electronics

MDPI

Article

Knowledge Mining of Interactions between Drugs from the Extensive Literature with a Novel Graph-Convolutional-Network-Based Method

Xingjian Xu *, **Fanjun Meng** and **Lijun Sun**

College of Computer Science and Technology, Inner Mongolia Normal University, Hohhot 010022, China
* Correspondence: xingjian@imnu.edu.cn

Abstract: Interactions between drugs can occur when two or more drugs are used for the same patient. This may result in changes in the drug's pharmacological activity, some of which are beneficial and some of which are harmful. Thus, identifying possible drug–drug interactions (DDIs) has always been a crucial research topic in the field of clinical pharmacology. As clinical trials are time-consuming and expensive, current approaches for predicting DDIs are mainly based on knowledge mining from the literature using computational methods. However, since the literature contain a large amount of unrelated information, the task of identifying drug interactions with high confidence has become challenging. Thus, here, we present a novel graph-convolutional-network-based method called DDINN to detect potential DDIs. Combining cBiLSTM, graph convolutional networks and weight-rebalanced dependency matrix, DDINN is able to extract both contexture and syntactic information efficiently from the extensive biomedical literature. At last, we compare our DDINN with some other state-of-the-art models, and it is proved that our work is more effective. In addition, the ablation experiments demonstrate the advantages of DDINN's optimization techniques as well.

Keywords: knowledge mining; drug–drug interaction; graph convolutional network; self-attention; deep learning

check for updates

Citation: Xu, X.; Meng, F.; Sun, L. Knowledge Mining of Interactions between Drugs from the Extensive Literature with a Novel Graph-Convolutional-Network-Based Method. *Electronics* **2023**, *12*, 311. https://doi.org/10.3390/electronics12020311

Academic Editors: Agnieszka Konys and Agnieszka Nowak-Brzezińska

Received: 29 November 2022
Revised: 4 January 2023
Accepted: 6 January 2023
Published: 7 January 2023

1. Introduction

When treating patients with drugs, doctors often use multiple drugs at the same time because the effectiveness of one drug is limited. Particularly in the case of severe and chronic diseases, many different drugs have to be used at the same time to treat lesions, relieve pain, prevent complications or are used for other medical reasons. As drugs are taken together, complex biochemical reactions may take place in vivo, resulting in unpredictable results, which are called drug–drug interactions (DDIs) [1]. In terms of their side effects, DDIs can be basically divided into two types: beneficial and adverse [2]. A beneficial drug interaction can improve patient outcomes, whereas adverse drug interactions can pose serious threats to patients' health, reducing the effectiveness of drugs, prolonging the course of disease, and even putting patients' lives at risk. Therefore, the identification of possible DDIs has always been a crucial research topic in clinical pharmacology [3]. A number of databases were constructed by researchers in order to document the DDIs found, such as DrugBank [4], DDInter [5], TwoSides [6] and SFINX [7].

The traditional method of obtaining DDIs involves the use of clinical trials, and these are time-consuming, expensive, and often have serious ethical implications [8]. In spite of the fact that in vivo trials remain the most accurate method for identifying DDIs, the disadvantages described above severely limit the pace at which DDIs can be identified. In recent years, many biomedical research papers have been published at high frequencies, which led researchers to study how meaningful information can be extracted from these papers. Clearly, manually curation is not feasible, so machine learning or other knowledge-mining-based methods must be employed [9]. The two examples in Figure 1 illustrates

DDI extraction from drug-related text sentences, for example, the published literature or drug descriptions. For sentence S1, the DDI type of Fluoxetine and Phenelzine is "Advice" (see Section 3.1 for a description of the specific DDI types). For sentence S2, the DDI type of PGF2alpha and Oxytocin is "Effect". Although these automated prediction methods may output false-positive and true-negative DDI predictions, they nevertheless became a mainstream approach for the DDI prediction task due to their efficacy. If it is necessary, researchers may then validate these high-confidence DDIs produced by automated DDI prediction methods clinically [10].

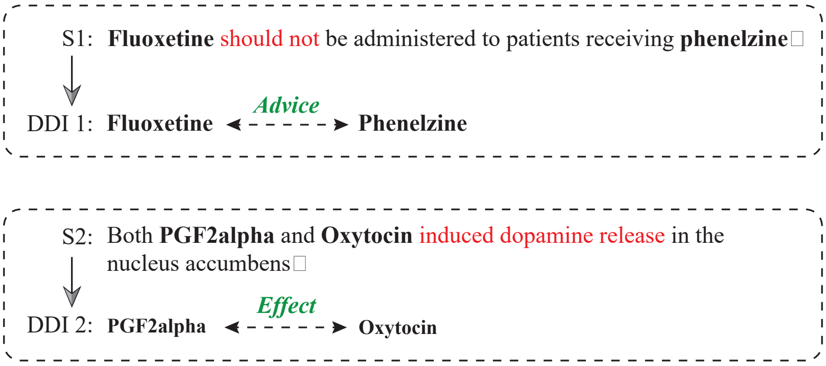
S1: Fluoxetine should not be administered to patients receiving phenelzine□
DDI 1: Fluoxetine
Advice
Phenelzine
S2: Both PGF2alpha and Oxytocin induced dopamine release in the nucleus accumbens□
DDI 2: PGF2alpha
Effect
Oxytocin

Figure 1. Two examples illustrating DDI extraction from drug-related text sentences.

Initially, there are mainly two kinds of traditional machine learning methods for automatically extracting DDI: pattern-based and feature-based ones. In pattern-based methods, experts with extensive domain knowledge are required to propose some recognizable patterns based on their own experiences [10]. Later, a number of feature-based methods are proposed, among which the best-performing ones are based on support vector machine (SVM), for example, FBK-irst [11] and NIL_UCM [12]. In general, machine learning methods that are based on features have experienced great success and are more portable than those that rely on patterns [13]. There is, however, an inherent disadvantage to these methods, which is that they heavily rely on tedious feature engineering and redundant feature selection, and defining the feature set in a supervised manner will also limit the identification of other valuable patterns. Moreover, as these methods are based on traditional machine learning models and are not capable of extracting deep features from input data, they will become much less effective when dealing with large data sets [14].

Deep learning can solve the above problems well, and it has been applied widely and successfully in a variety of other fields as well, such as in the field of computer vision, natural language processing (NLP) and speech recognition [15,16]. Deep learning methods based on graph structure have been proposed and successfully applied to the DDI prediction task [17,18]. The first wave of popular deep-learning-based DDI detection methods rely primarily on sequence-based networks, for example, the convolutional network (CNN) and recurrent neural networks (RNNs) [19]. In most cases, these methods can achieve better results than methods based on traditional machine learning models. However, the main drawback of this approach is that they cannot handle long or complex sentences in the literature's text or other information sources, mainly because of the inherent characteristics of CNN or RNN. The researchers then proposed dependency-based methods, which can be used to extract corpora that contain multiple long and complex sentences, incorporating structural information into a neural architecture for DDI prediction. As many DDI extraction corpora contain a large number of long sentences ($\geq$150 words) [20], dependency-based methods obviously have advantages over sequence-based ones. In regard to all these methods, there are still some challenges to overcome: (1) These methods only use the literature's text as input data and lack relevance to other information extraction sources; (2) due to the difficulty of parallelizing existing dependencies-based methods, such as tree-LSTM, they are often inefficient and have a disappointing runtime performance; (3) as their network is essentially linear, most of these methods are only capable of predicting the interaction of one pair of drugs at a time, which severely limits their practical usage.

