Harvesting Drug Effectiveness from Social Media

Zi Chai, Xiaojun Wan, Zhao Zhang and Minjie Li
Institute of Computer Science and Technology, Peking University
The MOE Key Laboratory of Computational Linguistics, Peking University
{chaizi,wanxiaojun}@pku.edu.cn,{cheung.pku,sherrylilmj}@gmail.com

ABSTRACT

Drug effectiveness describes the capacity of a drug to cure a disease, which is of great importance for drug safety. To get this information, a number of real-world patient-oriented outcomes are required. However, current surveillance systems can only capture a small portion of them, and there is a time lag in processing the reported data. Since social media provides quantities of patient-oriented user posts in real-time, it is of great value to automatically extract drug effectiveness from these data. To this end, we build a dataset containing 25K tweets describing drug use, and further harvest drug effectiveness by performing Relation Extraction (RE) between chemicals and diseases. Most prior works about RE deal with mention pairs independently, which is not suitable for our task since interactions across mention pairs are widespread. In this paper, we propose a model regarding mention pairs as nodes connected by multiple types of edges. With the help of graph-based information transfers over time, it deals with all mention pairs simultaneously to capture their interactions. Besides, a novel idea is used to perform multiple instance learning, a big challenge in general RE tasks. Extensive experimental results show that our model outperforms previous work by a substantial margin.

CCS CONCEPTS

• **Computing methodologies → Information extraction**; **Natural language processing**.

KEYWORDS

Drug effectiveness discovery; relation extraction; graph neural networks; graph-based information transfer over time

ACM Reference Format:
Zi Chai, Xiaojun Wan, Zhao Zhang and Minjie Li. 2019. Harvesting Drug Effectiveness from Social Media. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '19), July 21–25, 2019, Paris, France.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3331184.3331263

1 INTRODUCTION

Drug efficacy describes the capacity of a drug to produce an effect (e.g. lower blood pressure) under ideal circumstances. It should only be assessed under ideal conditions such as clinical trials [29]. However, a drug that is efficacious in clinical trials is often not very effective in actual use [19, 30]. In most cases, it is more appropriate to use drug effectiveness, which differs from efficacy in that it takes into account how well a drug works in real-world use [29, 36]. To get this information, a number of real-world patient-oriented outcomes are required. Although many spontaneous reporting systems are used for post-marketing surveillance, they heavily rely on human experts and often suffer from under-reporting problems [1, 12] as well as a time lag in processing reported data [34]. As a result, it is of great value to find an automatic way to harvest drug effectiveness to ensure drug safety.

Nowadays, social media such as Twitter has played a key role in human communication. Large scales of user posts mentioning drug use are generated in real time, making it possible for automatically harvesting drug effectiveness. As a resource of this task, we build a dataset containing 25K tweets from Twitter, each of which is grounded in the condition that a patient is dealing with some diseases by specific chemicals[1]. To measure drug effectiveness, we define three kinds of **relations**: "better", "worse" and "maintain".[2] A relation **only exists** between *a type of* chemical and *a type of* disease where the drug effectiveness is clearly described: if the chemical works on the disease, there is a "better" relation; if it has no effect, there is a "maintain" relation; if it worsens the disease, there is a "worse" relation. In Table 1, we list some examples. Our goal is to harvest all relations from tweets in which chemical and disease mentions are marked out[3], i.e. a **Relation Extraction (RE)** task. Besides, it is worth mentioning that other datasets related to this task, e.g. the BioCreative V Chemical Disease Relation (CDR) dataset [24, 43], are mostly come from biomedical literatures. They differ from ours (collected from social media) in many aspects, which will be further explained in Section 3.

To better illustrate our task, we define two concepts: a Chemical-Disease bag (**CD-bag**) is composed by *a type of* chemical and *a type of* disease; a Chemical-Disease instance (**CD-instance**) is a mention pair composed by *a certain* chemical mention and *a certain* disease mention. Under these definitions, there is a one-to-many relationship between CD-bags and CD-instances. Considering the tweet (ii) in Table 1: there are two CD-bags and four CD-instances since the same type of disease "cold" is mentioned twice. Furthermore, relations only exist in CD-bags where drug effectiveness is clearly described. For these bags, we give each of them a label from {"better", "worse", "maintain" }. For the rest CD-bags, we label them

[1]Following the convention of biochemical literature, we consider the words "drug" and "chemical" to be interchangeable.
[2]We asked several experts and they all agreed that using the three kinds of relation is acceptable. Our model is still suitable when more fine-grained relations are used.
[3]We do not pay attention to the Name Entity Recognition (NER) task since existing NER models can get pretty good performance: a CNN-LSTM-CRF model [27] can reach to an F1 score over 91%, and a BERT model [11] can reach to an F1 score over 94%.

Table 1: Typical examples in our dataset (we underline chemical mentions display disease mentions in bold).

Tweets	Relations
(i) I started off on Zoloft, I'm going next week to get changed. it helps my **ocd** and **anxiety**, but made my **depression** worse.	**better:** <ocd, Zoloft>, <anxiety, Zoloft> **worse:** <depression, Zoloft>
(ii) No **cold** medicine has ever cured a **cold**. # Mucinex # Robitussin.	**maintain:** <cold, Muxinex>, <cold, Robitussin>
(iii) OK I must sleep now. Despite all the normal meds I take at bedtime I had to add in an Imitrex for the **migraine** I feel coming.	No relation exists (since the effect of "migraine" is not mentioned).

with "none" indicating no relation exists, which is a commonly-used trick in RE researches. In this way, each CD-bag corresponds with one of the four labels depending on all CD-instances in it. Different from fully-supervised classification tasks where a model receives a set of labeled instances, we have to perform Multiple Instance Learning (MIL), where the model receives a set of labeled bags, each containing many instances. MIL is one of the major challenges for RE tasks.

There is a significant difference between our task and others: **interactions between CD-instances are widespread**, e.g. in the tweet (i) from Table 1, all CD-instances share the same chemical "Zoloft". This further makes CD-bags (relations) highly interacted. More specifically, if we randomly pick out a relation in our dataset, there is a probability of **49.4%** that another relation from the same tweet contains the same kind of chemical or disease. The widespread interactions **always appear** in pharmacological-relevant data [41], since people often use multiple chemicals for the same disease and vice versa. However, most existing RE models only deal with mention-pairs independently. Some of them considered the problem, but they either used a simple attention mechanism which is not powerful enough [14], or only captured parts of the interactions [6]. Another idea was focusing on interactions between all mentions [41]. However, this implicit method cannot guarantee that interactions between mention-pairs will be captured as well.

In this paper, we design a model that **simultaneously deals with all CD-instances** to **explicitly capture their interactions**. Our model regards each CD-instance as a node and connects them with multiple types of edges. After that, it performs Graph-based Information Transfers Over Time (GITOT) to capture interactions between all nodes. At every time-step, each node first receives information from its neighbors and then updates its representation by a gated-update mechanism. The final results of GITOT are further used by output layers to perform MIL. Our model can be viewed as a special type of Graph Convolutional Networks (GCNs), which is a certain kind of Graph Neural Networks (GNNs) based on spectral graph theory [4]. However, it differs from other existing GCNs in many aspects as will be explained in Section 4.5.

Above all, the contributions of this paper are threefold:

- We build a new dataset containing 25K tweets as a new benchmark for harvesting drug effectiveness. It is the first dataset based on social media for this task.
- We turn the drug effectiveness discovery task into a RE task and propose a model which is the first to explicitly capture interactions between all CD-instances simultaneously. It also uses a novel idea to perform MIL.
- We perform extensive experiments to show that our model outperforms previous work by a substantial margin. We further illustrate that the model can be viewed as a special kind of GCN, but it differs from others in many aspects.

Dataset and code for this paper are available at https://github.com/ChaiZ-pku/Harvesting-Drug-Effectiveness-from-Social-Media.

2 RELATED WORK

2.1 Pharmacological Information Extraction from Social Media

Data on social media have gained much attention for extracting pharmacological information because of their high volume and availability. Since patient-oriented outcomes are only a small portion of user posts, many researches [9, 18, 22, 33] focused on the classification task to pick them out. Another widely studied topic is pharmacological named entity recognition [8, 15, 39, 45], which aims to find chemical and disease mentions, or further extract descriptions about adverse drug events (ADE).

Recently, harvesting drug effectiveness by performing RE between chemicals and diseases is gradually getting more and more attention [23, 41]. