sensors

MDPI

Article

Graph Adaptation Network with Domain-Specific Word Alignment for Cross-Domain Relation Extraction

Zhe Wang [1], Bo Yan [2,*], Chunhua Wu [1], Bin Wu [1], Xiujuan Wang [3] and Kangfeng Zheng [1]

[1] School of Cyberspace Security, Beijing University of Posts and Telecommunications, Beijing 100876, China; wangxiaozhe@bupt.edu.cn (Z.W.); wuchunhua@bupt.edu.cn (C.W.); binwu@bupt.edu.cn (B.W.); kfzheng@bupt.edu.cn (K.Z.)
[2] School of Computer Science, Beijing University of Posts and Telecommunications, Beijing 100876, China
[3] School of Computer Science, Beijing University of Technology, Beijing 100124, China; xjwang@bjut.edu.cn
* Correspondence: mjkbyb@bupt.edu.cn; Tel.: +86-176-1124-2518

Received: 6 November 2020; Accepted: 8 December 2020; Published: 15 December 2020

Abstract: Cross-domain relation extraction has become an essential approach when target domain lacking labeled data. Most existing works adapted relation extraction models from the source domain to target domain through aligning sequential features, but failed to transfer non-local and non-sequential features such as word co-occurrence which are also critical for cross-domain relation extraction. To address this issue, in this paper, we propose a novel tripartite graph architecture to adapt non-local features when there is no labeled data in the target domain. The graph uses domain words as nodes to model the co-occurrence relation between domain-specific words and domain-independent words. Through graph convolutions on the tripartite graph, the information of domain-specific words is propagated so that the word representation can be fine-tuned to align domain-specific features. In addition, unlike the traditional graph structure, the weights of edges innovatively combine fixed weight and dynamic weight, to capture the global non-local features and avoid introducing noise to word representation. Experiments on three domains of ACE2005 datasets show that our method outperforms the state-of-the-art models by a big margin.

Keywords: relation extraction; domain adaptation; graph convolution network; non-local features

1. Introduction

The Internet of Things (IoT) is a large-scale paradigm. In the IoT paradigm, devices communicate with each other regardless of their owner. Communication is not only between machines and people, but also between machines [1] and between machines and intelligent objects [2]. The IoT of connected devices includes different fields, such as healthcare, agriculture, smart cities, smart homes, smart grids, automated vehicles, asset monitoring, environmental monitoring, education, industry, etc. [3]. Each field contains a large amount of data information, and the data distribution in different fields is different. At the same time, each of the above domains contains a large number of sensors, actuators, gateways, servers and related end-user applications [1,3,4]. Data collected from interconnected objects or things are used to produce results in different IoT applications. The basic modules of any IoT solution are connected devices, communication networks, services, management, security and applications [1,5]. Among them, the data information of different fields in the Internet of Things is equivalent to the source domain and target domain information in this article. The cross-domain relationship extraction in the IoT will help the collection, mining and classification of data information.

Collecting traffic information from social networks and using it for travel safety are two challenging issues in Intelligent Transportation Systems (ITSs). The transportation network can be monitored through sensor devices and social network data. Currently, ITSs uses sensor devices

to monitor all aspects of the transportation network. ITSs can use social network data to support traffic and control management and to check transportation services. The analysis results of these data can help travelers travel safely, solve problems related to traffic congestion, and increase travel in urban areas. However, ITSs may not be able to collect accurate traffic information from these sensors. In addition, in social media, information related to traffic comes with cross-domain text information, which makes the task of transport text mining and classification more difficult for ITSs [6–9]. This paper takes the relation extraction of cross-domain information as an entry point to explore cross-domain text mining and classification techniques. Relation extraction plays a pivotal role in addressing the issue of information extraction, which aims to detect the semantic relationship between real-world entities [10].

Relation extraction refers to extracting the semantic relation between two candidate entities, i.e., triples such as (United States, Country—President, Trump). In real life, the target domain often has only a small amount of labelled data or even no labelled data; thus, using source domain-labelled data to improve the extraction of relations from the target domain is necessary. However, due to the different data distributions of different domains, directly applying a model trained on the source domain to the target domain results in great performance decreases.

Most works have solved this issue by extracting shared features between the source and target domains and then using these features to perform relation extraction. References [11–13] used manually crafted features, such as word clustering and latent semantic analysis, to learn the general representations of different domain words. Due to the limited manual feature space, this method was unable to capture enough shared features for relation extraction. The authors in [14,15] used deep learning methods to project the source features and target features into one unified space and then performed adversarial training to automatically extract the shared features. To avoid introducing domain-specific features, the authors in [16] improved the domain-separated network [17] to extract domain-specific features and domain-independent features separately. However, this method suffered information loss as a result of introducing word embedding reconstruction.

Other works focused on aligning the domain-specific features between domains. References [18,19] believed that the features gradually transition from being domain-independent to being domain-specific as the depth of the network increases; thus, they proposed deep adaptation networks to align these deep specific features. However, both above methods only enhanced the transferability of the local or sequential features in domain-specific layers while ignoring non-local features. In addition, they applied their ideas in the image field rather than relation extraction. In some cases, non-local features, such as word co-occurrences and co-references, are also significant for cross-domain relation extraction. Figure 1 shows an example under this situation. The source domain data and target domain data have the same gold relation PER-SOC, but the contexts near entities are very different between the two domains; specifically, the domain-specific words have strong domain relevance. The traditional local-feature-based models cannot align these specific features because they are semantically irrelevant; as a result, they obtain the wrong relation ORG-AFF. On the other hand, it can be observed that some specific words are often co-occurring with shared words. According to [20], the more times these specific words appear together with shared words, the greater they should be aligned. Reference [20] aligned domain-specific words with a bipartite graph based on shared and specific word co-occurrences. This approach used spectral clustering to reduce the gap between specific features across domains. However, these manually crafted features are too limited to scale.

To solve the problems mentioned above, inspired by [20], we propose an end-to-end graph adaptation network to adapt non-local features between domains. Graph structures effectively capture non-local features in relation extraction [21,22], but no studies have applied graph structures to cross-domain relation extraction, the edges in the proposed graph only exist between specific words and shared words. In this paper, source domain words and target domain words indirectly connect through shared words as intermediate agents and word co-occurrence information as the fixed weights

of edges. However, word co-occurrence information has a strong dependence on the corpus and inevitably introduces some noise; therefore, we propose a method to calculate the dynamic weights of the edges using the attention mechanism. The fixed weights and dynamic weights are combined as the final edge weights.

Figure 1. Limitation of traditional methods. Words in red color indicate candidate entities. Shared word indicates the words both appeared in the source domain and target domain. Specific word indicates the words only appeared in the source/target domain.

The word information is propagated through graph convolutions on the tripartite graph so that the domain-specific word representations are aligned. After adapting the non-local features, i.e., obtaining the aligned word representations by the graph adaptation network, we perform adversarial training to extract the shared local features between domains. These word representations are aligned before being fed into downstream modules; therefore, the model can extract shared features effectively. Finally, the shared features are fed to a fully connected layer to perform relation extraction. Because we assume that only source domain data have labels, the relation classifier is trained using only source-labelled data. In addition, to prevent the transfer of irrelevant information from words, we select valuable edges and remove irrelevant edges depending on their fixed weights. Using this method, the model not only preserves useful information for adapting non-local features but also speeds up the calculation process.

The major contributions of our work are as follows:

- We propose a novel graph adaptation network to align domain-specific features. Local features and non-local features are transferred simultaneously for cross-domain relation extraction. This is also the first work to adapt the non-local features between domains.
- Unlike the traditional graph convolutional network, our network combines fixed weights and dynamic weights as the edge weights of the graph. In addition, rather than using a fully connected graph, we only keep valuable edges based on their edge weights. These strategies can transfer useful information effectively and avoid introducing irrelevant noise.
- Experiments have shown that non-local features such as word co-occurrences are also important for cross-domain relation extraction. The proposed method to calculate weights and select edges can capture more non-local features and better avoid noise interference than other methods.

The rest of this paper is organized as follows. Section 2 presents the work carried out in the field of cross-domain relation extraction. Section 3 introduces the task definition of the relation extraction.

Section 4 describes a graph adaptation network for cross-domain relation extraction. Section 5 presents experiments that analyze the effectiveness of our model. Section 6 concludes and presents future work.

2. Related Work

Data collection and data query are important applications in wireless sensor networks (WSN) and the IoT, and data collection and data query are usually information-centric. We noticed that WSN is a special network, in which each sensor will involve sensor data in different fields (for example, smart cities, healthcare, agriculture, etc.). Each field contains a large amount of data information, and the data distribution in different fields is different. In the last decade, deep learning (DL) has made breakthroughs in natural language processing (NLP), image processing and reinforcement learning, so that making breakthroughs in the field of artificial intelligence (AI) occupy the dominant position. Presently, much research contributed to information mining and the utility of massive data [23]. In addition, Lei et al. used multi-sensor data in fault detection of gearbox [24]. Safizadeh et al. studied multi-sensor data fusion to improve the performance of fault recognition for rolling element bearings [25]. Jing et al. combined deep neural networks and multi-sensor data fusion in the fault detection of planetary gearbox [26]. Compared with these fields, DL methods for cross-domain data analysis of sensors are relatively scarce [27]. At the same time, few papers researched cross-domain deep feature learning and fusion models. However, the representativeness of common features will obviously affect the ability of cross-domain information mining and classification. These problems encourage researchers to find a new method to adaptively extract cross-domain relationships, which in turn helps to mine and classify information in different fields.

Cross-domain relation extraction aims to solve the problem of the training set and test set with different data distributions. Reference [11] was the first work to adapt a relation extraction model to other domains, and it used generalized approaches such as word clustering to extract shared features. The authors in [13,28] combined hand-crafted features, such as dependency paths and learned word embeddings, for cross-domain relation extraction. These methods use manually crafted features to adapt existing features, so they are limited and lose some information. The deep learning approach was applied to cross-domain relation extraction in [29]. It combined feature-based methods and neural networks to exploit their advantages. Reference [15] used adversarial training to extract shared features by introducing a gradient reversal layer (GRL) [30], but it simply projected source features and target features into one unified space, inevitably introducing some domain-specific features that harmed the performance of extracting relations with the target domain. Reference [16] proposed a genre separation network to extract shared features and specific features separately. Reference [31] applied cross-view training [32] to a domain adversarial neural network (DANN) [15] and adapted shared features in different views; this is a highly fine-gained method.

Massive domain adaptation methods were also applied in other tasks. Reference [20] used spectral clustering to unify specific word representations in the context of text classification. Reference [33] used some labelled data from the target domain for learning domain-specific information. Reference [34] aligned different domain cells of the sequence model to perform domain adaptation; this is another fine-grained method. In the image field, [18,19] aligned deep specific features using distance metrics such as the maximum mean discrepancy (MMD). These methods provided us with inspiration to align domain-specific features for the task of cross-domain relation extraction task, but they only transferred local features while ignoring non-local features. Although [20] aligned specific features by using non-local features such as word co-occurrence information, complex feature engineering was needed.

Recently, graph structures were widely used in natural language processing tasks to capture non-local features. Reference [21] applied graph convolutions to pruned dependency trees and automatically captured the dependence information. Reference [35] used a graph convolutional network to model the co-referent and identical mentions between words. Reference [22] combined linear and dependency structures to improve the extraction of overlapping relations. Reference [36]

proposed an entity-relation graph to perform joint type inference on entities and relations and used the entity-relation bipartite graph in a highly efficient and interpretable way. Reference [37] proposed a graph-based method to improve word embeddings. Reference [38] used graph neural networks with generated parameters to improve multi-hop reasoning. To specify the weights of neighbors automatically without requiring any kind of costly matrix operation or depending on knowing the graph structure upfront [39], the authors in [39,40] introduced an attention mechanism for the graph structure. These studies inspire us to use a suitable graph structure to model cross-domain relation extraction problems.

3. Task Definition

3.1. Relation Extraction

Given a set of labeled corpus $D = \{(s_1, e_{11}, e_{12}, r_1), \ldots, (s_n, e_{n1}, e_{n2}, r_n)\}$, where e_{i1} and e_{i2} ($i = 1, 2, \ldots, n$) denote the first and second candidate entity respectively, r_i represents the relation type, s_i represents a sentence, relation extraction can be regarded as a classification task that applying a classifier f trained on D to the test datasets $D^{'} = \{(s_1^{'}, e_{11}^{'}, e_{12}^{'}), \ldots, (s_n^{'}, e_{n1}^{'}, e_{n2}^{'})\}$. In other words, considering the task where X is the input space and Y is the set of relation labels, the goal of the learning algorithm is to build a classifier $f{:}X \to Y$ with a low loss $L(D^{'}) = E_{(s^{'}, e_1^{'}, e_2^{'}, r^{'}) \sim D^{'}} P(f(s^{'}, e_1^{'}, e_2^{'}) \neq r^{'}))$.

3.2. Cross-Domain Relation Extraction

Given a set of source labeled corpus $D_s = \{(s_1, e_{11}, e_{12}, r_1), \ldots, (s_n, e_{n1}, e_{n2}, r_n)\}$ and target unlabeled corpus $D_t = \{(s_1^{'}, e_{11}^{'}, e_{12}^{'}), \ldots, (s_n^{'}, e_{n1}^{'}, e_{n2}^{'})\}$, the meanings of these symbols are the same as Section 3.1. It is worth noting that we assume that there is no labels in target domain data. Then cross-domain relation extraction can be regarded as a classification task that uses source domain labeled data D_s and target domain unlabeled data D_t to train a classifier f, and applies f to target domain. The goal of the learning algorithm is to build a classifier $f{:}X \to Y$ with a low loss $L(D_t) = E_{(s^{'}, e_1^{'}, e_2^{'}, r^{'}) \sim D_t} P(f(s^{'}, e_1^{'}, e_2^{'}) \neq r^{'}))$.

4. Our Methodology

In our work, we take D_s and D_t as inputs to design an algorithm that can improve the extraction of relations from the target domain. First, D_s and D_t are fed into an embedding layer to obtain an embedding matrix, and then the graph convolutions work on the embedding matrix to align domain-specific word representations. A feature extractor takes the adjusted embedding matrix as input to obtain the shared features. To force the feature extractor to extract these shared features, a domain discriminator is added after the feature extractor. Finally, the shared features are fed into a relation classifier to perform classification.

In brief, our model consists of four modules: an adaptation module, an embedding layer, a shared feature extractor and a relation classifier. The adaptation module contains a domain discriminator and a graph convolutional network (GCN) layer, which are responsible for local shared feature extraction and non-local feature alignment, respectively. Figure 2 shows the overall architecture of our model. We introduce these modules in detail below.

4.1. Adaptation Module

The adaptation module mainly contributes to extracting the shared features between domains and aligning domain-specific features. The traditional approach only adapts sequential features but ignores non-sequential or non-local features. Our adaptation module consists of two processes: local information adaptation and non-local information adaptation. Local information adaptation is applied at the sentence level; in other words, we extract the shared features of the source and target sentences. While non-local information adaptation is a word-level adaptation, it uses a GCN to

align domain-specific word representations. The remainder of this section elaborately introduces the adaptation layer.

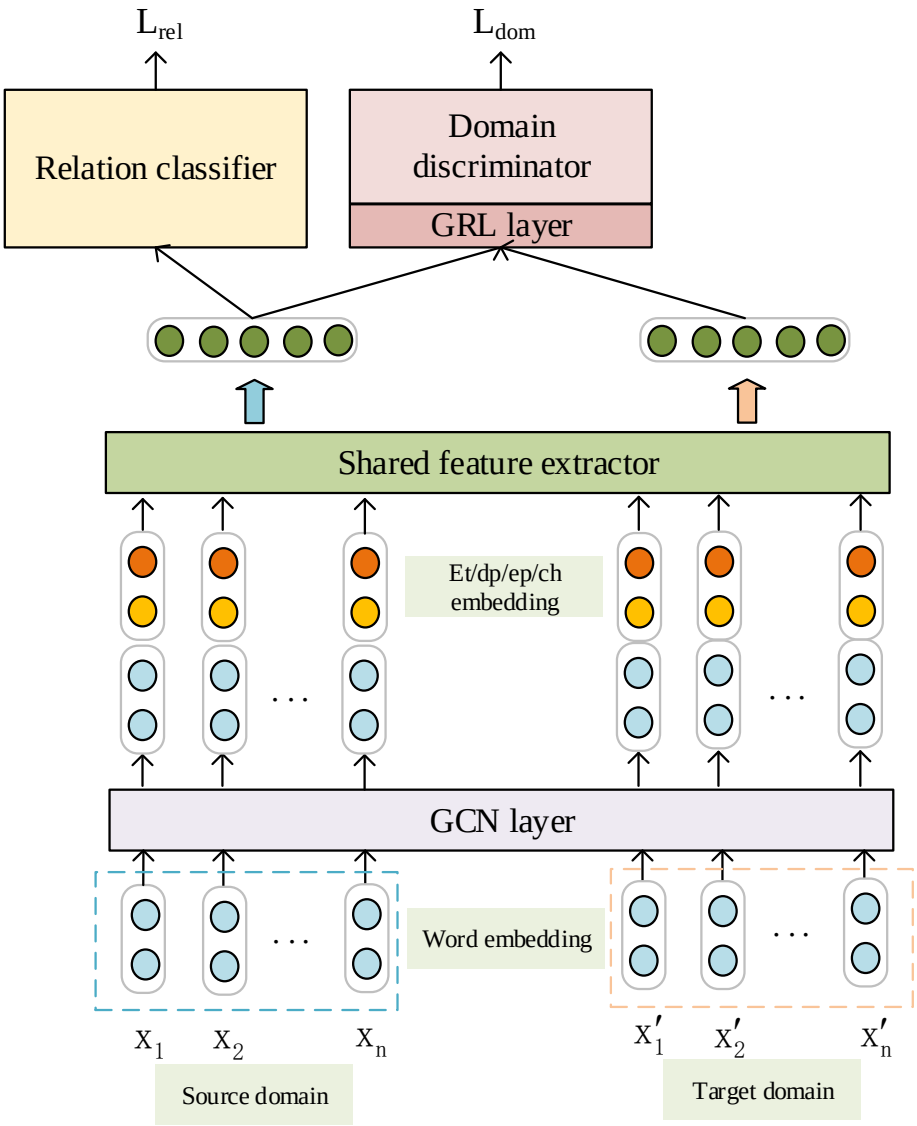


Figure 2. The overall architecture of our graph adaptation network.

(1) Local information adaptation (sentence-level).

To make the shared feature extractor capture domain-invariant features, a domain discriminator is added after the shared feature extractor. It takes s_s and s_t as inputs, where s_s and s_t represent the source and target features extracted by the shared feature extractor, respectively. The domain discriminator is implemented by a simple neural network with one hidden layer and performs binary classification to predict the domain that a sample comes from. The domain discriminator loss is defined as cross entropy loss:

$$L_{dom} = -\frac{1}{N_s + N_t} \sum_{i=1}^{N_s+N_t} (1 - y_i) log(1 - p_i) + y_i log p_i \quad (1)$$

In this equation, N_t denotes the total number of target domain data, p_i is the probability of one sample belonging to the source domain and $y_i \in \{0, 1\}$ indicates that the sample comes from the source domain (1) or the target domain (0).

To confuse the domain discriminator, a gradient reversal layer (GRL) [30] is used between the shared feature extractor and domain discriminator. Then, the forward and back propagations are formulated as follows:

$$GRL(x) = x \tag{2}$$

$$\frac{d(GRL(x))}{dx} = -I \tag{3}$$

Through reversing the gradient before domain discriminator, the parameters of domain discriminator are optimized to reduce the domain discriminator loss L_{dom} while the parameters of shared feature extractor will make L_{dom} increase. The adversarial training finally converged so that the discriminator cannot distinguish which sample comes from which domain, in other words, the shared feature extractor captures some domain-invariant features.

(2) Non-local information adaptation (word-level).

