fusion module. Specifically, four main steps were performed. Firstly, leveraging prompt learning, we encoded sentences by adding templates and using a masked language model. Then, we designed a GCN-based meta-knowledge extractor to facilitate instance-level interactions and extract meta-knowledge. The domain discriminator and GCN were employed in an adversarial manner to expand the sample feature space and enhance sample diversity. Finally, we fused the encoder output with the output of the GCN-based meta-knowledge extractor to obtain a probability distribution and make label predictions.

3.2.1. Coder with Template

In previous work, given an instance $\boldsymbol{x} = \{w_1, \ldots, w_n\}$ with label y, our goal was to map $\boldsymbol{x}$ to the feature space through the language model and eventually compute $P(y|\boldsymbol{x})$. In contrast, in prompted learning, we entered samples into the pre-trained language model in a manner similar to the following:

$$\boldsymbol{x}_{prompt} = x + \boldsymbol{TEMPLATE} = \{[CLS], w_1, \ldots, w_n, [SEP], p_1, \ldots, [MASK], \ldots, p_t, [SEP]\} \quad (1)$$

where ***TEMPLATE*** is the template statement with the [MASK] token, noted as $\{[SEP], p_1, \ldots, [MASK], \ldots, p_t\}$. Using sentiment comment classification, first, the prompt is constructed: $\boldsymbol{TEMPLATE} = \{\ It\ is\ a\ [MASK]\ comment.\}$; then, the original text is input with the prompt spliced with $\boldsymbol{x}_{prompt}$ into a masked language model named M to predict [MASK] as the predicted answer representation. Such an operation converts our goal from computing $P(y|\boldsymbol{x})$ to computing $P\left([MASK] = \boldsymbol{w}_y \middle| \boldsymbol{x}_{prompt}\right)$, where $\boldsymbol{w}_y \in R^d$ is the embedding representation of the labeled word y. We use vectors in continuous space as cue templates, and although the choice of templates is manual, the embedding representation obtained in this way is learnable, and the language model can be fine-tuned by example.

We define the parameters of the encoder as $\boldsymbol{\theta}_M$ and the embedding of the prompt template as $\boldsymbol{\theta}_p$. The encoder can be formulated as follows:

$$\boldsymbol{X} = M\left(\boldsymbol{x}_{prompt}; \boldsymbol{\theta}_M, \boldsymbol{\theta}_p\right) \tag{2}$$

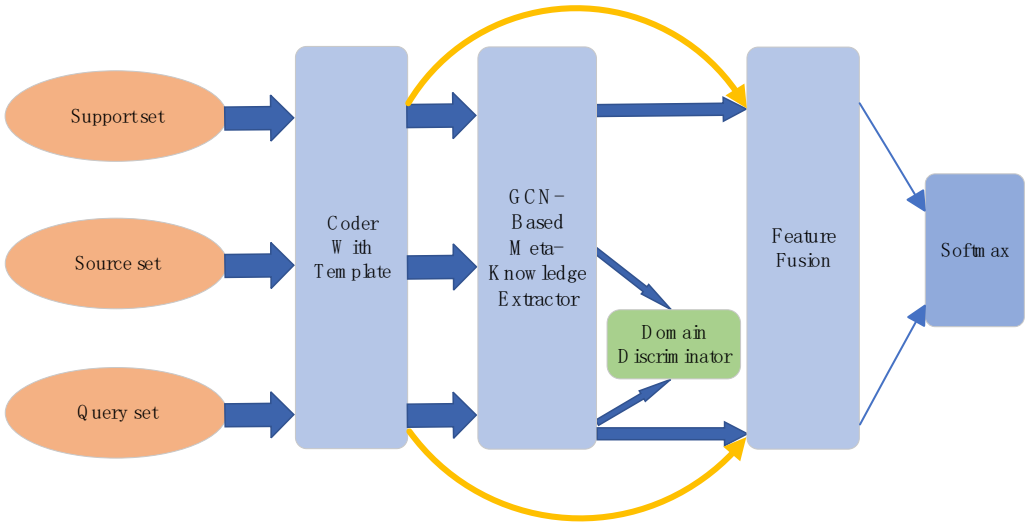

Figure 1. The overall PGCA model architecture. Source set is from the source domain, whereas Support set, Query set, and Source set are all from the target domain. The target and source domains do not cross each other. The Coder with Template module encodes instances using the mask language model and obtains label predictions at [MASK] as input for downstream tasks (Section 3.2.1); whether an instance is from the source domain or the target domain is determined by the Domain Discriminator module. (Section 3.2.2); the GCN-Based Meta-Knowledge Extractor module acquires meta-knowledge through inter-sentence interactions and expands the sample space by adversarially matching the Domain Discriminator to enhance sample diversity (Section 3.2.3); and the Feature Fusion module fuses the encoder output and the output of the GCN-Based Meta-Knowledge Extractor with features and performs classification scoring.