In order to resolve the issues outlined above, we propose DDINN (DDI Neural Network) for the DDI prediction task, which is a novel graph-convolutional-network-based method featured by the self-attention mechanism for pruning. Our method utilizes contextual features of sentences as vertices and syntactic features as edges to construct a graph, which will be fed to GCN layers sequentially. DDINN can capture more neighborhood information of the graph more effectively by stacking the convolution layer. In particular, we rebalance the weights of each edge via a self-attention mechanism. Thus, DDINN is able to exploit both the context and structure of the input sentence to the maximum extent possible. Our final step was to train and evaluate the DINN model on the dominant DDI extraction dataset from SemEval-2013 Task 9 of the DDIExtraction 2013 dataset [21]. Validation experiments and ablation study show the effectiveness of DDINN and its superiority compared to other similar methods. Performance assessments are also conducted on the DDINN model's components to show the improvement compared with other traditional methods.

To summarize, we can state the following as our main contribution:

- DDINN: Combining graph convolutional networks with recurrent networks, we propose a novel deep learning method, DDINN, that can effectively utilize the contextual and syntactic information of input literature text at the same time.
- Weight-rebalanced dependency matrix: On the basis of dependency-aware embedding representation and self-attention-based pruning strategy, we propose a method for rebalancing the weights of all edges in the dependency matrix for GCN.
- Extensive experiments: The experimental results show that our model can predict DDI with the best F-score and has a better performance in comparison with state-of-the-art models.

Following is the outline of the remainder of this paper. In Section 2, we review the characteristics of existing DDI extraction approaches and briefly summarize the improvements made in the DDINN method proposed here to overcome their shortcomings. Section 3 describes the implementation specifics of DDINN in detail. Then, the experiments and analysis of their results are presented in Sections 4 and 5. As a final point, in Section 6, our conclusions regarding the entire work of DDINN is presented.

2. Related Works

Currently, there are three main types of DDI extraction methods: feature-based, kernel-based, and deep learning neural-network-based methods. The representative methods below will serve as the baseline for further experimental validation.

2.1. Feature-Based Methods

Feature-based methods aim to find a way to distinctively represent data characteristics using some feature representation techniques, which are called feature engineering. This process involves transforming the original data into feature vectors that can better express the essence of the problem. Then, classifiers are trained based on various linguistic features extracted from the data. For example, UTurku [22] uses dependency graph features to mine entity associations and it achieved an F-value of 59.4% in the DDIExtraction 2013 competition. WBI-DDI [23] proposes a two-stage method that first classifies the results using multiple methods including APG (all path graph), Moara, SL (shallow linguistic), and TEES (urku event extraction system) separately, and then it votes on these classification results to obtain the best classification result, which achieved an F-value of 60.9%. FBK-irst [11] constructs a combined kernel classifier by combining the feature kernel, shallow linguistic kernel and closure tree kernel for binary classification, deleting negative examples and then constructing a combined kernel classifier to achieve multi-classification, which scored 65.1% in the DDIExtraction 2013 competition F-value.

2.2. Kernel-Based Methods

The purpose of kernel-based methods is to find and learn the mutual relationships in a set of data. Widely used kernel methods include support vector machines, Gaussian processes, etc. Kernel-based methods are an effective way to solve nonlinear pattern analysis problems. The core idea is as follows: First, the original data are embedded into a suitable high-dimensional feature space by some nonlinear mapping; then, the patterns are analyzed and processed in this new space using a generic linear learner. Feature- and kernel-based DDI extraction can achieve better results than the rule-based extraction, and these methods have been the mainstream method for DDI extraction for a long period of time. The disadvantage is that they are time-consuming and laborious for performing multiple complex feature extractions, so the extraction's performance is bottlenecked and cannot be improved significantly. In 2015, Kim et al. [13] constructed kernel functions by employing a set of lexical and syntactic features based on a series of lexical and syntactic features with an F-value of 67% in DDIExtraction 2013. In 2016, Zheng et al. [24] constructed kernel functions for a graph kernel with an F-value of 68.4%. This method became the best model among the current methods using feature-based and kernel functions. It is similar to our approach in that semantic and syntactic information is integrated. However, the performance of previous studies has not been satisfactory since they have only looked at the shortest dependency path (SDP).

2.3. Neural-Network-Based Methods

Neural networks have an extremely strong feature representation capability. Thus, deep learning methods have a significant advantage over other machine learning methods in terms of accuracy and do not require a complex pre-processing process. In classification tasks, neural networks can be treated as classifiers capable of automatically extracting features. With the rapid development of deep learning, many neural-network-based DDI extraction methods emerged in recent years and have excellent performances in DDI extraction task over traditional feature- or kernel-based methods. The relationship between drug entities can be extracted using neural networks in two basic ways: sequence-based and dependency-based methods.

Different neural architectures, including CNNs and RNNs, are used in sequence-based models. Quan et al. [25] proposed a multichannel convolutional neural network (MCCNN) for automated biomedical relation extraction. As a result of MCCNN's performance on the DDIExtraction 2013 challenge dataset, MCCNN was reported to achieve an overall F-score of 70.2% compared to the linear SVM-based standard system (e.g., 67.0%). Sun et al. [26] proposed a recurrent hybrid convolutional neural network (RHCNN) for DDI extraction from the biomedical literature in which semantic embeddings and position embeddings are both used to represent the texts mentioning two drug entities. RHCNN is reported to achieve DDI automatic extraction with a micro F-score of 75.48%. In addition to CNN-based models, RNN-based ones have also been adopted for extracting DDI effectively. For example, in GGNN [27], textual drug pairs are encoded with convolutional neural networks, while molecule pairs are encoded with graph convolutional networks. DDI relations are then extracted by concatenating the outputs of these two networks. Sahu et al. [28] present three long short-term memory (LSTM) network models for mining DDI relation from biomedical text, namely B-LSTM, AB-LSTM and Joint AB-LSTM. The experimental results on the DDIExtraction2013 dataset show that the Joint AB-LSTM model produces reasonable performances with an F-score of 69.39%.

Dependency-based neural network architectures are constructed using structural information of a given sentence. It is common for the DDI extraction corpus (literature text or drug description, etc.) to contain multiple long and complex sentences, and the longest sentence may contain over 150 words, so using only sequence-based networks for extraction is extremely challenging. It is therefore very helpful to introduce structural knowledge (such as dependency trees) into the DDI extraction task. For example, Zhao et al. [29] present a syntax convolutional neural network (SCNN) for DDI extraction. In SCNN, a new syntax word embedding method is proposed that incorporates syntactic sentence information.

2.4. Improvements Made by DDINN

In order to address the shortcomings of the approaches discussed above, we made considerable improvements with respect to DDINN for the DDI extraction task:

1. To avoid the lack of representation depth caused by using traditional sequence-based or dependency-based networks alone, DDINN combines the contextual features of sentences and syntactic features together to construct a graph, which will be fed to GCN layers sequentially. By stacking the convolution layer in GCN, DDINN is able to capture more neighborhood information about the graph.
2. Traditional GCN model only allows edges between nodes with a weight of 0 or 1. There are many complex interactions between drugs in the DDI extraction task that cannot be adequately described in this manner. Thus, we propose a new method to rebalance the weights of all edges in the dependency matrix of GCN based on the dependency-aware embedding representation, so that the weights can take values ranging from 0 to 1.
3. Full dependency trees are used to avoid losing key information during the extraction of syntactic features. Specifically, we propose an attention-based pruning mechanism to minimize the loss of important cues in the full dependency tree. Unlike the rule-based or SDP-based pruning algorithms used in previous studies, this pruning strategy can be used to achieve selective pruning with different weight ratios and to reflect the different strengths of the relatedness between nodes.

3. Materials and Methods

3.1. Problem Definition

Words in the literature's text can be denoted as $\mathbf{X} = [\mathbf{x_1}, \mathbf{x_2}, \cdots, \mathbf{x_i}, \cdots, \mathbf{x_n}] \in \mathbb{R}^{d\times n}$, where n denotes the total number of words and $\mathbf{x_i} \in \mathbb{R}^d$ denotes the d-dimensional i-th embedded token. Drugs described in this text can be denoted as $\mathscr{D} = \{D_k \mid k \in [1, n]\}$. The mapping relationship between words and drugs is already known, and it can be represented as $R_{xd}(\mathbf{x_i}, D_k), R_{xd} \subset \{0, 1\}$. If $R_{xd}(\mathbf{x_i}, D_k) = 0$, it means that there is no relationship between $\mathbf{x_i}$ and D_k; otherwise, it shows a positive relationship.

All drug entities can be annotated with the following five drug–drug interaction relationship types [21]:

1. Advice: Describes recommendations when two drugs are used together;
2. Mechanism: Describes the pharmacokinetic mechanisms of two drug entities;
3. Effect: The result of the interaction of two drugs is clearly stated;
4. Int: Indicates some relationship between the two drugs, but it does not define the specific type of relationship.
5. Negative: Indicates that there is no interaction between the two drugs.

Thus, in the problem of DDI relation extraction, $\mathscr{C}$ represents the overall prediction classes as follows.

$$\mathscr{C} = \{Advice, Mechanism, Effect, Int, Negative\} \tag{1}$$

Now, the problem of DDI predication can be defined as follows. Given $\mathbf{X}$ and the R_{xd}, our DDINN method will predict drug relation set $\mathscr{R}_D$.

$$\mathscr{R}_D = \{R_{dd}(D_a, D_b) \mid a \in [1, n], b \in [1, n], a \neq b\}, \tag{2}$$

$$R_{dd}(D_a, D_b) \in \mathscr{C} \tag{3}$$

3.2. Overview of Architecture

The outline of the overall architecture of our novelly proposed DDINN model is illustrated in Figure 2. Firstly, each word in the input literature text is transformed into a token vector that consists of the embeddings of the word itself, its dependency, part of speech, and distance in sentences. These embedding vectors are concurrently sent to cBiLSTM and the weight-rebalanced dependency parser to extract the contextual and syntactic features, respectively. Then, DDINN constructs a graph, which is fed to the GCN layers, by converting contextual features into graph vertices and syntactic features into graph edges. Consequently, the representations of drug pairs and sentences consisting of other remaining words are obtained by masking the output of GCN layers. At the last step, the PPI prediction classifier, which is the final output of DDINN, is generated by concatenating the representations above sequentially and passing them to the softmax and linear layers. Below, we will provide a detailed description of the process for building the DDINN model.

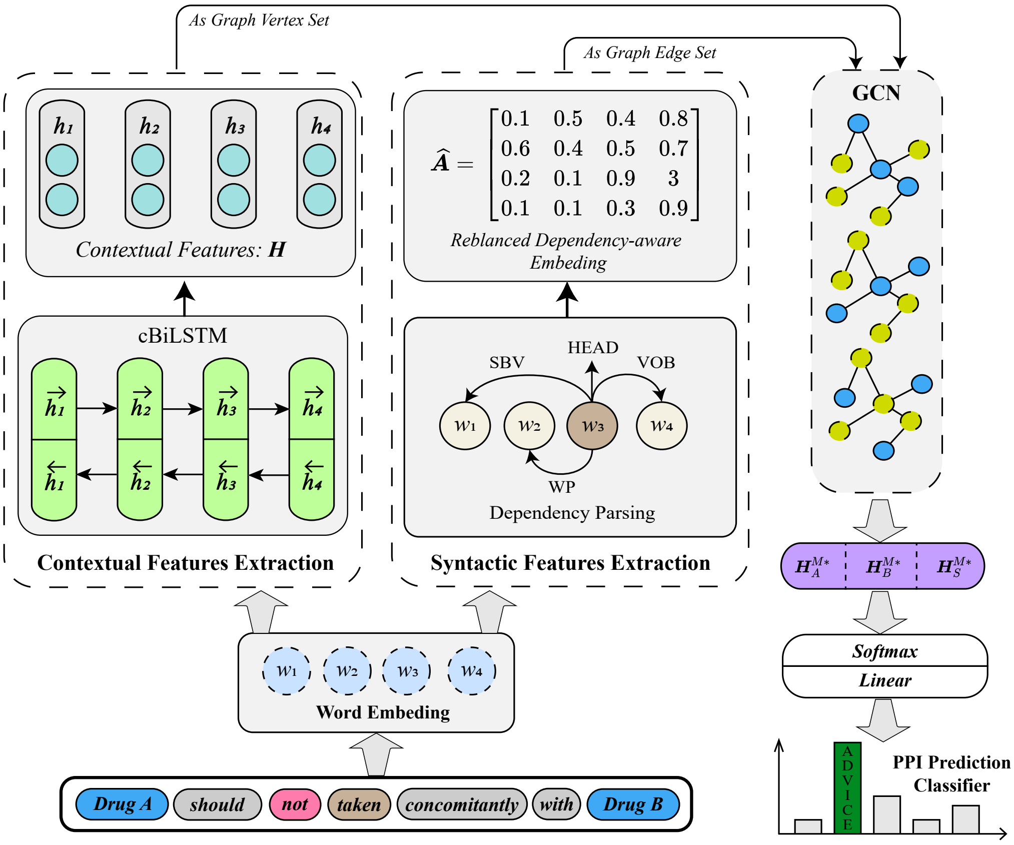
As Graph Vertex Set
As Graph Edge Set
GCN
h_1 h_2 h_3 h_4
Contextual Features: $\boldsymbol{H}$
cBiLSTM
$\overrightarrow{h_1}$ $\overrightarrow{h_2}$ $\overrightarrow{h_3}$ $\overrightarrow{h_4}$
$\overleftarrow{h_1}$ $\overleftarrow{h_2}$ $\overleftarrow{h_3}$ $\overleftarrow{h_4}$
Contextual Features Extraction
$\hat{A} = \begin{bmatrix} 0.1 & 0.5 & 0.4 & 0.8 \\ 0.6 & 0.4 & 0.5 & 0.7 \\ 0.2 & 0.1 & 0.9 & 3 \\ 0.1 & 0.1 & 0.3 & 0.9 \end{bmatrix}$
Reblanced Dependency-aware Embeding
HEAD
SBV
VOB
w_1 w_2 w_3 w_4
WP
Dependency Parsing
Syntactic Features Extraction
H_A^{M*} H_B^{M*} H_S^{M*}
Softmax
Linear
ADVICE
PPI Prediction Classifier
w_1 w_2 w_3 w_4
Word Embeding
Drug A should not taken concomitantly with Drug B

Figure 2. Architecture overview of our proposed DDINN method.

3.3. Contextual Feature Representations

In our work, the contextual and syntactic representation of sentences is used to analyze the literature's text. The concept of a bag-of-words model is often used in traditional sentiment analysis, where a document is viewed as a collection of terms or combinations of short compound words regardless of grammatical and word order. As a result, when processing sentences, word vectors are often used. It is very common for obtaining word embeddings by pre-training, and the word representation obtained in this way is often independent of the sentence's context. However, due to polysemy, the word itself can have different meanings in different contexts. Therefore, it is impossible to accurately describe the contextual meaning of the word itself in a certain context only by using the word vector. The use of context-sensitive vectors can enhance the representations of semantic relations between sentences [30].

Our solution to these issues involves the use of contextual bidirectional long short-term memory recurrent neural networks (cBiLSTM). In cBiLSTM, the contextual information extraction problem is viewed as a sequence classification problem, and a type of pooling will be performed to obtain sentence-level polarity after using RNNs as discriminative binary classifiers. There are two separate layers of LSTM in cBiLSTM. As for word token $\mathbf{x_i}$, these two LSTM layers are responsible for capturing both forward and reverse contextual information, respectively. By estimating the probability of a word based on its complete left and right contexts, the networks process the bi-directional period adjacent to the position of a word in the sentence. Therefore, the cBiLSTM is able to understand the contextual meaning of words more effectively than traditional network models.