Word-vectorized representations, such as word2vec [41] and Glove [42], have greatly improved downstream applications. However, in cross-domain relation extraction, the representations of domain-specific words differ significantly between domains, and this causes poor performance when applying a model to other domains. Most previous works only focused on aligning different domain features at the sentence level [15,16,19] while ignoring word-level alignment. Inspired by [20], rather than using feature-based methods, we use a GCN to model the word co-occurrences of different domains. Through this alignment, the word representation gap between the source domain and target domain can be reduced, thereby enabling the downstream module to extract the shared features in a fine-grained way. Figure 3 shows the architecture of the GCN layer.

Word co-occurrence tripartite graph construction: The key idea of non-local information adaptation is, in the tripartite graph,if two domain-specific words have connections to more common domain-independent words in the graph, they tend to be aligned together with higher probability, i.e., have similar word representation [20]. Given the source domain sentence set $D_s = \{S_1, S_2, \ldots, S_n\}$ and target domain sentence set $D_t = \{S_1^{'}, S_2^{'}, \ldots, S_n^{'}\}$, we construct a graph $G = (V_s \cup V_i \cup V_t; E_{si} \cup E_{ti})$ for any two sentence $S_i = \{w_1, w_2, \ldots, w_n\}$ and $S_j^{'} = \{w_1^{'}, w_2^{'}, \ldots, w_n^{'}\}$. Here, V_s, V_i, V_t denote the graph vertex that corresponds to domain-specific words in S_i, domain-independent words in $S_i \cup S_j^{'}$ and domain-specific words in $S_j^{'}$ respectively, E_{si} represents the graph edges between V_s and V_i, E_{ti} represents edges between V_t and V_i. See Figure 3 for details.

We use two types of weights for the graph edges: fixed weights and dynamic weights. First, the pointwise mutual information (PMI) of two words is used as a fixed weight. The PMI is an algorithm for calculating the correlation between two variables. Here, we use the PMI to measure the co-occurrence relationship of two words:

$$A_{i \neq j} = log \frac{p(w_i, w_j)}{p(w_i)p(w_j)} \tag{4}$$

$$p(w_i, w_j) = \frac{\#win(w_i, w_j)}{\#win} \tag{5}$$

$$p(w_i) = \frac{\#win(w_i)}{\#win} \tag{6}$$

where $\#win(w_i, w_j)$ refers to the number of sliding windows that w_i and w_j appeared together. $\#win$ is the total number of sliding windows. We calculate these on the whole corpus. A higher PMI value means that w_i and w_j appear together on the corpus more times; thus, they have a higher correlation. A small PMI value means there is little correlation between w_i and w_j because they seldom appear together. After we obtain the weight matrix $A \in R^{m*m}$, (m is the length of the dictionary), the weights are normalized by:

$$f_{i \neq j} = \frac{A_{ij} - a}{b - a} \tag{7}$$

where a is the minimum element of $A_{i \neq j}$ and b is the maximum element of $A_{i \neq j}$. Then the fixed weights are defined as:

$$F_{ij} = \begin{cases} f_{ij}, & i \neq j \& f_{ij} > \alpha \\ 1, & i = j \\ 0, & others \end{cases} \tag{8}$$

The graph only keeps edges with $f_{ij} > \alpha$, where $\alpha > 0$ is a hyperparameter. When α increases, there will be fewer edges in the graph. The effects of different values of α are discussed in the experiment later in this paper.

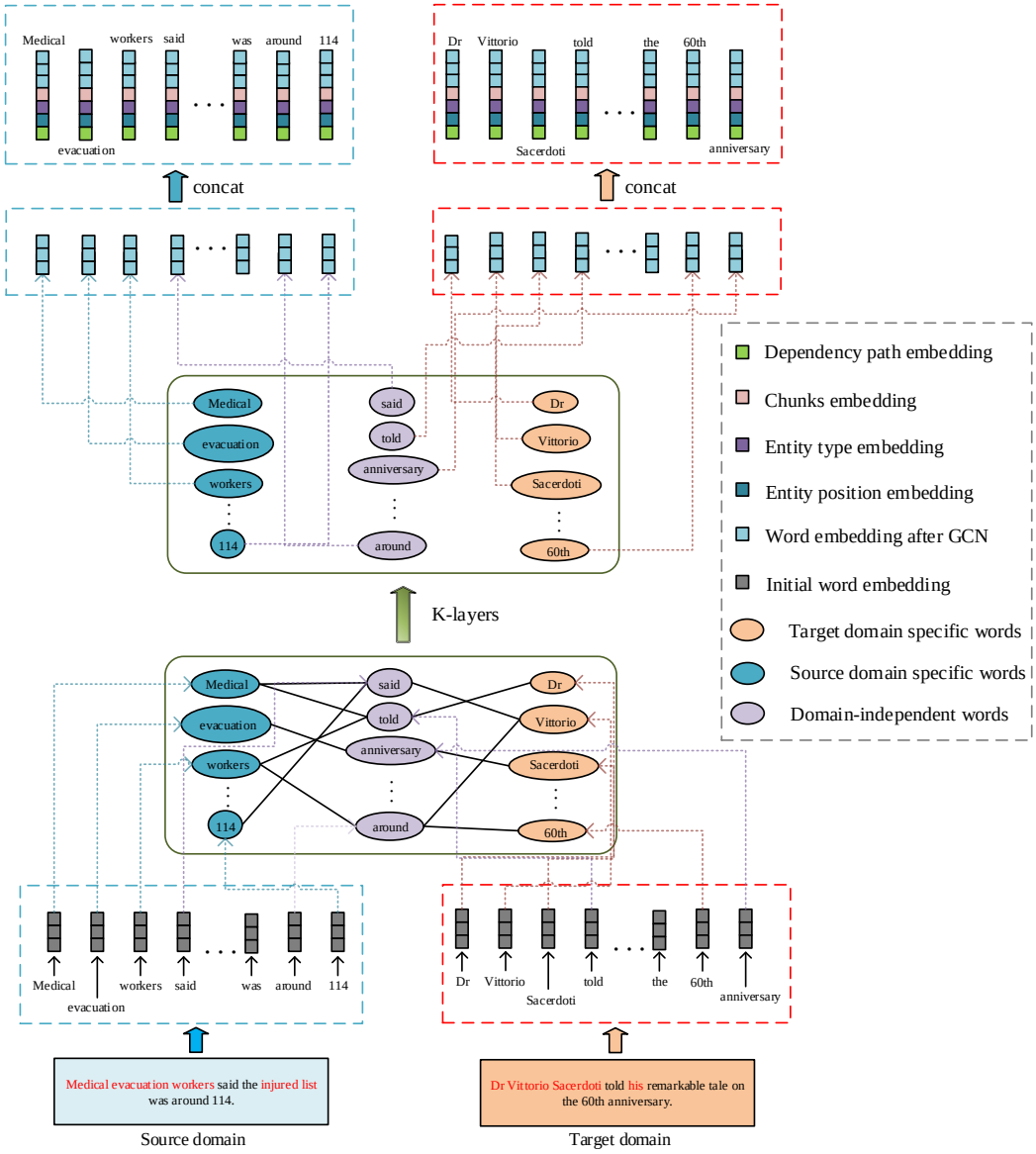


Figure 3. The architecture of GCN layer for aligning domain-specific words.

Fixed weights capture the word co-occurrence features, but they have difficulty aligning domain-specific words. There are two limitations of using fixed weights: (1) some English stop

words such as "*is*" and "*the*" are often domain-independent, and they have a high probability of appearing together with domain-specific words; therefore, the fixed weight between them is large. However, these stop words have little semantic meaning and harm the word representations. (2) Some domain-specific words are rare and seldom appear together with domain-independent words, so the fixed weight is almost 0, but these words should probably also be aligned; fixed weights cannot achieve this.

To compensate for the limitations of fixed weights, inspired by [39], attention mechanism-based dynamic weights are used. First, to increase the power of the feature expressions, a linear transformation is applied on every node in the graph:

$$h_i = w_l h_i + b_l \tag{9}$$

where $h_i \in R^n$ is the vector representation of node i, $w_l \in R^{n*n}$ and $b_l \in R^n$ is the parameters of linear transformation. For a node i in graph, $N(i)$ is defined as the nodes which directly connect to the node i and $\forall j \in N(i)$, $f_{ij} > \beta$. We set $\beta = 0.3$ for balancing computational efficiency and model effectiveness. Then we calculate the attention weight on $N(i)$ for every node i:

$$e_{ij} = LeakyReLU(h_i^T W_{att} h_i) \tag{10}$$

where $W_{att} \in R^{n*n}$ is the attention parameter and $j \in N(i)$, e_{ij} is the attention weight that indicates how important the node j to node i, $LeakyReLU$ is an activate function. The dynamic weights of node i are normalized by $softmax$ function:

$$D_{ij} = \frac{exp(e_{ij})}{\sum_{j \in N(i)} exp(e_{ij})} \tag{11}$$

Dynamic weights can be adjusted during training to provide a flexible way to train the parameters of the model. In addition, each fixed weight can be seen as a global weight because it is calculated based on statistics of the whole corpus, while a dynamic weight can be seen as a local weight because it only uses two sentences per calculation. In the end, we combine the fixed weights and dynamic weights as our final graph weights:

$$W_{ij} = F_{ij} + D_{ij} \tag{12}$$

The process of calculating the edge weights is illustrated in Figure 4 (left).

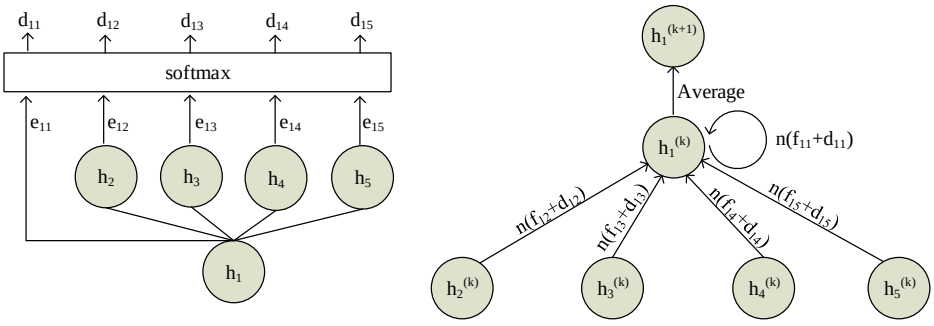


Figure 4. (**Left**): the process of calculating the dynamic attention weights. (**Right**): the information aggregation process. $n(\cdot)$ denotes a normalized function.