3.2.2. Domain Discrimination

Firstly, let us provide an introduction to the input of the domain discriminator. In the meta-training procedure, the target domain is the category that contains the samples for the goal task, and the source domain is the category that contains the remaining training data. In other words, a meta-task's support set and query set samples are drawn from the target domain, while the task's source set is drawn from a subset of the source domain that is drawn from the same size as the query set samples. Data from both the query set and the source set are fed into the domain discriminator, whose job it is to determine if the data come from the source domain or the target domain.

The discriminator is a three-layer feedforward neural network, and the probability distribution $P\left(y_p \mid \boldsymbol{x}\right)$ is computed by applying a softmax function in the output layer. We employ a cross-entropy loss function to determine the loss $Loss_{Domain}$, and the prediction labels of the discriminator are either 0 or 1, indicating that the samples come from the query set or the source set, respectively:

$$Loss_{Domain} = -\left[y_p log\hat{y} + (1 - y_p) log(1 - \hat{y})\right] \tag{3}$$

where y_p denotes the predicted label result and $\hat{y}$ denotes the actual label.

Specifically, our model training is not aimed at minimizing $Loss_{Domain}$. Instead, we want to make it impossible for the discriminator to tell whether the feature vector produced by the generator belongs to the source domain or the target domain. Such an operation expands the feature space of the samples, enhances the diversity of sample features, and improves the ability of the generative model to adapt to new tasks. Our justification for incorporating adversarial networks in the meta-learning framework is that, according to

domain adaptation theory, predictions must be based on features that cannot distinguish between the source and target domains in order to achieve compelling domain transfer.

3.2.3. GCN-Based Meta-Knowledge Extractor

We constructed the meta-knowledge extractor using a graph convolutional neural network (GCN). For a given N-way K-shot task, we constructed a graph $G = (V, E, A)$ to represent the relationships between instances, where V and E denote the set of nodes and the set of edges, respectively. Each instance is treated as a node $v_i \in V$, and the edges $e_{i,j} \in E$ correspond to the connectivity of neighboring nodes v_i and v_j. $A = R^{n \times n}$ denotes the adjacency matrix, where each element represents the connectivity between nodes and $n = N \times K$. Let $\boldsymbol{H} \in R^{n \times d}$ be the matrix containing all n nodes and their features. d is the dimension of the feature vector, and each row $\boldsymbol{h}_i \in R^d$ is the feature vector of node v_i. GCN can capture information about their direct neighbors through one layer of convolution, and when multiple GCN layers are stacked, information about larger neighbors can be obtained.

The output of the graph convolution operation for the L-th layer is defined as:

$$\boldsymbol{H}^{(\boldsymbol{L}+1)} = \boldsymbol{\sigma}\left(\widetilde{\boldsymbol{D}}^{-\frac{1}{2}}\widetilde{\boldsymbol{A}}\widetilde{\boldsymbol{D}}^{-\frac{1}{2}}\boldsymbol{H}^{(L)}\boldsymbol{W}_G^{(L)}\right) \tag{4}$$

where $\boldsymbol{H}^{(L)} \in R^{n \times d}$ is the node feature matrix of this layer, representing the meta-knowledge contained in the sentence obtained by the meta-knowledge extractor, $\boldsymbol{W}_G^{(l)}$ is the learnable weight matrix of the graph convolution of this layer, and σ is the activation function. In the masked language model, the node feature matrix $\boldsymbol{H}^{(0)}$ at layer 0-th is the label prediction output $\boldsymbol{X}$ of $\boldsymbol{x}_{prompt}$. $\widetilde{\boldsymbol{A}} = \boldsymbol{A} + \boldsymbol{I}_n$, where $\boldsymbol{I}_n$ is the unit matrix. The adjacency matrix is constructed as shown in Figure 2, and the adjacency matrix is constructed after experimental verification to obtain the optimal structure. $\widetilde{\boldsymbol{D}}_{ii} = \sum_j \widetilde{\boldsymbol{A}}_{ij}$ is the degree matrix of the adjacency matrix, which is used to normalize the node features. GCN convolves the features of neighboring nodes and propagates their embedded vectors to their nearest neighbors, learning global features through the information interaction of neighboring nodes, fully mining the useful information carried in small-sample instances, and obtaining a better representation of sample embeddings.