3.3.1. Word Embedding

The first step is the vectorization of words to obtain $\mathbf{X}$. Considering that the word T_i in it does not necessarily have a mapping relationship in $\mathbf{X}$, in this case, this paper will use a uniform distribution on interval $[-0.5, 0.5]$ for its random initialization. Let $\mathbf{x}(T_i)$ denote the vector of word T_i; this representation rule is described as follows:

$$\mathbf{x}(T_i) = \begin{cases} \mathbf{x}_i, & T_i \in \mathbf{X}, \\ Uniform([-0.5, 0.5])^d, & T_i \notin \mathbf{X}. \end{cases} \tag{4}$$

3.3.2. Construct cBiLSTM

Later, word vector $\mathbf{x}$ will be processed by cBiLSTM, which will produce the forward $\overrightarrow{h_i}$ and backward $\overleftarrow{h_i}$ for word vector $\mathbf{x_i}$.

$$\overrightarrow{h_i} = LSTM(x_i, \overrightarrow{h_{i-1}}) \tag{5}$$

$$\overleftarrow{h_i} = LSTM(x_i, \overleftarrow{h_{i-1}}) \tag{6}$$

Then, we can calculate the contextual feature, h_i, of word vector $\mathbf{x_i}$ by concatenating $\overrightarrow{h_i}$ and $\overleftarrow{h_i}$ as follows.

$$h_i = [\overrightarrow{h_n}; \overleftarrow{h_i}] \in \mathbb{R}^d \tag{7}$$

At the final step of this section, all contextual information (denoted as $\boldsymbol{H}$) of the sentences will be fed to the later networks for parsing.

$$\boldsymbol{H} = (h_1, h_2, \cdots, h_n) \in \mathbb{R}^{n*d} \tag{8}$$

3.4. Syntactic Feature Representations

Dependent syntactic analyses aim to parse the text into a dependent syntactic tree. This is performed by obtaining the dependencies and association paths between words. Thus, the method gives the model a better understanding of natural language by extracting text features based on sentence structure. In addition to contextual information, syntactic information is also important. In fact, contextual and syntactic features complement each other. Here, we adopt the graph convolutional network (GCN) [31,32] to extract syntactic information. The syntactic structure of texts is more similar to that of graph data. For such non-Euclidean spatial data, traditional deep learning models do not effectively exploit or may even corrupt its intrinsic information. By extending convolution to graph-structured data, GCN is proposed, which has the ability to model common graph data in reality, and then it explores the complex relationships in it. In this paper, we use the full dependency tree as the input of the graph convolutional network and introduce the attention mechanism during the training process so as to selectively focus on the dependency substructure.

3.4.1. Construct Dependency Matrix

Based on the dependency structure, we first generate the corresponding adjacency matrix $\boldsymbol{A} \in \mathbb{R}^{n \times n}$. Most traditional dependency-tree-based networks do not employ full dependency trees to convey syntactic information from sentences. These methods often use 1 or 0 to encode syntactic dependencies between words, which indicate that the elements in the adjacency matrix $\boldsymbol{A}$ take values of only 1 or 0. However, this approach ignores the impact of different dependencies on the target task and introduces other redundant features. As a result of such strategies, which are normally determined by rule-based preprocessing, crucial information may also be lost [33,34].

To address the problems above, we introduce two more steps: a dependency-aware embedding representation method based on dependency relations in the layers and self-attention-based pruning. The dependency-aware embedding representation not only focuses on the dependency correlations between words but also considers the dependency tag types and the semantics of the words associated with the tags. The following paragraphs provide the implementation details of the dependency-aware embedding representation method.

For $\mathbf{X}$, if there is a dependency relationship between word i and j and the dependency type is φ, the corresponding dependency-type embedded vector is $\aleph_\varphi \in \mathbb{R}^{d_\phi \times 1}$, and the dependency relationship between these two words can be embedded represented as follows:

$$a_{ij} = Sigmod(Avg[\mathbf{x_i}, \mathbf{x_j}] \times \omega_\varphi \times N_\varphi + b_\varphi) \tag{9}$$

where ω_φ and b_φ are trainable parameters, Avg denotes the average value function, $Sigmod$ denotes the activation function and $\aleph_\varphi$ is initialized before the model's training and will be updated during the training process. Thus, if words i and j have syntactic dependency, the elements in matrix $\boldsymbol{A}$ can be represented as $\boldsymbol{A}_{ij} = a_{ij}$; otherwise, $\boldsymbol{A}_{ij} = 0$.

3.4.2. Self-Attention-Based Pruning

Then, in order to exploit syntactic dependencies more fully, self-attention-based pruning is employed to assign weights to all edges in the dependency graph. By incorporating the self-attention mechanism, we transform $\boldsymbol{A}$ into a soft adjacent matrix $\widehat{\boldsymbol{A}}$. Self-attention has the advantage of noticing the relationship between different positions in a single sequence. Thus, the edge weights of all node pairs in the graph are reassigned regardless of whether they are directly or indirectly connected. This is why we call output $\widehat{\boldsymbol{A}}$ as a *soft* adjacent matrix.

In the specific calculation process, we use query and key pairs of $\mathbf{x}_i$ as self-attention function parameters. By employing multi-head attention [35,36], we were able to capture a different context from multiple perspectives. In particular, the soft adjacent matrix, $\widehat{\boldsymbol{A}}$, can be calculated as follows:

$$\widehat{\boldsymbol{A}} = Softmax\left(\frac{Q\boldsymbol{W}_h{}^Q \times (K\boldsymbol{W}_h{}^K)^T}{\sqrt{d}}\right) \tag{10}$$

where $Softmax$ is the activation function and Q and K are the features of the previous convolutional layer $h^{(l-1)}$. $\boldsymbol{W}_h{}^Q \in \mathbb{R}^{d*d}$ and $\boldsymbol{W}_h{}^K \in \mathbb{R}^{d*d}$ are used for projection parameters, where h denotes the h-th head in $\boldsymbol{H}$, which is defined in Equation (8).

3.4.3. Construct GCN

Then, contextual information $\boldsymbol{H}$, which is the output of Equation (8), and adjacency matrix $\widehat{\boldsymbol{A}}$ will be fed into the l-level GCN:

$$\boldsymbol{H}^{(l)} = Relu(\widehat{\boldsymbol{D}}^{-\frac{1}{2}} \widehat{\boldsymbol{A}}_D \widehat{\boldsymbol{D}}^{-\frac{1}{2}} \boldsymbol{H}^{(l-1)} \boldsymbol{W}^{(l-1)} + b_l) \tag{11}$$

where $Relu$ is the activation function, $\widehat{\boldsymbol{A}}_D$ is the edge matrix of $\widehat{\boldsymbol{A}}$, $\widehat{\boldsymbol{D}}$ denotes the degree matrix of $\widehat{\boldsymbol{A}}_D$, $\boldsymbol{H}^{(l-1)}$ denotes the node features of the $(l-1)$-th level GCN (when $l = 1$, $\boldsymbol{H}^{(l-1)} = \boldsymbol{H}$) and $\boldsymbol{W}^{(l-1)}$ denotes the weight matrix of the $(l-1)$-th level GCN.

Finally, in order to further enhance the generalization capability of the model, the output of the GCN layers above will be processed by a pooling layer, dropout layer, and Relu layer:

$$\boldsymbol{H}^* = \omega \times Relu(Dropout(Pooling(\boldsymbol{H}^{(l)}))) + b \tag{12}$$

where $\boldsymbol{H}^*$ is the final output of GCN, which holds the contextual and syntactic feature information of text $\mathbf{X}$ at the same time.

3.5. Extract DDI

3.5.1. Extract Masked Representations

After completing the above steps, we have hidden representations of each word in the input literature text, which can be simply denoted as $\mathbf{w}_i$ for word i. The problem in this step can be defined as follows: Within the input word representations $[\mathbf{w}_1, \cdots, \mathbf{w}_n]$, drug A is mapped to $\mathbf{w}_a$, and drug B is mapped to $\mathbf{w}_b$; we want to extract the relationship between drug A and B. In order to achieve this, we first calculate the masked representations of drug A, drug B, and the sentence including other words (i.e., words except for $\mathbf{w}_a$ and $\mathbf{w}_b$), which are denoted as $\boldsymbol{H}_A^{M*}$, $\boldsymbol{H}_B^{M*}$, and $\boldsymbol{H}_S^{M*}$, respectively. The calculation process is as follows:

$$\boldsymbol{H}_S^{M*} = MaxPooling(Mask_S(\boldsymbol{H}^*)) \tag{13}$$

$$\boldsymbol{H}_A^{M*} = MaxPooling(Mask_A(\boldsymbol{H}^*)) \tag{14}$$

$$\boldsymbol{H}_B^{M*} = MaxPooling(Mask_B(\boldsymbol{H}^*)) \tag{15}$$

where $\boldsymbol{H}^*$ is the output of Equation (12), $MaxPooling$ denotes an activation function that can transform n output vectors to only one vector, i.e., $MaxPooling \in \mathbb{R}^{n \times d} \to \mathbb{R}^d$. $Mask_A$, $Mask_B$ and $Mask_S$ denote functions that can select only representations for drug A, drug B and sentences formed by the remaining words, respectively.

3.5.2. Construct DDI Classifier

Finally, we can predict the DDI by using a classifier. Firstly, we concatenate the masked representations above and then feed them to a fully connected layer [37]. The final result of this classifier is denoted as $\boldsymbol{H}_{Final}$, which is calculated as follows:

$$\boldsymbol{H}_{Final} = FC(Concat(\boldsymbol{H}_A^{M*}, \boldsymbol{H}_B^{M*}, \boldsymbol{H}_S^{M*})) \tag{16}$$

where FC is the fully connected layer, and $Concat$ is the function that concatenates all its parameters. $\boldsymbol{H}_{Final}$ will then be fed into a linear layer and a softmax layer to output the probability distribution for the DDI relationship between these drugs [38,39]:

$$\boldsymbol{P} = Softmax(Linear(\boldsymbol{H}_{Final})) \tag{17}$$

4. Experiments

4.1. Dataset

In this paper, we evaluate DDINN on the DDIExtraction2013 dataset [20], which is most widely used when comparing the performances of different DDI extraction algorithms. Prior to 2011, there were relatively few studies related to the DDIExtraction task due to the lack of standard datasets, and almost all of those studies were rule-based. These rules have to be formulated by professionals, and the DDI extraction is achieved by matching the DDI expressions in the sentences with the formulated rules. This approach is more effective for composing simple sentences. However, for long and complex sentences, especially those with many subordinate clauses, the performance of this method is much less effective. In 2011, the SemEval 2011 competition established the DDIExtraction subtask and provided the standard DDIExtraction dataset for the first time. Subsequently, in 2013, the SemEval 2013 competition supplemented and improved the dataset, which can be referred to as DDIExtraction2013.

The text corpus of this dataset has two sources: (1) literature abstracts in the discipline of drug interactions downloaded from the MedLine (https://medline.com/, accessed on 20 February 2022) medical literature retrieval system and (2) articles studying drug interactions downloaded from the DrugBank (https://drugbank.com/, accessed on 23 February 2022) online database. A total of 18,491 pharmacological substances and 4999 drug–drug interactions were manually annotated in this DDI corpus, which consists of 1017 documents (784 paragraphs from DrugBank and 233 abstracts from MedLine). All documents contain 5806 sentences and 127,653 tokens. The details of the DDIExtraction2013 dataset are listed in Table 1.

Table 1. The statistics information of DDIExtraction 2013 dataset.

Type	Training	Test	Total
Advice	826	221	1047
Mechanism	1319	302	1621
Effect	1687	360	2047
Int	188	96	284
Negative	23,772	4737	28,509

4.2. Training

In the training process, cross entropy cost function and L^2 regularization are used as the optimization objective. The cross entropy is defined as follows:

$$l_i = -\ln {Y_i}^T P_i \tag{18}$$

where Y_i denotes the one-hot representation of the i-th instance label, and P_i is the model output, which is defined in Equation (17). For a mini batch $\mathcal{M} = [\mathbf{X}_1, \mathbf{X}_2, \cdots, \mathbf{X}_M]$, we defined the optimization objective as follows:

$$\mathcal{J}(\theta) = \frac{1}{|\mathcal{M}|} \sum_{i=1}^{|\mathcal{M}|} l_i + \lambda \|\theta\|_2^2 \tag{19}$$

where θ includes all the parameters in our model. At the final step, parameter θ in the objective function, $\mathcal{J}(\theta)$, is optimized with Nadam [40], which is an algorithm that performs first-order gradient optimization on an efficient stochastic objective function.

The models are randomly initialized at the beginning, so if a higher learning rate is selected at this point, the model may become unstable or oscillate, while a lower learning rate will result in a slower convergence speed. The learning rate scheduler with exponential decay [41] is used to control the dynamic change of the learning rate during the training process (see Figure 3). It can slow down overfitting in the initial stages and maintain the stability of the deep layer. Upon the completion of training, the model that can predict interactions between two drugs is obtained.

4.3. Experiment Setup

The DDINN is implemented with PyTorch (https://pytorch.org/, accessed on 10 January 2022) and open-sourced at Github (https://github.com/xingjianxu/DDINN, accessed on 10 January 2022). We use pre-trained word embeddings from GloVe [42] combined with PMCVec [43,44], which is based on unlabeled biomedical texts from PubMed (https://pubmed.ncbi.nlm.nih.gov/, accessed on 10 January 2022) and PubMed Central (https://www.ncbi.nlm.nih.gov/pmc/, accessed on 10 January 2022). In order to obtain the dependency tree, dependency label, and POS tag of each word, we use the Stanford Parser (https://nlp.stanford.edu/software/lex-parser.shtml, accessed on 10 January 2022). All experiments are conducted with two RTX 3090 GPUs. The detailed parameters are listed in Table 2.

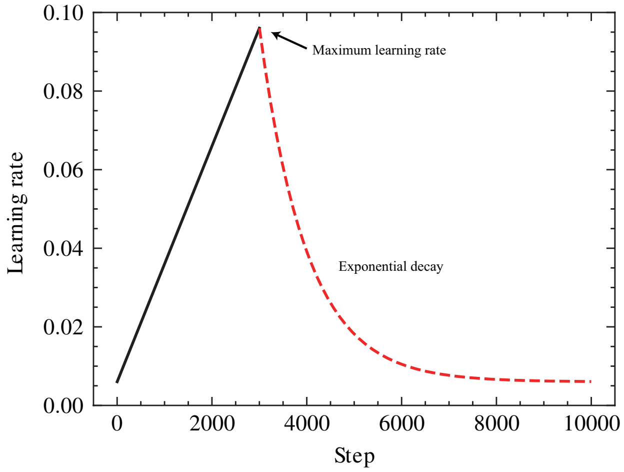

Figure 3. Learning rate exponential decay.

Table 2. The main hyperparameter settings used in DDINN implementations and evaluation experiments.

Module	Parameter	Value
Word embedding	Size (per word)	200
Stanford Parser	Word dimension	200
	Dependency dimension	20
	Distance embedding	20
cBiLSTM	Dimension	400
	Dropout	0.4
GCN	Dimension	300
	Layer	3
Attention	Dimension	300
	Dropout	0.2
Training	Epoch	30
	L^2 regularization	5×10^{-5}
	Batch	30
	Maximum learning rate	0.005

4.4. Assessment Metrics

In order to evaluate the quality of prediction results, micro-precision, micro-recall, and micro-F score are employed as assessment metrics, which are denoted as P_{micro}, R_{micro}, and F_{micro}, respectively. As described in Table 1, we can define the prediction classes. We set $\mathscr{D}$ as

$$\mathscr{D} = \{Advice, Mechanism, Effect, Int, Negative\} \tag{20}$$

and these metrics above can be calculated as follows:

$$P_{micro} = \frac{\overline{TP}}{\overline{TP} + \overline{FP}} = \frac{\sum_{i=1}^{n} TP_i}{\sum_{i=1}^{n} TP_i + \sum_{i=1}^{n} FP_i} \tag{21}$$

$$R_{micro} = \frac{\overline{TP}}{\overline{TP} + \overline{FN}} = \frac{\sum_{i=1}^{n} TP_i}{\sum_{i=1}^{n} TP_i + \sum_{i=1}^{n} FN_i} \tag{22}$$

$$F_{micro} = \frac{2 \times P_{micro} \times R_{micro}}{P_{micro} + R_{micro}} \tag{23}$$

where TP_i denotes the true positives in the prediction class $i \in \mathscr{D}$, FP_i denotes the false positives and FN_i denotes the false negatives.

4.5. Baselines

The following two kinds of methods are selected as the baseline for evaluating the performance of DDINN in this paper:

- Traditional statistical-model-based methods, including UTurku [22], FBK-irst [11] and WBI-DDI [23]: Such kinds of methods mainly use features and kernel functions to predict the DDI relationship.
- Deep learning neural-network-model-based methods, including MCCNN [25], Joint AB-LSTM [28], GGNN [27], RHCNN [26] and GCNN [45]: The application of neural networks significantly improved prediction performances compared to methods based on traditional statistical models.

5. Results and Discussion

5.1. Performance Comparison

As shown in in Table 3, we compare the performance of our DDINN method to those of the other eight baseline methods. For each method, the F_{micro} score for four kinds of DDI types and the overall precision, recall and F_{micro} score are listed. The performance statistics are obtained by conducting test experiments on the DDIExtraction2013 dataset, except for UTurku, GGNN and GCNN, which are directly cited from their original papers. This is because we cannot find available codes or runnable binaries for these methods, and they all conducted the performance test on the DDIExtraction2013 dataset. The highest values in each test are marked in bold, and the second best ones are marked underlined.

In comparison with all baseline methods, except for the PPI type of Int, DDINN exhibited the highest performance scores. The main reason for this is that DDINN requires a relatively large amount of training data, and training data with the Int PPI type only rarely (1.68% in total training data) appears in the DDIExtraction2013 dataset (see Table 1). The experimental results proved that the series of optimization used in DDINN finally worked and successfully improved the quality of the results of the DDI prediction task.

The training process of this model on the DDIExtraction2013 dataset is shown in Figure 4, which shows the changes in the precision, recall, and the F_{micro} score values over the epoch. From the figure, it can be seen that all these values improve faster in the early stage of the training, and then they fluctuate continuously to find the local optimal value; finally, they gradually converge to smooth values.

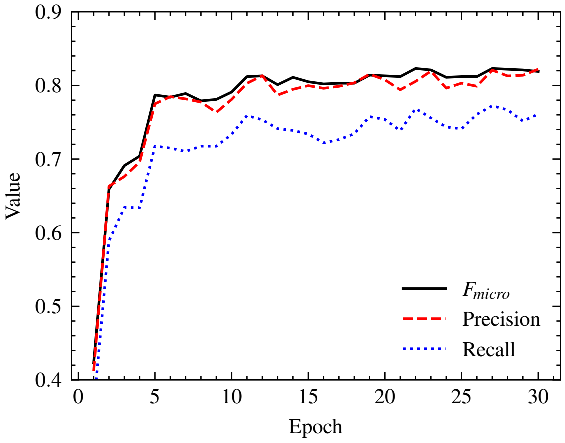

Figure 4. Precision, recall, and F_{micro} value on entire test dataset in the training process.

5.2. Error Analysis

Figure 5 shows the confusion matrix of the model in this paper. Each column of the matrix represents an instance prediction of a class, while each row represents an actual instance of the class. The darker color in the figure indicates a larger proportion of error. To clearly highlight the misclassification of the DDI predicted by our model, the values in the confusion matrix are normalized.

Table 3. Performance comparisons with other DDI prediction methods.

Method		**F-Score for Each DDI Type**				**Overall**		
		Advice	**Mechanism**	**Effect**	**Int**	**Precision**	**Recall**	**F-Score**
Traditional models	UTurku	0.621	0.586	0.601	0.504	0.728	0.489	0.599
	FBK-irst	0.695	0.671	0.626	0.554	0.651	0.651	0.658
	WBI-DDI	0.627	0.611	0.608	0.510	0.650	0.563	0.606
Deep learning networks	MCCNN	0.785	0.719	0.683	0.510	0.759	0.652	0.702
	Joint AB-LSTM	0.796	0.761	0.674	0.461	0.733	0.698	0.715
	GGNN	0.817	0.735	0.710	0.460	0.734	0.719	0.726
	RHCNN	0.806	0.780	0.735	**0.578**	0.773	0.735	0.753
	GCNN	0.834 [1]	0.798	0.759	0.514	0.800	0.738	0.769
	DDINN (ours)	**0.863** [2]	**0.820**	**0.772**	0.566	**0.822**	**0.761**	**0.816**

[1] The second best value of the column is marked by underline style. [2] The best value of the column is marked by bold style.

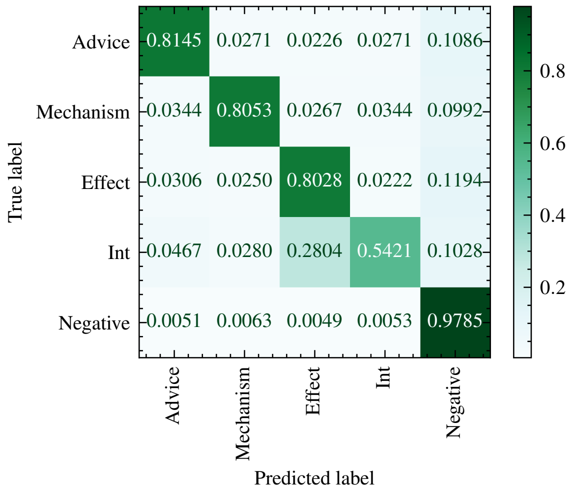

Figure 5. Confusion matrix with L1 normalization.

From Figure 5, we can see that there are two main types of classification errors for the model: (1) the class of relations with the Int type is often incorrectly classified as the Advice type; (2) the four positive classes of relations (Advice, Mechanism, Effect and Int) are often incorrectly classified in the negative class.

For the first type of error, which is already briefly discussed in Section 5.1, the reason is that the number of Int DDI type is too small, with only 96 instances in the training set, and we observed in this paper that the instances of DDI type Int and Effect in the dataset have similar semantics, resulting in the model's inability in classifying these two categories well. The second type of error is also mainly caused by the dataset, where the number of negative categories in the dataset is 28,509, while the number of remaining positive examples is only 4999, which inevitably allows a small number of DDI types to be misclassified into the negative DDI type.

5.3. Ablation Study

Additional ablation experiments are conducted in order to evaluate the influence of different modules or optimizations on DDI prediction. Firstly, the impact of contextual representation methods has been investigated. The corresponding results are shown in Table 4, in which method "GCN only" refers to the model without any contextual representation engagement, and the others are models using GRU, LSTM and cBiLSTM to extract contextual representations, respectively. From Table 4, we can see that cBiLSTM improves the F-score of the GCN-only model by 6.1%, and the cBiLSTM model is indeed more suitable for DDI prediction tasks than some other RNN models.

Table 4. Ablation study on different contextual representation methods.

Method	Precision	Recall	F-Score
GCN only	0.761	0.722	0.777
+GRU	0.784	0.735	0.783
+LSTM	0.796	0.741	0.803
+cBiLSTM	0.822	0.761	0.816

We also investigate the influence of the self-attention pooling strategy used in the construction of the weight-rebalanced dependency matrix, and the results are listed in Table 5. "Full tree" means the method without any pruning strategy. "LAC ($k = n$)" means using the LCA strategy [46] to conduct the tree pruning, and the subtree only includes tokens with the range of n words. From Table 5, we can see that the self-attention-based pruning strategy improved the F-score by 5.4% compared with the full tree strategy. Self-attention adds some complexity to the model, but it is worth it.

Table 5. Ablation study on different syntactic dependency extraction methods.

Method	Precision	Recall	F-Score
Full tree	0.762	0.749	0.762
LCA ($k = 0$)	0.727	0.694	0.725
LCA ($k = 1$)	0.749	0.703	0.738
LCA ($k = 2$)	0.747	0.711	0.744
LCA ($k = 3$)	0.761	0.729	0.759
Self-attention	0.822	0.761	0.816

6. Conclusions

In this paper, we proposed a novel graph-convolutional-network-based method for the knowledge mining of interactions between drugs from the extensive literature, which is called DDINN. Our method makes full use of cBiLSTM to capture the contextual information of input sentences and target drug entities. Additionally, the self-attention mechanism is used to maximize the acquisition of syntactic information related to the DDI extraction task and discard irrelevant information. At last, the output of cBiLSTM and weight-rebalanced dependency matrix will be fed into GCN layers to obtain the DDI type classifier.

The evaluation experiments prove that the DDINN model in this paper achieved higher performance results compared to other state-of-the-art DDI prediction methods in the DDIExtraction2013 dataset. In future work, we will consider data augmentation and other schemes to improve the performance of the DDINN relative to the imbalanced dataset. Additionally, we hope to improve the interpretability [47,48] of deep learning networks in DDINN, which will enhance its utility in the medical field.

Author Contributions: Conceptualization, X.X.; funding acquisition, X.X. and F.M.; project administration, X.X.; software, X.X. and L.S.; validation, L.S. and F.M.; writing—original draft, X.X. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by grants from the Fundamental Research Funds for Inner Mongolia Normal University (2022JBQN105), Inner Mongolia JMRH Project (JMRKX202201) and Fundamental Research Funds for Inner Mongolia Normal University (2022JBQN109).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The datasets and codes used in this paper to produce the experimental results are publicly available at GitHub (https://github.com/xingjainxu/DDINN, accessed on 29 December 2022). The project code of biolitNER is also open sourced and accessible at GitHub under the GPLv3 license.

Acknowledgments: We thank Sun for their help in setting up the experiment's server node.

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

Abbreviations

The following abbreviations are used in this manuscript:

DDI	Drug–drug interaction;
GCN	Graph convolutional network;
cBiLSTM	Contextual bidirectional long short-term memory recurrent neural networks;
RNN	Recurrent neural network;
GRU	Gate recurrent Unit;
POS	Part of speech.

References

1. Becker, M.L.; Kallewaard, M.; Caspers, P.W.J.; Visser, L.E.; Leufkens, H.G.M.; Stricker, B.H.C. Hospitalisations and emergency department visits due to drug-drug interactions: A literature review. *Pharmacoepidemiol. Drug Saf.* **2007**, *16*, 641–651. [CrossRef] [PubMed]
2. Chee, B.W.; Berlin, R.; Schatz, B. Predicting Adverse Drug Events from Personal Health Messages. *AMIA Annu. Symp. Proc.* **2011**, *2011*, 217–226.
3. van der Heijden, P.G.M.; van Puijenbroek, E.P.; van Buuren, S.; van der Hofstede, J.W. On the assessment of adverse drug reactions from spontaneous reporting systems: The influence of under-reporting on odds ratios. *Stat. Med.* **2002**, *21*, 2027–2044. [CrossRef] [PubMed]
4. Wishart, D.S.; Feunang, Y.D.; Guo, A.C.; Lo, E.J.; Marcu, A.; Grant, J.R.; Sajed, T.; Johnson, D.; Li, C.; Sayeeda, Z.; et al. DrugBank 5.0: A major update to the DrugBank database for 2018. *Nucleic Acids Res.* **2018**, *46*, D1074–D1082. [CrossRef] [PubMed]
5. Xiong, G.; Yang, Z.; Yi, J.; Wang, N.; Wang, L.; Zhu, H.; Wu, C.; Lu, A.; Chen, X.; Liu, S.; et al. DDInter: An online drug–drug interaction database towards improving clinical decision-making and patient safety. *Nucleic Acids Res.* **2022**, *50*, D1200–D1207. [CrossRef] [PubMed]
6. Tatonetti, N.P.; Ye, P.P.; Daneshjou, R.; Altman, R.B. Data-Driven Prediction of Drug Effects and Interactions. *Sci. Transl. Med.* **2012**, *4*, 125ra31. [CrossRef]
7. Böttiger, Y.; Laine, K.; Andersson, M.L.; Korhonen, T.; Molin, B.; Ovesjö, M.L.; Tirkkonen, T.; Rane, A.; Gustafsson, L.L.; Eiermann, B. SFINX-a drug-drug interaction database designed for clinical decision support systems. *Eur. J. Clin. Pharmacol.* **2009**, *65*, 627–633. [CrossRef]
8. Zhang, L.; Zhang, Y.D.; Zhao, P.; Huang, S.M. Predicting Drug–Drug Interactions: An FDA Perspective. *AAPS J.* **2009**, *11*, 300–306. [CrossRef]
9. Zhao, L.; Au, J.L.S.; Wientjes, M.G. Comparison of methods for evaluating drug-drug interaction. *Front. Biosci. Elite Ed.* **2010**, *2*, 241–249.
10. Roblek, T.; Vaupotic, T.; Mrhar, A.; Lainscak, M. Drug-drug interaction software in clinical practice: A systematic review. *Eur. J. Clin. Pharmacol.* **2015**, *71*, 131–142. [CrossRef]
11. Chowdhury, M.F.M.; Lavelli, A. *FBK-irst: A Multi-Phase Kernel Based Approach for Drug-Drug Interaction Detection and Classification that Exploits Linguistic Information*; Association for Computational Linguistics: Atlanta, GA, USA, 2013; pp. 351–355.
12. Bokharaeian, B.; Díaz, A. NIL_UCM: Extracting Drug-Drug interactions from text through combination of sequence and tree kernels. In *Proceedings of the Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*; Association for Computational Linguistics: Atlanta, GA, USA, 2013; pp. 644–650.
13. Kim, S.; Liu, H.; Yeganova, L.; Wilbur, W.J. Extracting drug-drug interactions from literature using a rich feature-based linear kernel approach. *J. Biomed. Inform.* **2015**, *55*, 23–30. [CrossRef]
14. Vilar, S.; Friedman, C.; Hripcsak, G. Detection of drug-drug interactions through data mining studies using clinical sources, scientific literature and social media. *Brief. Bioinform.* **2018**, *19*, 863–877. [CrossRef] [PubMed]
15. LeCun, Y.; Bengio, Y.; Hinton, G. Deep learning. *Nature* **2015**, *521*, 436–444. [CrossRef] [PubMed]
16. Cho, K.; Courville, A.; Bengio, Y. Describing Multimedia Content Using Attention-Based Encoder-Decoder Networks. *IEEE Trans. Multimed.* **2015**, *17*, 1875–1886. [CrossRef]

17. Karim, M.R.; Cochez, M.; Jares, J.B.; Uddin, M.; Beyan, O.; Decker, S. Drug-Drug Interaction Prediction Based on Knowledge Graph Embeddings and Convolutional-LSTM Network. In Proceedings of the 10th ACM International Conference on Bioinformatics, Computational Biology and Health Informatics, Niagara Falls, NY, USA, 7–10 September 2019; Association for Computing Machinery: New York, NY, USA, 2019; pp. 113–123. [CrossRef]
18. Nicholson, D.N.; Greene, C.S. Constructing knowledge graphs and their biomedical applications. *Comput. Struct. Biotechnol. J.* **2020**, *18*, 1414–1428. [CrossRef]
19. Kumar Shukla, P.; Kumar Shukla, P.; Sharma, P.; Rawat, P.; Samar, J.; Moriwal, R.; Kaur, M. Efficient prediction of drug-drug interaction using deep learning models. *IET Syst. Biol.* **2020**, *14*, 211–216. [CrossRef]
20. Herrero-Zazo, M.; Segura-Bedmar, I.; Martínez, P.; Declerck, T. The DDI corpus: An annotated corpus with pharmacological substances and drug–drug interactions. *J. Biomed. Inform.* **2013**, *46*, 914–920. [CrossRef]
21. Segura-Bedmar, I.; Martínez Fernández, P.; Herrero Zazo, M. *SemEval-2013 Task 9: Extraction of Drug-Drug Interactions from Biomedical Texts (DDIExtraction 2013)*; Accepted: 2015–04-23T13:42:53Z; Association for Computational Linguistics: Atlanta, GA, USA, June 2013.
22. Björne, J.; Kaewphan, S.; Salakoski, T. UTurku: Drug Named Entity Recognition and Drug-Drug Interaction Extraction Using SVM Classification and Domain Knowledge. In *Proceedings of the Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*; Association for Computational Linguistics: Atlanta, GA, USA, 2013; pp. 651–659.
23. Thomas, P.; Neves, M.; Rocktäschel, T.; Leser, U. WBI-DDI: Drug-Drug Interaction Extraction using Majority Voting. In *Proceedings of the Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*; Association for Computational Linguistics: Atlanta, GA, USA, 2013; pp. 628–635.
24. Zheng, W.; Lin, H.; Zhao, Z.; Xu, B.; Zhang, Y.; Yang, Z.; Wang, J. A graph kernel based on context vectors for extracting drug–drug interactions. *J. Biomed. Inform.* **2016**, *61*, 34–43. [CrossRef]
25. Quan, C.; Hua, L.; Sun, X.; Bai, W. Multichannel Convolutional Neural Network for Biological Relation Extraction. *BioMed Res. Int.* **2016**, *2016*, e1850404. [CrossRef]
26. Sun, X.; Dong, K.; Ma, L.; Sutcliffe, R.; He, F.; Chen, S.; Feng, J. Drug-Drug Interaction Extraction via Recurrent Hybrid Convolutional Neural Networks with an Improved Focal Loss. *Entropy* **2019**, *21*, 37. [CrossRef]
27. Asada, M.; Miwa, M.; Sasaki, Y. Enhancing Drug-Drug Interaction Extraction from Texts by Molecular Structure Information. In Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers), Melbourne, Australia, 15–20 July 2018; Association for Computational Linguistics: Melbourne, Australia, 2018; pp. 680–685. [CrossRef]
28. Sahu, S.K.; Anand, A. Drug-drug interaction extraction from biomedical texts using long short-term memory network. *J. Biomed. Informatics* **2018**, *86*, 15–24. [CrossRef] [PubMed]
29. Zhao, Z.; Yang, Z.; Luo, L.; Lin, H.; Wang, J. Drug drug interaction extraction from biomedical literature using syntax convolutional neural network. *Bioinformatics* **2016**, *32*, 3444–3453. [CrossRef]
30. Camacho-Collados, J.; Pilehvar, M.T. From word to sense embeddings: A survey on vector representations of meaning. *J. Artif. Intell. Res.* **2018**, *63*, 743–788. [CrossRef]
31. Zhang, S.; Tong, H.; Xu, J.; Maciejewski, R. Graph convolutional networks: A comprehensive review. *Comput. Soc. Netw.* **2019**, *6*, 11. [CrossRef]
32. Santoro, A.; Raposo, D.; Barrett, D.G.; Malinowski, M.; Pascanu, R.; Battaglia, P.; Lillicrap, T. A simple neural network module for relational reasoning. In Proceedings of the 31st International Conference on Neural Information Processing Systems, Long Beach, CA, USA, 4–9 December 2017; Curran Associates Inc.: Red Hook, NY, USA, 2017; NIPS'17, pp. 4974–4983.
33. Lin, J.; Sun, X.; Ma, S.; Su, Q. Global Encoding for Abstractive Summarization. In Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers), Melbourne, Australia, 15–20 July 2018; Association for Computational Linguistics: Melbourne, Australia, 2018; pp. 163–169. [CrossRef]
34. Yu, A.W.; Dohan, D.; Luong, T.; Zhao, R.; Chen, K.; Le, Q. QANet: Combining Local Convolution with Global Self-Attention for Reading Comprehension. *arXiv* **2018**, arXiv:1804.09541.
35. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser.; Polosukhin, I. Attention is all you need. In Proceedings of the 31st International Conference on Neural Information Processing Systems, Long Beach, CA, USA, 4–9 December 2017; Curran Associates Inc.: Red Hook, NY, USA, 2017; pp. 6000–6010.
36. Hacene, G.B.; Lassance, C.; Gripon, V.; Courbariaux, M.; Bengio, Y. Attention Based Pruning for Shift Networks. IEEE Computer Society. In Proceedings of the 2020 25th International Conference on Pattern Recognition (ICPR), Milan, Italy, 10–15 January 2021; pp. 4054–4061. [CrossRef]
37. Lee, K.; He, L.; Lewis, M.; Zettlemoyer, L. End-to-end Neural Coreference Resolution. In Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, Copenhagen, Denmark, 7–11 September 2017; Association for Computational Linguistics: Copenhagen, Denmark, 2017; pp. 188–197. [CrossRef]
38. Mohammed, A.A.; Umaashankar, V. Effectiveness of Hierarchical Softmax in Large Scale Classification Tasks. In Proceedings of the 2018 International Conference on Advances in Computing, Communications and Informatics (ICACCI), Bangalore, India, 19–22 September 2018; pp. 1090–1094. [CrossRef]

39. Qi, X.; Wang, T.; Liu, J. Comparison of Support Vector Machine and Softmax Classifiers in Computer Vision. In Proceedings of the 2017 Second International Conference on Mechanical, Control and Computer Engineering (ICMCCE), Harbin, China, 8–10 December 2017; pp. 151–155. [CrossRef]
40. Kingma, D.P.; Ba, J. Adam: A Method for Stochastic Optimization. *arXiv* **2017**, arXiv:1412.6980.
41. You, K.; Long, M.; Wang, J.; Jordan, M.I. How Does Learning Rate Decay Help Modern Neural Networks? *arXiv* **2019**, arXiv:1908.01878.
42. Pennington, J.; Socher, R.; Manning, C.D. GloVe: Global Vectors for Word Representation. In Proceedings of the Empirical Methods in Natural Language Processing (EMNLP), Doha, Qatar, 26–28 October 2014; pp. 1532–1543.
43. Moen, S.; Ananiadou, T.S.S. Distributional semantics resources for biomedical text processing. *Proc. LBM* **2013**, 39–44.
44. Gero, Z.; Ho, J. PMCVec: Distributed phrase representation for biomedical text processing. *J. Biomed. Inform.* **2019**, *100*, 100047. [CrossRef]
45. Yi, Z.; Li, S.; Yu, J.; Tan, Y.; Wu, Q.; Yuan, H.; Wang, T. Drug-Drug Interaction Extraction via Recurrent Neural Network with Multiple Attention Layers. In Proceedings of the Advanced Data Mining and Applications; Lecture Notes in Computer Science, Singapore, 5–6 November 2017; Cong, G., Peng, W.C., Zhang, W.E., Li, C., Sun, A., Eds.; Springer International Publishing: Cham, Switzerland, 2017; pp. 554–566. [CrossRef]
46. 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; Association for Computational Linguistics: Brussels, Belgium, 2018; pp. 2205–2215. [CrossRef]
47. Murdoch, W.J.; Singh, C.; Kumbier, K.; Abbasi-Asl, R.; Yu, B. Definitions, methods, and applications in interpretable machine learning. *Proc. Natl. Acad. Sci. USA* **2019**, *116*, 22071–22080. [CrossRef]
48. Meng, C.; Trinh, L.; Xu, N.; Enouen, J.; Liu, Y. Interpretability and fairness evaluation of deep learning models on MIMIC-IV dataset. *Sci. Rep.* **2022**, *12*, 7166. [CrossRef] [PubMed]

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.