Graph convolutions on the tripartite graph: In this section, we first introduce the graph convolution operation and edgewise gating mechanism and then elaborate on how these methods are used in our model.

A GCN [43] is used to capture non-local and nonsequential information. Specifically, given a graph $G = (V, E)$, where V is the node set and E is the edge set, the graph convolution operation

is applied on every node and propagates information to other nodes along the edges. For a 1-layer GCN, the information only transfers to neighboring nodes, and the information of an n-layer GCN can transfer to further nodes as n increases. The information propagation from layer k to layer $k+1$ can be formulated as:

$$h_i^{(k+1)} = \frac{1}{d(i)} \sum_{j \in N(i)} W_{ij}(w_g^{(k)} h_j^{(k)} + b_g^{(k)}) \tag{13}$$

where $d(i) = \sum_{j \in N_{(i)}} W_{ij}$ is sum of all the weights between node i and its neighbors. $w_g^{(k)} \in R^{m*m}$ and $b_g^{(k)} \in R^m$ are layer specific parameters. $h_j^{(k)} \in R^m$ is vector representation of node j. Figure 4 (right) gives visual representation of the information propagation.

Edge-wise gate [44] is proposed to control how much information is transferred from neighbors. The scalar gate value of each neighbor is calculated as:

$$g_j^{(k)} = \sigma(w_e^{(k)} h_j^{(k)} + b_e^{(k)}) \tag{14}$$

Here $w_e^{(k)} \in R^m$ and $b_e^{(k)} \in R$ are layer specific parameters, and σ is a non-linear activate function.

According to [37], we integrated edge-wise gating mechanism into the graph convolution network, the final propagation function is:

$$h_i^{(k+1)} = \frac{1}{d(i)} \sum_{j \in N(i)} W_{ij} g_j^{(k)} (w_g^{(k)} h_j^{(k)} + b_g^{(k)}) \tag{15}$$

when $k = 0$, $h_j^{(0)} = v_j$ is the input of GCN, i.e., the word original representation. For a n-layer GCN, $h_i^{(n)}$ is the word final representation after the non-local information adaptation.

4.2. Embedding Layer

External knowledge, such as entity positions and dependency trees, is important for relation extraction [45–47]. Furthermore, when adapting a model from a source domain to a target domain, external knowledge can be seen as general knowledge that can improve cross-domain task performance. Following previous works [15,28,33], we use the following five types of external knowledge:

Real-valued word embedding vector. We obtain one word's embedding vector e_i from the word embedding matrix, which is pretrained as in [41]. This process yields continuous vector representations of words by training the CBOW or skim-gram model on very large data sets, and the vectors include the words' semantic information. The words that do not exist in the word embedding matrix are randomly initialized.

Words' relative distances from candidate entities. Use i and j denote the two entities' in a sentence, for each word x_k with index k, its relative distances are $k - i$ and $k - j$, respectively. The word's relative distances can inform the model of the entities' positions. Every word has two relative distances vectors d_1 and d_2.

Entity type. Each entity type is predefined, and every entity has an entity type to which it belongs. In the sentence "*He will blow* ***a city*** *off* ***the earth*** *in a minute if he can get the hold of the means to do it*", the bold words are candidate entities, and their entity types are GPE and LOC. Entity types are essential knowledge for relation extraction. In some cases, we can infer the relation of two candidate entities only by the entity types. In our setting, we only indicate that the entity types and nonentity words are randomly initialized in the same vector. Every word has two entity type vectors t_1 and t_2 because we have two candidate entities per sentence.

Sematic chunks. A chunk is an indivisible fixed phrase in a sentence, and we inform the model to regard all chunks as a whole so that the semantic information of the chunks is not disrupted. We use the B-I-O format to indicate chunks, and every word obtains a chunk vector c_i.

Shortest dependency path between two entities. The shortest dependency path refers to the shortest path between two entities in the dependency tree. See Figure 5 for an example. In relation extraction, the information required to assert a relationship between two entities is mostly captured by the words in the shortest dependency path between the two entities [45]. Therefore, the shortest dependency path can help the model to distinguish between valuable information and noise. We use a vector d_i to indicate whether a word is in the shortest dependency path between two entities.

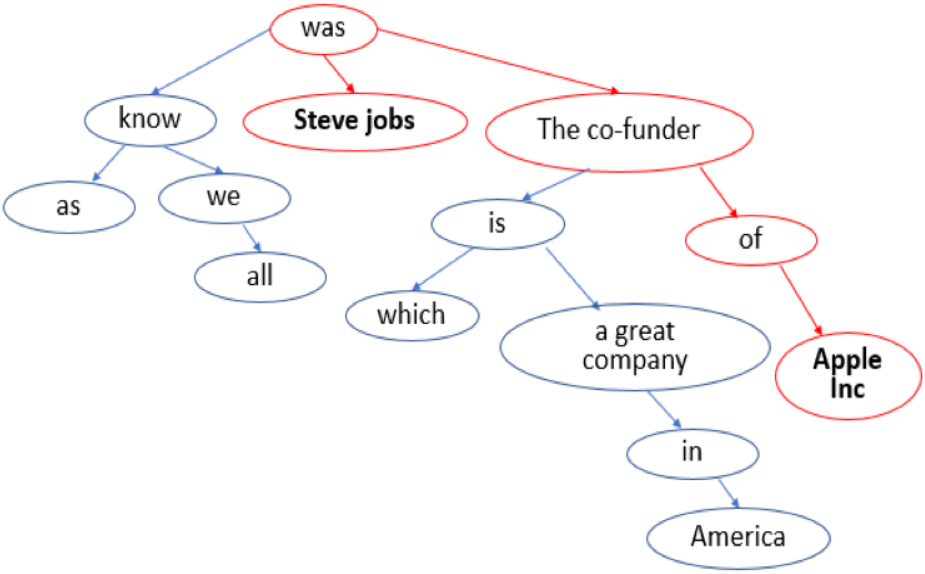


Figure 5. Dependency tree of the sentence "*As we all know, Steve Jobs was the co-founder of Apple Inc. which is a great company in America*". Bold words are two candidate entities, red lines indicates the shortest dependency path between the two entities.

After getting all above types of embedding vectors, we transform every word into a real-valued vector v_i by concatenating them: $v_i = [gcn(e_i); p_{i1}; p_{i2}; t_{i1}; t_{i2}; c_i; d_i]$. The $gcn(\cdot)$ is the transformation of GCN layer described in Section 4.1, see Figure 3 for details. The whole sentence with length n can be represented as $v = [v_1, v_2, \ldots, v_n]$.

4.3. Shared Feature Extractor

We use a simple CNN architecture proposed by [48] as shared feature extractor. Let $v_i \in R^d$, and a convolution operation with a kernel $w \in R^{rd}$ is applied to v, where r is numbers of words the kernel spanned and v is the word representation matrix after GCN layer. A feature c_i is generated from words $v_{i:i+r-1}$:

$$c_i = ReLU(wv_{i:i+r-1} + b) \tag{16}$$

Here, $ReLU$ is the activation function and $b \in R$ is a bias. The kernel moves one step at a time in the direction of word sequences and get $n - h + 1$ features totally:

$$c = [c_1, c_2, ..., c_{n-r+1}] \tag{17}$$

Then we perform max-over-time pooling operation [49] on c, i.e., $c^{'} = max\{c\}$, to get the most important feature. To capture various features, we use multiple kernel size (keep d unchanged and use different r) and each kernel size has multiple feature maps. For a fixed kernel size, different feature maps will get different $c_i^{'}$, so the feature vector $c_i^{'}$ corresponding to one kernel size i is:

$$c_i^{'} = [c_1^{'}, c_2^{'}, \ldots, c_m^{'}] \tag{18}$$

m is the number of feature maps. Finally, we concatenate all $c_i^{'}$ to get shared feature extractor's output $s = [c_1^{'}; c_2^{'}, \ldots, c_k^{'}]$, k is the number of kernel size.

4.4. Relation Classifier

Since there only exists labels in the source domain, the relation classifier only takes the shared features s_s from source domain as input to perform relation classification. The relation classifier is a 2-layer fully connected neural network h with $tanh$ as activation function and followed by a $softmax$ layer:

$$p_i = softmax(h(s_s; \theta_s)) \tag{19}$$

where θ_s is the parameters of the hidden layer. $p_i \in R^r$ is the relation distribution of i-th source domain data, and r is the number of relation type. The relation classify loss L_{rel} is defined as below:

$$L_{rel} = -\frac{1}{N_s} \sum_{i=1}^{N_s} \sum_{j=1}^{r} y_{ij} log p_{ij} \tag{20}$$

Here N_s is the total number of source domain data, $y_{ij} \in [0, 1]$ to indicate whether the example i has relation j. p_{ij} is obtained through $softmax$ layer and indicates the probability of example i containing the relation j.

During training, we combine all the losses mentioned above to get the final loss function and optimize jointly:

$$L_{loss} = L_{rel} + \gamma L_{dom} \tag{21}$$

γ is a hyparameter and we set $\gamma = 0.1$ through validation dataset. In the test stage, due to lacking of source domain data, a heuristic algorithm is designed to select domain-specific words of source domain. First, for every shared word w_i appearing in one target domain sentence, we find the source domain-specific word w_j which have the highest PMI value with w_i. All the w_j consist of the top-pmi set $w = \{w_1, w_2, \ldots, w_n\}$ which n is the number of shared word appeared in the target domain sentence. Then we sort the elements of w in descending order. Finally, the top m words are selected to form the source domain sentence and we set $m = 10$ through the performance in validation dataset.