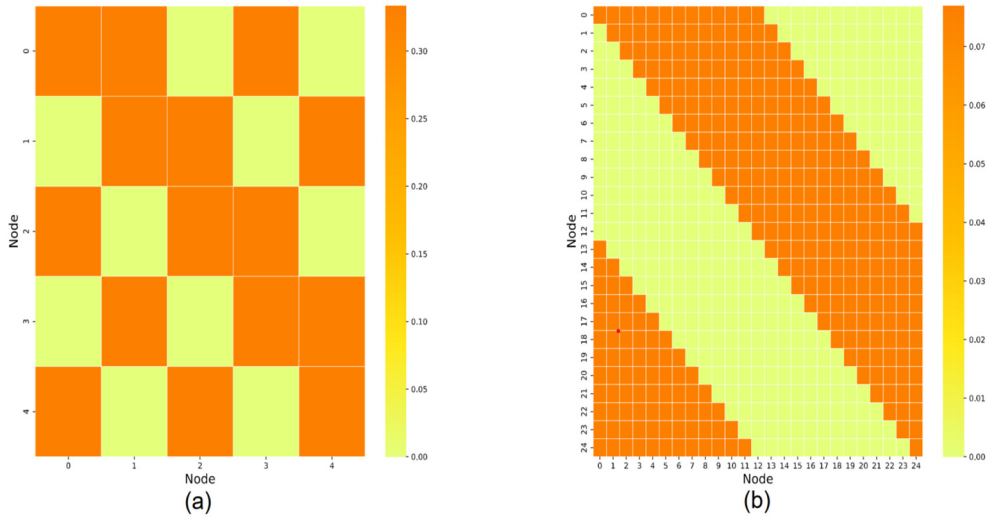

Figure 2. Visual heat map of the adjacency matrix, with orange indicating directed connections of vertical axis nodes to horizontal axis nodes and yellow indicating no connections. (**a**) is the adjacency matrix for the GCN in the 5-way 1-shot task, and (**b**) is the adjacency matrix for the GCN in the 5-way 5-shot task.

3.2.4. Class Prototype

Our meta-learning method follows an N-way K-shot training, where N-way denotes a total of N categories in the task, which we denote by $\{C_1, C_2, \ldots, C_N\}$. To obtain adequate label semantics for each category label, we introduced external knowledge to construct a class-extended label list that contains the original label words as well as words that are similar to the label words, e.g., category C_i denotes a computer-related product whose class-extended label list could be ["computer", "calculator", "machine", " analog computer", "computing", "personal computer", "programmer", ...]. The relevant terms for labeling were derived from an external knowledge graph: https://relatedwords.org (accessed on 20 May 2023). We set the length of the class extended label list to $B = 20$. Then, we encoded each element of the extended label list for each class as a d-dimensional vector using the pre-trained language model as an encoder, denoted as $\left\{ \boldsymbol{w}_{C_i}^b \right\}_{b=1}^{B}$, $\boldsymbol{w}_{C_i}^b \in R^d$. For each class prototype, we took the B d-dimensional vectors and average to obtain the final initialized class prototype:

$$\boldsymbol{w}_i^0 = \boldsymbol{mean}\left(\left\{ \left\{ \boldsymbol{w}_{C_i}^{\boldsymbol{b}} \right\}_{\boldsymbol{b}=1}^{\boldsymbol{B}} \right\} \right) \tag{5}$$

where $\boldsymbol{w}_i^0 \in R^d$ is the initial class prototype vector of class C_i; ultimately, we obtained the initial class prototype vector of N classes forming the initial class prototype matrix, denoted as $\boldsymbol{W}^0 \in R^{N \times d}$.

In particular, the encoding embedding process of the initial class prototype matrix W^0 was not part of the model training process; W^0 was initialized and saved locally, and the model training process was then imported into our framework while we iteratively updated W^0 as a learnable feature in the internal loop of meta-learning and participated in the computation of the probability distribution of the instance classification.

3.2.5. Feature Fusion Module

The specifics of our feature fusion module are shown in Figure 3, where we multiply the output $\boldsymbol{X}$ of the encoder with the template with our class prototype embedding to obtain the sentence-level probability distribution. The output $\boldsymbol{H}^{(L)}$ of the GCN feature extractor is then passed through a linear neural network classifier to obtain the class-level probability distribution:

$$P_{sen} = softmax\left(\boldsymbol{X}\boldsymbol{W}^{t\,T} + \boldsymbol{b} \right) \tag{6}$$

$$P_{class} = Classifier_{Linear}\left(\boldsymbol{H}^{(L)} \right) \tag{7}$$

where $\boldsymbol{W}^t \in R^{N \times d}$ is the initial class prototype matrix $\boldsymbol{W}^0$ updated by t-step iterations to obtain:

$$\boldsymbol{W}^t = \boldsymbol{W}^{t-1} - \beta_{task} \nabla_{\boldsymbol{W}^{t-1}} Loss_{inner} \tag{8}$$

where β_{task} is the meta-learning internal loop learning rate and $Loss_{inner}$ denotes the training loss of the support set in the meta-learning inner loop. We add up the two outputs to the final classification score.

3.2.6. Optimization

The output of the feature fusion module is passed through a $softmax$ function to obtain the final classification result and calculate the loss:

$$logit = softmax(P_{sen} + P_{class}) \tag{9}$$

$$Loss = -\frac{1}{N} \sum_{i=1}^{N} y_k (log\_softmax(logit)) \tag{10}$$

where y_k is the true data label after one-hot encoding and the $log\_softmax()$ activation function takes the logarithm of the $softmax$ function.

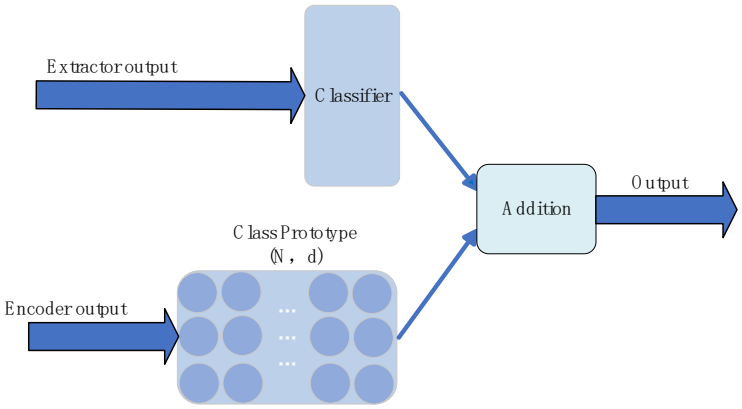

Figure 3. Framework of the feature fusion module.

In the internal loop of meta-learning, the class prototype representation $W^{(t)}$ is updated through the inner loop training loss $Loss_{inner}$ on the support set, and in the external loop the GCN is updated and the Bert fine-tuned by combining the classification loss on the query set and the discriminator loss:

$$Loss = Loss_{classification} - Loss_{Domain} \tag{11}$$

$$\theta_{M+GCN} = \theta_{M+GCN} - \beta_{meta} \nabla_{\theta_{M+GCN}} Loss \tag{12}$$

where β_{meta} is the meta-learning external loop learning rate.

4. Experiments

In this section, we conducted extensive tests to compare our suggested model with five baselines and assess performance on four publicly accessible text classification datasets, where we used accuracy as an evaluation parameter to accurately carry out our task.

4.1. Datasets

For the performance evaluation of text categorization, we selected 4 publicly accessible datasets. Table 1 provides a statistical summary for all datasets.

Table 1. Information on all experimental datasets that were used.

Dataset	Classes	Samples	Average Length
HuffPost	41	36,900	11
Amazon	24	24,000	141
Reuters	31	620	186
20 News	20	18,828	341

The **HuffPost** [35] topic classification dataset is divided into 41 categories, which also contain smaller lengths of text, and is considered more challenging for text classification. The distribution of the 41 topics is partitioned into training, validation, and test sets, with proportions of 20, 5, and 16 topics, respectively, facilitating the systematic evaluation and assessment of the model's generalization capabilities.

The **Amazon** [36] product review dataset is so big that we only used a subset of 1000 reviews from each category for text classification studies. The dataset contains review information for 24 goods. The 24 classes are segregated into training, validation, and test sets with respective allocations of 10, 5, and 9 classes, enabling rigorous examination and measurement of the model's performance across distinct categories.

The **Reuters** [37] news dataset: We used a subset of this dataset under the standard ApteMode version, of which we performed classification experiments on 31 categories. The set of 24 classes is partitioned into training, validation, and test subsets, with 10, 5, and 9

classes, respectively, facilitating systematic evaluation and benchmarking of the model's effectiveness across diverse class labels.

The **20 Newsgroups** [38] news topic classification dataset is divided into 20 different news topics, with a total of approximately 20,000 news documents. The set of 20 classes is divided into training, validation, and test subsets, with 10, 5, and 5 classes, respectively, enabling comprehensive analysis and performance assessment of the model on unseen class labels during training.

4.2. Baselines

We compared our PGCA to a number of competing baselines, which are briefly summarized as follows:

MAML obtains a set of initialization parameters for a model by constructing a meta-training task that allows the model to maximize the performance of a new task with one or a few gradient updates on a small number of samples.

PROTO-BERT uses BERT as an encoder, averages support instance embeddings as class prototypes, and calculates distances to class prototypes to predict query label results.

MLADA [17] investigates the adversarial network framework to extract sentence features and enhance the generalization and effectiveness of the meta-learning system across contexts.

SaAML [39] proposes a meta-learning framework for sentence-aware confrontation, using TCN as the core generator.

FROG-GNN [31] is a graph neural network that accumulates neighboring nodes from several aggregations to learn query embeddings.

PBML [21] combines prompt tuning and meta-learning, with base learners learning task-specific tag words and meta-learners fine-tuning template and encoder tasks. We ran the publicly available code for PBML as experimental results for comparison.

4.3. Parameter Settings

The pre-trained language model in our model makes use of $Bert_{base}$ [40]. The external learning rate was set to 3×10^{-6}, the internal learning rate to 0.01, the external update every four tasks for the 5-way 1-shot task, the external update every two tasks for the 5-way 5-shot task, and the dropout to 0.2. For the HuffPost news dataset, we set the maximum length of the encoder input to 50, whereas we set it to 300 for the other datasets. We set a three-layer graph convolutional neural network as the knowledge extractor with an input dimension and output dimension of 768. We set up a three-layer feedforward neural network as the domain discriminator. The Adam optimizer [41] was used to train our model. In addition, all experiments in this section were performed on a computer with an Intel Core i9 13900K/F CPU@5.8 GHz and a GeForce RTX 3090 GPU card with 24 G of video memory, and the model was implemented based on the Pytorch 1.12.1 framework.