5. Experiments

5.1. Dataset and Evaluation

(1) Dataset

We conduct our experiment on the English part of the ACE2005 dataset. There exist 6 domains (broadcast news (bn), broadcast conversations (bc), newswire (nw), weblog (wl), usenet (un), and conversational telephone speech (cts)) and 7 unidirectional relations (artefact (ART), gen-affiliation (GEN-AFF), org-affiliation (ORG-AFF), part-whole (PART-WHOLE), person-social (PER-SOC), physical (PHYS) and none (METONYMY)). If we consider the directions of the entity pairs, there are 11 directional relations (METONYMY, PER-SOC and PHYS are symmetric relations). We process the datasets as in [28] (Table 1).

From Table 1, we can see that negative examples account for a large proportion of the data, so correctly distinguishing between positive and negative examples is an important indicator for measuring the effect of the model. Figure 6 explicitly displays the differences between domains in terms of their data distributions. These large differences also pose a challenge for our model. Following previous works [15,16,33], we use bn+nw as the source domain, adjust the hyperparameter on half of the bc domain, and use the remaining half of bc, as well as all of cts and wl, as the target domain to evaluate the performance of our model.

Table 1. ACE2005 dataset statistics.

Domain	Total	#Entity Type	Negative Relation Rate
bn+nw	43,497	43	0.916
bc_dev	7004	33	0.912
bc_test	8083	33	0.911
cts	15,903	40	0.961
wl	13,882	30	0.949

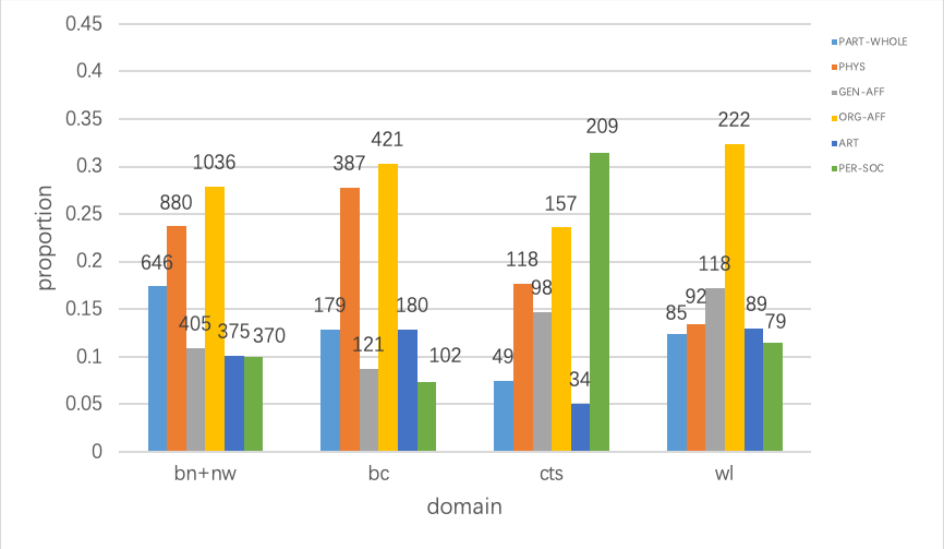


Figure 6. Relation distributions of different domains. Note that we ignore the NONE type because it accounts for a large portion in all three domains.

(2) Evaluation

The precision, recall and macro-F1 are used as evaluation method in our experiment. Specifically, we calculate precision (P_i) and recall (R_i) for every relation type, and get all relation precision (P) and recall (R) by averaging:

$$P = \frac{1}{r}\sum_{i=1}^{r} n_i P_i \tag{22}$$

$$R = \frac{1}{r}\sum_{i=1}^{r} n_i R_i \tag{23}$$

where r is the number of relation type and n_i is the number of samples belong to i-th relation. The macro-F1 is calculated using:

$$\text{macro-F1} = 2 \times \frac{P \times R}{P + R} \tag{24}$$

5.2. Parameter Setting

We use pretrained, 300-dimensional word embeddings generated by word2vec [11]. The dependency entity positions (eps) are obtained from *ace-data-prep* (https://github.com/mgormley/ace-data-prep). The embeddings are randomly initialized and optimized during training except for these pretrained word embeddings because we found that there is no improvement when pretraining all embeddings. All the sentences are padded or cut to 155 characters. The learning rate is set to 0.001 and halved every 4 epochs. We use Adam [50] as the optimizer and apply gradient clipping during optimization. To avoid overfitting,

the dropout technique [51] is used in the embedding layer and GCN layer. The details of the parameter settings are shown in Table 2.

Table 2. Parameters settings of our model. *dd* and *rc* refer to the domain discriminator and relation classifier, respectively.

Parameters	Settings
Word embedding dimensions	300
Dt/ch/et/ep embedding dimensions	50
kernel size	[2, 3, 4, 5]
Number of kernels	150
Hidden dimension of dd, rc	300
Hidden dimension of GCN layer	100
Learning rate	0.001
Dropout rate	0.5
Batch size	128
Interval of gradient clipping	[−1, 1]
Windows size of words co-occrrences	[20]

5.3. Baseline Models

We use the following baseline models for evaluation purposes:

NNM & log-linear model: Basic neural network models (NNM), such as CNNs and RNNs, were used in [29] to improve relation extraction. In addition, the authors achieved state-of-the-art performance by stacking these models. We use their single model bidirectional RNN (BRNN), CNN, log-linear model, and combined model (called hybrid-voting system (HVS)) as our baseline models.

FCM & hybrid FCM: The feature-rich compositional embedding model (FCM) was proposed in [28]. The key idea is to combine (unlexicalized) hand-crafted features with learned word embeddings. The hybrid FCM (HFCM) combines the basic FCM and existing log-linear models.

LRFCM: The low-rank approximation of the FCM (LRFCM) [13] is an improvement of the FCM. It replaces manual features with feature embeddings so that it can easily scale to a large number of features.

DANN: The domain adversarial neural network (DANN) [15] was the first to introduce adversarial training to cross-domain relation extraction. It simply projects source domain features and target domain features into one unified space and uses adversarial training to extract domain-independent features. We propose a graph adaptation network based on this model.

GSN: The genre separate network (GSN) [16] uses a domain separate network [17] to extract domain-independent features and domain-specific features separately, therefore avoiding the introduction of some domain-specific features into the shared feature space.

CVAN: The cross-view adaptation network (CVAN) [33] uses cross-view training to extract shared features from different views and constructs various input views that have proven to be useful for cross-domain relation extraction.

AGGCN: The attention guided graph convolutional networks (AGGCN) [52], a novel model which directly takes full dependency trees as inputs.This model can be understood as a soft-pruning approach that automatically learns how to selectively attend to the relevant sub-structures useful for the relation extraction task.

MAPDA: A novel model based on a multi-adversarial module for partial domain adaptation (MAPDA) is proposed in this study [10]. This paper design a weight mechanism to mitigate the impact of noise samples and outlier categories, and embed several adversarial networks to realize various category alignments between domains.

5.4. Results Analysis

(1) Performance comparison with existing methods.

Table 3 provides a comparison between existing models and our method. Note that the models marked $*$ are reimplemented version because their precision and recall values have not been reported. The model marked $+$ denotes that it contains multiple models, and we only report the best results among them. The results of our models are obtained under the parameter settings that achieve the best results for the development dataset (GCN layers = 3, α = 0.4). From the table, we can see that a model removing adversarial training and only using fixed weights (i.e., CNN+GCN) outperforms the CNN by 2% in terms of macro-F1 score and obtains results that are comparable to those of the DANN. This means that local feature adaptation and non-local feature adaptation are equally important for cross-domain relation extraction.

Table 3. Precision (P), recall (R) and macro-F1 (%) comparison with existing methods on three domains.

Models	bc			cts			wl		
	P	**R**	**F1**	**P**	**R**	**F1**	**P**	**R**	**F1**
BRNN	65.23	61.06	63.07	66.15	49.26	56.47	55.91	51.56	53.65
CNN	65.62	61.06	63.26	65.92	48.12	55.63	54.14	53.68	53.91
Log-linear	68.44	50.07	57.83	73.62	41.57	53.14	54.14	47.31	53.06
HVS $^{+}$	70.40	63.84	66.96	65.91	52.21	58.26	58.81	55.81	57.27
FCM	66.56	57.86	61.9	65.62	44.35	52.93	57.80	44.62	50.36
HFCM	74.39	55.35	63.48	74.53	45.01	56.12	65.63	47.59	55.17
LRFCM	65.1	54.7	59.4	-	-	-	-	-	-
GSN *	66.78	64.12	65.42	68.89	48.71	57.07	59.32	52.91	55.93
CVAN	73.95	60.92	66.81	70.43	53.61	60.18	59.2	56.13	57.62
DANN *	70.21	59.36	64.33	71.43	47.72	57.2	58.37	54.26	56.24
AGGCN	–	–	63.47	–	–	59.70	–	–	56.50
MADA-weight	–	–	65.86	–	–	56.33	–	–	56.10
CNN+GCN	69.82	61.46	65.37	64.57	52.63	57.99	62.59	50.67	56
DANN+GCN	75.12	60.46	67	64.67	57.35	60.79	60.52	54.8	57.52
DANN+GCN^2	70.56	63.28	66.72	60.08	58.57	59.32	63.12	53.08	57.67
DANN+GCN+DA	72.55	64.31	68.18	65.4	59.1	62.1	65.93	53.17	58.87

Model marked with $^{+}$ contains multiple models, and only the best results among them are reported. Models marked with * present reimplemented versions, as their precision and recall values have not been reported.

After adding adversarial training, DANN+CNN achieves results that are comparable to those of the state-of-the-art model (CVAN) in terms of macro-F1. DANN + GCN^2 indicates the model that uses only dynamic attention weights; this also improves the baseline DANN model but only achieves similar results to those of DANN+GCN. We assume that dynamic weights cannot capture all word co-occurrence information due to a lack of global statistical knowledge. The ensemble model (HVS) performs the best among the methods from previous works, especially in the bc domain. To illustrate the performance of our model more convincingly, we also compare it with the ensemble model. Our combined model (DANN+GCN+DA) outperforms all the existing models, including the ensemble model (HVS), in terms of macro-F1, and achieves almost all of the best precision and recall performance in the three domains.

(2) Data distribution of source and target domains.

In this paper, the source domain dataset bn+nw and the target domain dataset cts are taken as examples. To see the change in data distribution more intuitively, we used the high-dimensional data dimensionality reduction algorithm t-distributed stochastic neighbor embedding (t-SNE) to map the data distribution in two-dimensional space. The data distribution before and after applying our method is shown in Figure 7. As can be seen from Figure 7a,b, after using graph adaptation network,

the cts overlapping data is effectively reduced and data distribution is better than the previous data distribution. The t-SNE visualization also indirectly proves the effectiveness of our model.

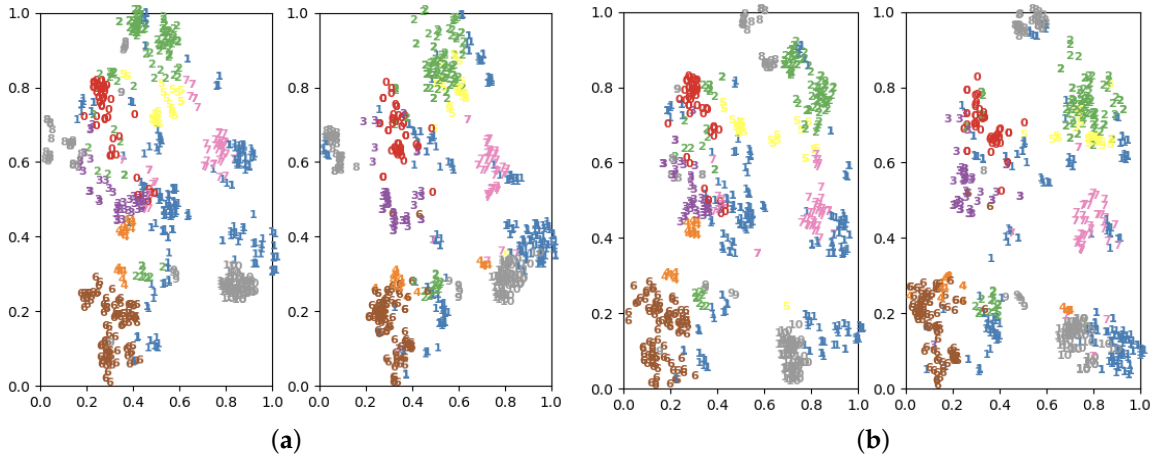


Figure 7. Data distribution under different relation types. (**a**) Original data distribution of source domain (left) and target domain (right). (**b**) Data distribution of source domain (left) and target domain (right) under our method.

(3) Performances with different numbers of GCN layers.

From Equation (14), we know that the number of GCN layers plays an important role in the process of information propagation. In a 1-layer GCN, information only flows between neighbors, so for our tripartite graph, the number of GCN layers is at least 2 so that the information can flow from the source to the target domain. To better verify this intuition and further illustrate the interpretability of our model, we fix the threshold (α) described in Equation (3) as 0.4 and draw Figure 8 to display the relation between the number of GCN layers and the performance (macro-F1) of our model. We report the macro-F1 values for three domains under different numbers of GCN layers L within [1, 2, 3, 4, 5, 6]. It is worth noting that we only draw the line when $L \leq 6$ because there is no improvement obtained when $L > 6$.

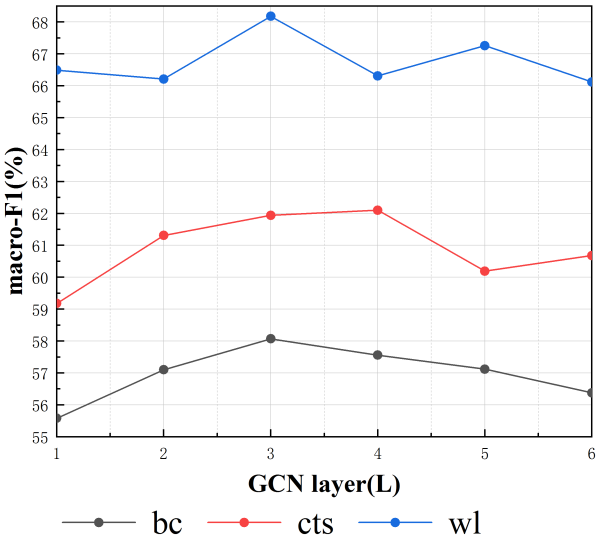


Figure 8. Macro-F1 values with different numbers of GCN layer.

From Figure 8, we can see that the 1-layer GCN has a relatively poor performance on all three domains, but when the number of layers is 3 or 4, the best performance is achieved for all three

domains; this illustrates that the 1-layer GCN cannot transfer information from the source domain to the target domain and that a GCN with at least 2 layers can capture the word co-occurrences between the source and target domains. When $L > 4$, the macro-F1 declines to different degrees for all three domains, and we assume that the word representations are distorted due to too many instances of information propagation.

(4) Performances with different thresholds (α).

The threshold (α) controls the number of edges whose fixed weights >0. Because the fixed weights of edges are normalized by Equation (6), α varies within the interval [0,1]. From Table 4, we can see that when α is smaller, there are more edges with fixed weights >0 in the graph and vice versa. In particular, when $\alpha = 1$, there only exist self-loop edges whose fixed weights >0. Figure 9 shows the macro-F1 values with different values of α under the setting "GCN layers = 3". We can clearly see that when α decreases, macro-F1 suffers from a significant drop. The reason for this phenomenon is that the number of edges with fixed weights >0 increases sharply and therefore inevitably includes more irrelevant information.

Table 4. The total and average number of edges whose fixed weights >0 under different values of α. The total number of edges is calculated based on the combination of all source and target domain data. The average edge number indicates the average number of edges in one sentence. None of the statistics include self-loop edges.

α	#Total Edge Number (Average Edge Number)		
	bc	**cts**	**wl**
0	392,552 (348)	401,541 (360)	454,692 (321)
0.1	288,202 (172)	290,678 (241)	344,694 (221)
0.2	191,256 (129)	191,858 (154)	242,238 (144)
0.3	110,734 (78)	106,068 (85)	150,442 (80)
0.4	48,838 (30)	45,870 (32)	73,212 (51)
0.5	15,352 (10)	14,342 (9)	25,184 (19)
0.6	4220 (4)	3952 (3)	6632 (4)
0.7	1012 (2)	954 (2)	1602 (2)
0.8	262 (1)	272 (2)	344 (1)
0.9	6 (0.05)	44 (0.3)	18 (0.2)
1.0	0 (0)	0 (0)	0 (0)

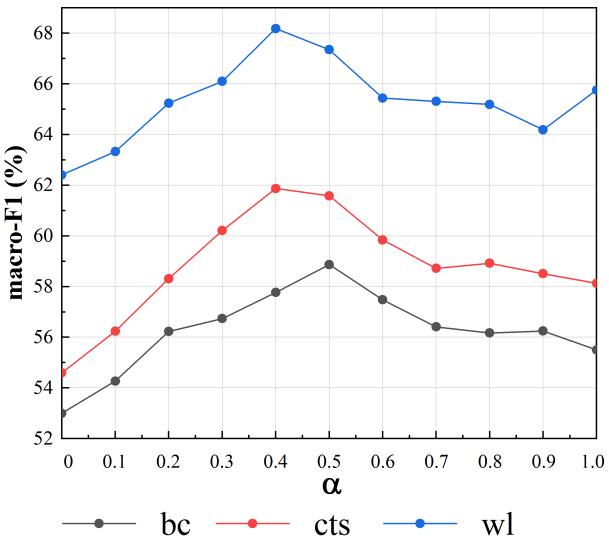


Figure 9. Macro-F1 values with different values of α.

The model obtains the best results when α = 0.4 or 0.5, for which the corresponding edge numbers are optimal. When α > 0.5, the performance on all three domains suffers from dramatic decreases. We argue that this is because almost all edges' fixed weights = 0, and dynamic weights cannot capture complete word co-occurrence information due to a lack of global statistical knowledge. When α is in the interval [0.6, 1], the macro-F1 value remains stable because the graph has a few edges with fixed weights >0. The model performance under different values of α illustrates that simply increasing the number of edges indefinitely is not ideal. The use of a proper number of edges can not only yield better performance but also speed up calculations.

(5) Effects of the GCN and dynamic weights.

To illustrate the effectiveness of the GCN and the use of dynamic attention weights, we draw the precision-recall curves for every relation type in Figure 10. The values of precision and recall are averaged across all domains. CNN+adv is the reimplemented version of the DANN [11], +cv refers to adding cross-view training, and DA refers to the dynamic attention weights we proposed. From Figure 10, we can see the following: (1) Only using the CNN to perform cross-domain relation extraction is far from enough, and it almost achieves the worst performance for all 6 relations, indicating that it is worth trying domain adaptation on this dataset. (2) The performance of CNN+adv and CNN+adv+cv are better than that obtained when only using the CNN, but CNN+adv+cv is significantly worse than the CNN under the relation GAN-AFF, and CNN+adv is also worse than the CNN under the relation ORG-AFF. We find that the GAN-AFF and ORG-AFF relations are similar and have the most subtypes, so these two relations provide a strong challenge for our model. (3) CNN+adv+GCN yields large performance increases for GAN-AFF and ORG-AFF, and the curves are almost all above the curve of CNN+adv.

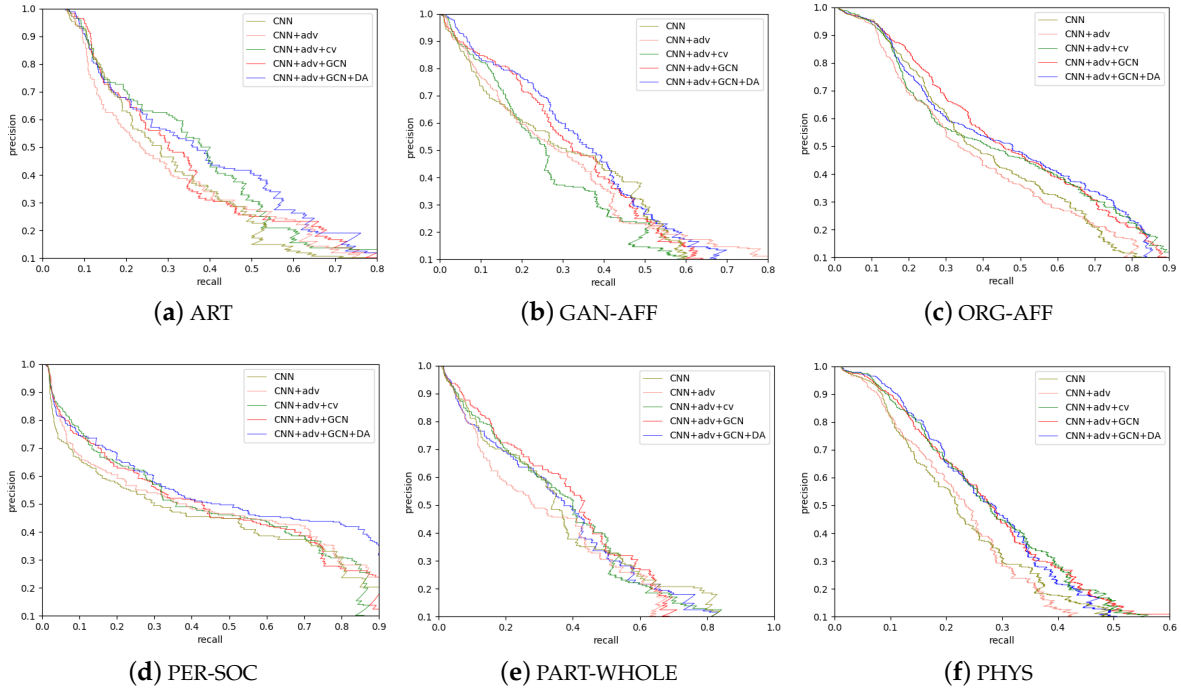
This illustrates the effects of the GCN layer. When adding DA to the GCN layer, the improvement is highly obvious, especially under the PER-SOC relation; specifically, there is a 10% improvement in precision when recall >0.7 in Figure 10e. This verifies that DA can compensate for the weakness of fixed weights and consider many words' connections. For a quantitative analysis of the effects of each model, we average the AUCs under different relations for every method, and these are shown in Table 5. It can be clearly seen that GCN+DA improves the baseline by a large margin (by 4% compared with CNN+adv, by 7% compared with the CNN).

Table 5. AUCs of different methods.

Models	**CNN**	**CNN + adv**	**CNN + adv + cv**	**CNN + adv + GCN**	**CNN + adv + GCN + DA**
AUC	0.345	0.378	0.385	0.397	0.41

5.5. Case Study

An example of graph weight visualization is shown in Figure 11. The x-axis denotes the shared words of different domain sentences, the red part of the y-axis denotes source-specific words, and the blue part denotes target-specific words. Each pixel corresponds to the weight W_{ij} of the i-th shared word and the j-th specific word described in Equation (11), and the deeper the colour is, the greater the W_{ij}. All the weights corresponding to the i-th shared word are normalized. The colors between the word *new* and some target-specific words such as *Mexico* and *Kansas* are deepest because their fixed weights are large according to Equation (7). In addition, some specific words with strong domain relevance, such as *Funeral* and *School*, also have deep colors, while the colors of words with weak domain relevance, such as *functions* and *apointed*, are relatively shallow. These facts illustrate that the dynamic weight mechanism pays more attention to aligning specific words with stronger domain relevance; the sole use of fixed weights cannot be achieved.



(**a**) ART (**b**) GAN-AFF (**c**) ORG-AFF

(**d**) PER-SOC (**e**) PART-WHOLE (**f**) PHYS

Figure 10. Precision/recall curves under different relation types.

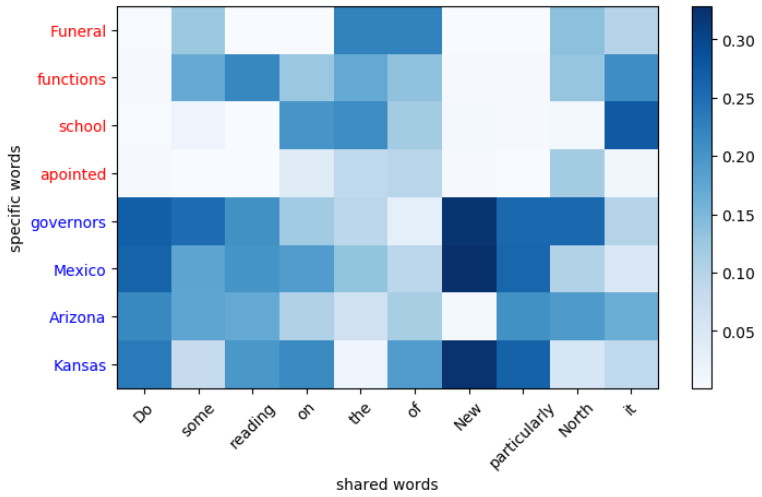


Figure 11. Graph edge weights visualization.

We also present some typical examples in Table 6 that demonstrate the effectiveness of our model. In some cases, such as the first two samples, traditional models and our model can all correctly predict the labels. However, when the domain-specific words have strong domain relevance or account for a high proportion of the sentence, traditional models mistake all the labels for "None" (as in the last three examples) because negative samples account for a large proportion (Table 1). Our model correctly distinguishes between negative and positive samples and predicts true labels in all the above cases. We argue that traditional models only capture shared features based on local information, i.e., words in sequential order; therefore, when the proportion of domain-specific words increases, the number of shared features decreases. In addition, the shared feature extractor may be unable to capture shared features because these specific words have strong domain relevance; in other words, the domain discriminator still has the ability to determine which domain a given sample comes from. Through the GCN layer we proposed, these target-specific words are aligned with source-specific words, so the interference of domain-specific features is reduced.

Table 6. Some predictions of typical samples comparison. We use DANN and CVAN as traditional models. The words with bold are domain-specific.

Predict Label		Examples
Traditional	**Ours**	
ART √	ART √	In which, the state of Israel **buys** many **expensive** military weapons, which is used to **oppress** and to kill a lot of Palestinians.
ORG-AFF √	ORG-AFF √	It could be done so **easily** because most small **towns** are protected by a small town police force.
None ×	PHYS √	well, my sister usually comes in from **Ohio** because it's not that far like–two hundred **fifty** miles or something, and she'll come in for **Thanksgiving**.
None ×	GAN-AFF √	That's in **Fairmont**, West **Virginia**, it's like—oh, between **Charleston** and **Pittsburgh**.
None ×	PER-SOC √	My **brother-in-law** still lives in the city but we were **Long Island** people.

6. Conclusions

In this article, a novel graph adaptation network for cross-domain relation extraction is proposed. First, a novel graph adaptation network is constructed to align domain-specific features. The model aligns domain-specific features through applying graph convolutions on a source-shared-target words tripartite graph. Secondly, unlike the traditional methods only adapting local or sequential features, our model adapts local and non-local features jointly to improve the performance of cross-domain relation extraction. Finally, a cross-domain relation extraction model is constructed by inputting the fused all features into softmax.

In addition, unlike the traditional graph convolutional network, in order to compensate for the limitation of fixed weight, a dynamic attention weight is combined together with fixed weight. To transmit useful information more effectively, we retain valuable edges based on their edge weights. Experiments on the three domains of ACE2005 datasets verified the effectiveness of GCN layer and dynamic attention weight, which achieve state-of-the-art on all three domains.

We will further explore the cross-domain relation extraction when a few labeled examples can be explored or new relations appearing in the target domain. In addition, we hope our work can be applied to other domain adaptation tasks in the future.

Author Contributions: Conceptualization, Z.W. and B.Y.; data curation, B.Y.; methodology, Z.W. and B.Y.; validation, B.Y.; writing, original draft, Z.W. and B.Y.; writing, review and editing, Z.W.; project administration, C.W., B.W. and K.Z.; funding acquisition, X.W. and K.Z. All authors read and agreed to the published version of the manuscript.

Funding: This research was funded by the National Key R & D Program of China Grant Numbers 2017YFB0802800 and Beijing Natural Science Foundation (4202002).

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

References

1. Sadique, K.M.; Rahmani, R.; Johannesson, P. Towards Security on Internet of Things: Applications and Challenges in Technology. *Procedia Comput. Sci.* **2018**, *141*, 199–206. [CrossRef]
2. Miorandi, D.; Sicari, S.; De Pellegrini, F.; Chlamtac, I. Internet of things: Vision, applications and research challenges. *Procedia Comput. Sci.* **2012**, *10*, 11497–11516. [CrossRef]
3. Macedo, E.L.C.; de Oliveira, E.A.R.; Silva, F.H.; Mello, R.R.; França, F.M.G.; Delicato, F.C.; de Rezende, J.F.; de Moraes, L.F.M. On the security aspects of Internet of Things: A systematic literature review. *J. Commun. Netw.* **2019**, *21*, 444–457. [CrossRef]

4. Bandyopadhyay, D.; Sen, J. Internet of Things: Applications and Challenges in Technology and Standardization. *Wirel. Pers. Commun.* **2011**, *58*, 149–169. [CrossRef]
5. Ray, P.P. A survey on Internet of Things architectures. *J. King Saud Univ. Comput. Inf. Sci.* **2018**, *30*, 291–319.
6. Ali, F.; Kwak, D.; Khan, P.; Islam, S.M.R.; Kim, K.H.; Kwak, K.S. Fuzzy ontology-based sentiment analysis of transportation and city feature reviews for safe traveling. *Transp. Res. Part C Emerg. Technol.* **2017**, *77*, 33–48. [CrossRef]
7. Whaiduzzaman, M.; Sookhak, M.; Gani, A.; Buyya, R. A survey on vehicular cloud computing. *J. Netw. Comput. Appl.* **2014**, *40*, 325–344. [CrossRef]
8. Ali, F.; Kwak, D.; Islam, S.M.R.; Kim, K.H.; Kwak, K.S. Fuzzy Domain Ontology-based Opinion Mining for Transportation Network Monitoring and City Features Map. *J. Korea Inst. Intell. Transp. Syst.* **2016**, *15*, 109–118. [CrossRef]
9. Ali, F.; El-Sappagh, S.; Kwak, D. Fuzzy Ontology and LSTM-Based Text Mining: A Transportation Network Monitoring System for Assisting Travel. *Sensors* **2019**, *19*, 234. [CrossRef]
10. Cao, X.; Yang, J.; Meng, X. Partial Domain Adaptation for Relation Extraction Based on Adversarial Learning. In Proceedings of the European Semantic Web Conference (ESWC 2020), Online Conference, 2–4 June 2020; pp. 89–104.
11. Plank, B.; Moschitti, A. Embedding Semantic Similarity in Tree Kernels for Domain Adaptation of Relation Extractione. In Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (ACL 2013), Sofia, Bulgaria, 4–9 August 2013; Volume 1, pp. 1498–1507.
12. Nguyen, T.H.; Grishman, R. Employing word representations and regularization for domain adaptation of relation extraction. In Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (ACL 2014), Baltimore, MD, USA, 22–27 June 2014; pp. 68–74.
13. Yu, M.; Gormley, M.R.; Dredze, M. Combining Word Embeddings and Feature Embeddings for Fine-grained Relation Extraction. In Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Denver, CO, USA, 31 May–5 June 2015; pp. 1374–1379.
14. Rios, A.; Kavuluru, R.; Lu, Z. Generalizing biomedical relation classification with neural adversarial domain adaptation. *Bioinformatics* **2018**, *3417*, 12973–12981. [CrossRef]
15. Fu, L.; Nguyen, T.H.; Min, B.; Grishman, R. Domain Adaptation for Relation Extraction with Domain Adversarial Neural Network. In Proceedings of the 8th International Joint Conference on Natural Language Processing, Taipei, Taiwan, 27 November–1 December 2017; Volume 2, pp. 425–429.
16. Shi, G.; Feng, C.; Huang, L.; Zhang, B.; Ji, H.; Liao, L.; Huang, H. Genre Separation Network with Adversarial Training for Cross-genre Relation Extraction. In Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, 31 October–4 November 2018; pp. 1018–1023.
17. Bousmalis, K.; Trigeorgis, G.; Silberman, N.; Krishnan, D.; Erhan, D. Domain Separation Networks. In Proceedings of the 30th Conference on Neural Information Processing Systems (NIPS 2016), Barcelona, Spain, 5–10 December 2016; pp. 343–351.
18. Tzeng, E.; Hoffman, J.; Zhang, N.; Saenko, K.; Darrell, T. Deep Domain Confusion: Maximizing for Domain Invariance. *arXiv* **2014**, arXiv:1412.3474.
19. Long, M.; Cao, Y.; Wang, J.; Jordan, M. Learning Transferable Features with Deep Adaptation Networks. In Proceedings of the 32nd International Conference on International Conference on Machine Learning, Lille, France, 7–9 July 2015; Volume 37, pp. 97–105.
20. Pan, S.J.; Ni, X.; Sun, J.-T.; Yang, Q.; Chen, Z. Cross-domain Sentiment Classification via Spectral Feature Alignment. In Proceedings of the 19th International Conference on World Wide Web, Raleigh, NC, USA, 26–30 April 2010; pp. 751–760.
21. Zhang, Y.; Qi, P.; Manning, C.D. Graph Convolution over Pruned Dependency Trees Improves Relation Extraction. In Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, 31 October–4 November 2018; pp. 2205–2215.
22. Fu, T.-J.; Li, P.-H.; Ma, W.-Y. GraphRel: Modeling Text as Relational Graphs for Joint Entity and Relation Extraction. In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, Florence, Italy, 28 July–2 August 2019; pp. 1409–1418.
23. Song, J.; Liu, Y.; Shao, J.; Tang, C. A Dynamic Membership Data Aggregation (DMDA) Protocol for Smart Grid. *IEEE Syst. J.* **2020**, *14*, 900–908. [CrossRef]

24. Lei, Y.; Lin, J.; He, Z.; Kong, D. A method based on multi-sensor data fusion for fault detection of planetary gearboxes. *Sensors* **2012**, *12*, 2005–2017. [CrossRef] [PubMed]
25. Safizadeh, M.S.; Latifi, S.K. Using multi-sensor data fusion for vibration fault diagnosis of rolling element bearings by accelerometer and load cell. *Inf. Fusion* **2014**, *18*, 1–8. [CrossRef]
26. Jing, L.; Wang, T.; Zhao, M.; Wang, P. An Adaptive Multi-Sensor Data Fusion Method Based on Deep Convolutional Neural Networks for Fault Diagnosis of Planetary Gearbox. *Sensors* **2017**, *17*, 414. [CrossRef]
27. Schmidt, P.; Reiss, A.; Dürichen, R.; Laerhoven, K.V. Wearable-Based Affect Recognition—A Review. *Sensors* **2019**, *19*, 4079. [CrossRef]
28. Gormley, M.R.; Yu, M.; Dredze, M. Improved Relation Extraction with Feature-Rich Compositional Embedding Models. In Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, Lisbon, Portugal, 17–21 September 2015; pp. 1774–1784.
29. Nguyen, T.H.; Grishman, R. Combining Neural Networks and Log-linear Models to Improve Relation Extraction. *arXiv* **2015**, arXiv:1511.05926.
30. Ganin, Y.; Lempitsky, V. Unsupervised Domain Adaptation by Backpropagation. In Proceedings of the 32nd International Conference on International Conference on Machine Learning, Lille, France, 6–11 July 2015; Volume 37, pp. 1180–1189.
31. Yan, B.; Zhang, D.; Wang, H.; Wu, C. Cross-View Adaptation Network for Cross-Domain Relation Extraction. In Proceedings of the China National Conference on Chinese Computational Linguistics (CCL 2019), Kunming, China, 18–20 October 2019; pp. 306–317.
32. Clark, K.; Luong, M.T.; Manning, C.D.; Le, Q.V. Semi-Supervised Sequence Modeling with Cross-View Training. *arXiv* **2018**, arXiv:1809.08370.
33. Peng, M.; Zhang, Q.; Jiang, Y.-G.; Huang, X. Cross-Domain Sentiment Classification with Target Domain Specific Information. In Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, Melbourne, Australia, 15–20 July 2018; Volume 1, pp. 2505–2513.
34. Cui, W.; Zheng, G.; Shen, Z.; Jiang, S.; Wang, W. Transfer Learning for Sequences via Learning to Collocate. *arXiv* **2019**, arXiv:1902.09092.
35. Qian, Y.; Santus, E.; Jin, Z.; Guo, J.; Barzilay, R. GraphIE: A Graph-Based Framework for Information Extraction. *arXiv* **2018**, arXiv:1810.13083.
36. Sun, C.; Gong, Y.; Wu, Y.; Gong, M.; Jiang, D.; Lan, M.; Sun, S.; Duan, N. Joint Type Inference on Entities and Relations via Graph Convolutional Networks. In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, Florence, Italy, 28 July–2 August 2019; pp. 1361–1370.
37. Vashishth, S.; Bhandari, M.; Yadav, P.; Rai, P.; Bhattacharyya, C.; Talukdar, P. Incorporating Syntactic and Semantic Information in Word Embeddings using Graph Convolutional Networks. In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, Florence, Italy, 28 July–2 August 2019; pp. 3308–3318.
38. Zhu, H.; Lin, Y.; Liu, Z.; Fu, J.; Chua, T.-S.; Sun, M. Graph Neural Networks with Generated Parameters for Relation Extraction. In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, Florence, Italy, 28 July–2 August 2019; pp. 1331–1339.
39. Veličković, P.; Cucurull, G.; Casanova, A.; Romero, A.; Liò, P.; Bengio, Y. Graph Attention Networks. *arXiv* **2017**, arXiv:1710.10903.
40. Thekumparampil, K.K.; Wang, C.; Oh, S.; Li, L.-J. Attention-based Graph Neural Network for Semi-supervised Learning. *arXiv* **2018**, arXiv:1803.03735.
41. Mikolov, T.; Chen, K.; Corrado, G.S.; Dean, J. Efficient Estimation of Word Representations in Vector Space. *arXiv* **2013**, arXiv:1301.37811.
42. Pennington, J.; Socher, R.; Manning, C.D. GloVe: Global Vectors for Word Representation. In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), Doha, Qatar, 25–29 October 2014; pp. 1532–1543.
43. Kipf, T.N.; Welling, M. Semi-Supervised Classification with Graph Convolutional Networks. *arXiv* **2016**, arXiv:1609.02907.
44. Marcheggiani, D.; Titov, I. Encoding Sentences with Graph Convolutional Networks for Semantic Role Labeling. In Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, Copenhagen, Denmark, 9–11 September 2017; pp. 1506–1515.

45. Bunescu, R.; Mooney, R. A Shortest Path Dependency Kernel for Relation Extraction. In Proceedings of the Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing, Vancouver, BC, Canada, 6–8 October 2005; pp. 724–731.
46. Nguyen, T.H.; Plank, B.; Grishman, R. Semantic Representations for Domain Adaptation: A Case Study on the Tree Kernel-based Method for Relation Extraction. In Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing, Beijing, China, 26–31 July 2015; Volume 1, pp. 635–644.
47. Sun, A.; Grishman, R.; Sekine, S. Semi-supervised Relation Extraction with Large-scale Word Clustering. In Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, Portland, OR, USA, 19–24 June 2011; pp. 521–529.
48. Kim, Y. Convolutional Neural Networks for Sentence Classification. *arXiv* **2014**, arXiv:1408.5882.
49. Collobert, R.; Weston, J.; Bottou, L.; Karlen, M.; Kavukcuoglu, K.; Kuksa, P. Natural Language Processing (almost) from Scratch. *arXiv* **2011**, arXiv:1103.0398.
50. Kingma, D.P.; Ba, J. Adam: A Method for Stochastic Optimization. *arXiv* **2014**, arXiv:1412.6980.
51. Srivastava, N.; Hinton, G.; Krizhevsky, A.; Sutskever, I.; Salakhutdinov, R. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *J. Mach. Learn. Res.* **2014**, *15*, 1929–1958.
52. Guo, Z.; Zhang, Y.; Lu, W. Attention Guided Graph Convolutional Networks for Relation Extraction. In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, Florence, Italy, 28 July–2 August 2019; pp. 241–251.