4.4. Results and Analysis

4.4.1. Main Results

We compare the experimental outcomes of our suggested PGCA approach with competing methods on four widely used datasets in this section. In Table 2, the results of several models are shown for the four datasets, taking into account the 5-way 1-shot and 5-way 5-shot configurations. Our PGCA model performs best on all four datasets, with an average accuracy of 83.94% in the 5-way 1-shot classification and 88.38% in the 5-way 5-shot classification.

It is important to note that for few-shot text classification, the performance improvement in the prompt-based approach is more significant. This is because, in few-shot scenarios, where samples carry minimal useful information, pre-trained language models learn much prior knowledge and can significantly improve the classification performance of downstream tasks. Prompt learning bridges the gap between the pre-trained language

model and the downstream task, embedding the text into a feature space more suited to the target task, effectively solving the problem of overfitting.

Table 2. The reported findings demonstrate the performance of 5-way 1-shot and 5-way 5-shot text classification on four benchmark datasets in terms of accuracy (%).

	Amazon		**HuffPost**		**Reuters**		**20 News**	
	1-shot	**5-shot**	**1-shot**	**5-shot**	**1-shot**	**5-shot**	**1-shot**	**5-shot**
MAML	39.3	47.2	43.7	54.3	84.8	94.0	33.8	43.7
PROTO-BERT	68.1	82.5	52.9	69.3	86.7	93.9	37.8	45.3
MLADA	68.4	86.0	45.0	64.9	82.3	96.7	59.6	77.8
SaAML	71.47	86.37	51.26	69.44	-	-	70.79	84.30
FROG-GNN	71.5	83.6	54.1	69.6	-	-	-	-
PBML	80.03	87.58	71.54	75.71	94.91	96.97	87.50	92.32
PGCA (ours)	**80.36**	**87.68**	**72.10**	**76.10**	**95.20**	**97.29**	**88.09**	**92.45**

The bolded numerals in the table indicate optimal performance.

Our PGCA compared to PBML [21] shows an average performance improvement of 0.44% on the 5-way 1-shot task and 0.24% on the 5-way 5-shot task, which is because PBML ignores the interaction information between samples of different classes, which may result in losing some knowledge from the instances. Our PGCA uses GCN as a meta-knowledge extractor to capture the relationship information between different classes of samples through interactions between nodes, and we also incorporate adversarial networks to extend the sample feature space, increase the diversity of samples and enhance the model's capacity for generalization.

4.4.2. Ablation Experiment

We conducted an ablation study on the Amazon dataset and 20 News dataset to validate the effectiveness of each component of our PGCA. For the ablation tests, two 5-way 1-shot tasks were set up, and two tasks were internally looped for a single external loop update. Table 3 displays the outcomes of the ablation trial.

Table 3. Comparison of the accuracy (%) of ablation experiments using 5-way, 1-shot settings on the Amazon dataset and 20 News dataset. DD stands for domain discriminator.

	Amazon	**20 News**
PGCA	79.55	88.09
−w/o DD	79.32	87.74
−w/o GCN	78.60	87.68
−w/o DD + GCN	78.42	87.50

We first removed the domain discriminator and source set to verify the contribution of adversarial network to the classification task. The performance in the 5-way 1-shot tasks decreased by 0.22% and 0.35%, respectively. This confirms that our adversarial network helps to improve model performance.

We then investigated the advanced performance of GCN in extracting features. Referring to the work of [17], we replaced the GCN in the meta-knowledge extractor with a bi-directional LSTM, and the performance in the 5-way 1-shot tasks decreased by 0.95% and 0.41%, respectively. Thus, it is more effective for us to use GCN as a meta-knowledge extractor to extract the meta-knowledge contained in sentences.

Finally, we removed the meta-knowledge extractor and considered only the cue-based pre-trained language model as an encoder, with a performance drop of 1.13% and 0.59% in the 5-way 1-shot tasks. We argue that considering only the cue-based and trained language model as an encoder loses some of the information of the sentence itself, leading to a performance drop in the classification task. At the same time, our meta-knowledge

extractor can obtain more feature information through sentence interactions, which helps to improve classification performance.

4.4.3. Visualization

We used visualization to demonstrate that our model can extend the feature space of samples and enhance sample diversity. Using t-SNE [42] to visualize the phrase feature embeddings extracted using various techniques on the query set, we tested the 5-way 1-shot PGCA model on a test set of 20 Newsgroups. Compared to the embeddings obtained by the PBML model, the sentence features extracted by our PGCA model represent a larger feature space and a richer feature diversity of the samples (Figure 4).

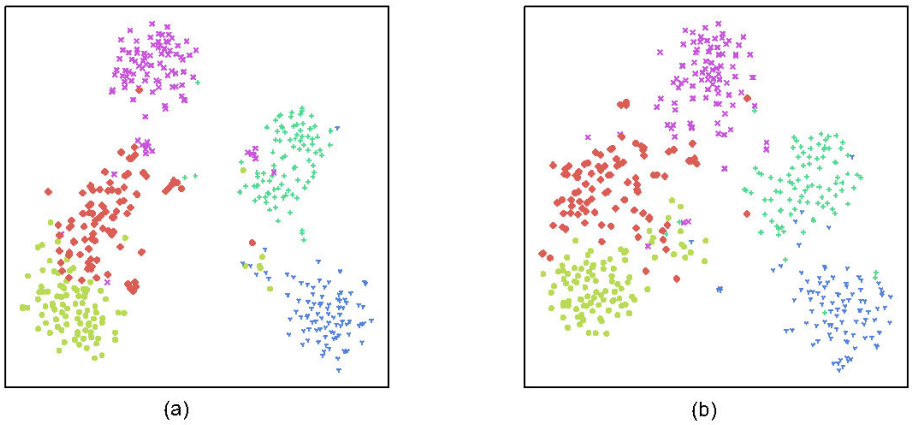

Figure 4. The t-SNE visualization displays the representation of the feature extraction module's output for the test set sampled from the 20 Newsgroups dataset. The differently colored and shaped blocks in the illustration represent instances from distinct categories. It is crucial to highlight that the five categories shown in the picture were not discernible throughout the model training phase. (**a**) corresponds to the embedding representation in the PBML model, while (**b**) represents the feature representation extracted by our PGCA method.

We also ran the PGCA model on the Amazon product review dataset to compare and visualize actual and predicted labels, and the results are shown in Figure 5. The results of the experiment show that our PGCA is relatively poor at predicting categorical labels for categories that humans perceive to be relatively similar (such as 'Instant Video' and 'Movies and TV'), which are relatively poorly differentiated, and those categories that are considered less similar by humans (such as 'Clothing Shoes and Jewelry' and the other four categories), which are more in line with human perceptions and are relatively well differentiated.

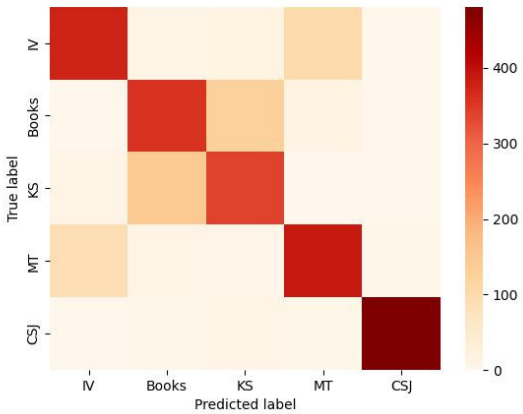

Figure 5. Above is a comparative graph depicting the real labels and model predictions of our PGCA model on the 5-way 5-shot task using the Amazon product review dataset. The labels represent the following categories: IV (Instant Video), Books (Book Products), KS (Kindle Store—Amazon's service for customer reading), MT (Movies and TV), and CSJ (Clothing Shoes and Jewelry).

5. Conclusions

In order to enhance the adaption of complicated models in short sample situations, we present a cue-based graph convolutional adversarial meta-learning framework (PGCA) in this research. Specifically, our approach utilizes a prompt-enhanced pre-trained language model as an encoder, effectively bridging the gap between pre-trained language models and downstream tasks. We then use a GCN-based knowledge extractor to obtain sentence dependencies through inter-instance interactions and further extract meta-knowledge to generate higher-quality tag embeddings. We also incorporate adversarial networks into the meta-learning framework, employing domain discriminators to counteract the knowledge extractor. This integration aims to augment the variety of sample features, extend the feature space of samples, and enhance the model's capability to adapt to new tasks. In particular, we also construct an extended list of class prototypes by introducing external knowledge to reduce the impact of extreme samples on class prototypes. Finally, we conduct experiments on four standard datasets, which show that our method PGCA is more effective compared to previous small-sample text classification methods.

Author Contributions: Conceptualization and methodology, R.G.; writing—original draft preparation, R.G.; investigation, R.G., X.Q. and W.R.; validation, R.G.; visualization, R.G.; supervision, X.Q.; project administration, X.Q. and W.R. All authors have read and agreed to the published version of the manuscript.

Funding: This study was financially supported by the major science and technology special project of Xinjiang Uygur Autonomous Region (2020A03001) and its sub-project Key technology development and application demonstration of integrated food data supervision platform in Xinjiang region (2020A03001-2).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Heidarysafa, M.; Kowsari, K.; Brown, D.E.; Meimandi, K.J.; Barnes, L.E. An improvement of data classification using random multimodel deep learning (rmdl). *arXiv* **2018**, arXiv:1808.08121.
2. Jiang, M.; Liang, Y.; Feng, X.; Fan, X.; Pei, Z.; Xue, Y.; Guan, R. Text classification based on deep belief network and softmax regression. *Neural Comput. Appl.* **2018**, *29*, 61–70. [CrossRef]
3. Kowsari, K.; Jafari Meimandi, K.; Heidarysafa, M.; Mendu, S.; Barnes, L.; Brown, D. Text classification algorithms: A survey. *Information* **2019**, *10*, 150. [CrossRef]
4. Lai, S.; Xu, L.; Liu, K.; Zhao, J. Recurrent convolutional neural networks for text classification. In Proceedings of the AAAI Conference on Artificial Intelligence, Austin, TX, USA, 25–30 January 2015.
5. Minaee, S.; Kalchbrenner, N.; Cambria, E.; Nikzad, N.; Chenaghlu, M.; Gao, J. Deep learning--based text classification: A comprehensive review. *ACM Comput. Surv. CSUR* **2021**, *54*, 62. [CrossRef]
6. Kim, Y. Convolutional neural networks for sentence classification. *arXiv* **2014**, arXiv:1408.5882.
7. Zhang, X.; Zhao, J.; LeCun, Y. Character-level convolutional networks for text classification. In *Advances in Neural Information Processing Systems 28, Proceedings of the Annual Conference on Neural Information Processing Systems 2015 (NIPS 2015), Montreal, QC, Canada, 7–12 December 2015*; Cortes, C., Lawrence, N., Lee, D., Sugiyama, M., Garnett, R., Eds.; NeurIPS: La Jolla, CA, USA, 2015; Volume 28.
8. Tang, D.; Qin, B.; Liu, T. Document modeling with gated recurrent neural network for sentiment classification. In Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, Lisbon, Portugal, 17–21 September 2015; pp. 1422–1432.
9. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, Ł.; Polosukhin, I. Attention is all you need. In Proceedings of the 30th Conference on Neural Information Processing Systems (NIPS 2017) Advances in Neural Information Processing Systems 30, Long Beach, CA, USA, 4–9 December 2017; Volume 30.
10. Brown, T.; Mann, B.; Ryder, N.; Subbiah, M.; Kaplan, J.D.; Dhariwal, P.; Neelakantan, A.; Shyam, P.; Sastry, G.; Askell, A. Language models are few-shot learners. In *Advances in Neural Information Processing Systems 33, Proceedings of the 33rd Conference on Neural Information Processing Systems (NIPS 2020), Virtual Event, 6–12 December 2020*; The MIT Press: Cambridge, MA, USA, 2020; Volume 33, pp. 1877–1901.

11. Lester, B.; Al-Rfou, R.; Constant, N. The power of scale for parameter-efficient prompt tuning. *arXiv* **2021**, arXiv:2104.08691.
12. Bao, Y.; Wu, M.; Chang, S.; Barzilay, R. Few-shot text classification with distributional signatures. *arXiv* **2019**, arXiv:1908.06039.
13. Dong, B.; Yao, Y.; Xie, R.; Gao, T.; Han, X.; Liu, Z.; Lin, F.; Lin, L.; Sun, M. Meta-information guided meta-learning for few-shot relation classification. In Proceedings of the 28th International Conference on Computational Linguistics, Barcelona, Spain, 8–13 December 2020; pp. 1594–1605.
14. Finn, C.; Abbeel, P.; Levine, S. Model-agnostic meta-learning for fast adaptation of deep networks. In Proceedings of the International Conference on Machine Learning, Sydney, Australia, 6–11 August 2017; pp. 1126–1135.
15. Geng, R.; Li, B.; Li, Y.; Zhu, X.; Jian, P.; Sun, J. Induction networks for few-shot text classification. *arXiv* **2019**, arXiv:1902.10482.
16. Hospedales, T.; Antoniou, A.; Micaelli, P.; Storkey, A. Meta-learning in neural networks: A survey. *IEEE Trans. Pattern Anal. Mach. Intell.* **2021**, *44*, 5149–5169. [CrossRef]
17. Han, C.; Fan, Z.; Zhang, D.; Qiu, M.; Gao, M.; Zhou, A. Meta-learning adversarial domain adaptation network for few-shot text classification. *arXiv* **2021**, arXiv:2107.12262.
18. Nichol, A.; Achiam, J.; Schulman, J. On first-order meta-learning algorithms. *arXiv* **2018**, arXiv:1803.02999.
19. Lei, T.; Hu, H.; Luo, Q.; Peng, D.; Wang, X. Adaptive Meta-learner via Gradient Similarity for Few-shot Text Classification. *arXiv* **2022**, arXiv:2209.04702.
20. Snell, J.; Swersky, K.; Zemel, R. Prototypical networks for few-shot learning. In Proceedings of the 30th Conference on Neural Information Processing Systems (NIPS 2017) Advances in Neural Information Processing Systems 30, Long Beach, CA, USA, 4–9 December 2017; Volume 30.
21. Zhang, H.; Zhang, X.; Huang, H.; Yu, L. Prompt-based meta-learning for few-shot text classification. In Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, Abu Dhabi, United Arab Emirates, 7–11 December 2022; pp. 1342–1357.
22. Gori, M.; Monfardini, G.; Scarselli, F. A new model for learning in graph domains. In Proceedings of the 2005 IEEE International Joint Conference on Neural Networks, Montreal, QC, Canada, 31 July–5 August 2005; pp. 729–734.
23. Scarselli, F.; Gori, M.; Tsoi, A.C.; Hagenbuchner, M.; Monfardini, G. The graph neural network model. *IEEE Trans. Neural Netw.* **2008**, *20*, 61–80. [CrossRef] [PubMed]
24. Battaglia, P.; Pascanu, R.; Lai, M.; Jimenez Rezende, D. Interaction networks for learning about objects, relations and physics. In Proceedings of the 29th Conference on Neural Information Processing Systems (NIPS 2016) Advances in Neural Information Processing Systems 29, Barcelona, Spain, 5–10 December 2016; Volume 29.
25. Hoshen, Y. Vain: Attentional multi-agent predictive modeling. In Proceedings of the 30th Conference on Neural Information Processing Systems (NIPS 2017) Advances in Neural Information Processing Systems 30, Long Beach, CA, USA, 4–9 December 2017; Volume 30.
26. Liang, X.; Shen, X.; Feng, J.; Lin, L.; Yan, S. Semantic object parsing with graph lstm. In Proceedings of the Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, 11–14 October 2016; pp. 125–143.
27. Wang, X.; Ye, Y.; Gupta, A. Zero-shot recognition via semantic embeddings and knowledge graphs. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 6857–6866.
28. Garcia, V.; Bruna, J. Few-shot learning with graph neural networks. *arXiv* **2017**, arXiv:1711.04043.
29. Yao, L.; Mao, C.; Luo, Y. Graph convolutional networks for text classification. In Proceedings of the AAAI Conference on Artificial Intelligence, Honolulu, HI, USA, 27 January–1 February 2019; pp. 7370–7377.
30. Linmei, H.; Yang, T.; Shi, C.; Ji, H.; Li, X. Heterogeneous graph attention networks for semi-supervised short text classification. In Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), Hong Kong, China, 3–7 November 2019; pp. 4821–4830.
31. Xu, S.; Xiang, Y. Frog-GNN: Multi-perspective aggregation based graph neural network for few-shot text classification. *Expert Syst. Appl.* **2021**, *176*, 114795. [CrossRef]
32. Schick, T.; Schütze, H. Exploiting cloze questions for few shot text classification and natural language inference. *arXiv* **2020**, arXiv:2001.07676.
33. Li, X.L.; Liang, P. Prefix-tuning: Optimizing continuous prompts for generation. *arXiv* **2021**, arXiv:2101.00190.
34. Vinyals, O.; Blundell, C.; Lillicrap, T.; Wierstra, D. Matching networks for one shot learning. In Proceedings of the 29th Conference on Neural Information Processing Systems (NIPS 2016) Advances in Neural Information Processing Systems 29, Barcelona, Spain, 5–10 December 2016; Volume 29.
35. Misra, R. News category dataset. *arXiv* **2022**, arXiv:2209.11429.
36. He, R.; McAuley, J. Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering. In Proceedings of the 25th International Conference on World Wide Web, Montreal, QC, Canada, 11–15 April 2016; pp. 507–517.
37. Lewis, D. *Reuters-21578 Text Categorization Test Collection, Distribution 1.0*; AT&T Labs-Research: Atlanta, GA, USA, 1997.
38. Lang, K. Newsweeder: Learning to filter netnews. In *Machine Learning Proceedings 1995, Proceedings of the Twelfth International Conference on Machine Learning, Tahoe City, CA, USA, 9–12 July 1995*; Elsevier: Amsterdam, The Netherlands, 1995; pp. 331–339.
39. Wang, S.; Liu, X.; Liu, B.; Dong, D. Sentence-aware Adversarial Meta-Learning for Few-Shot Text Classification. In Proceedings of the 29th International Conference on Computational Linguistics, Gyeongju, Republic of Korea, 12–17 October 2022; pp. 4844–4852.

40. Devlin, J.; Chang, M.-W.; Lee, K.; Toutanova, K. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv* **2018**, arXiv:1810.04805.
41. Kingma, D.P.; Ba, J. Adam: A method for stochastic optimization. *arXiv* **2014**, arXiv:1412.6980.
42. Van der Maaten, L.; Hinton, G. Visualizing data using t-SNE. *J. Mach. Learn. Res.* **2008**, *9*, 2579–2605.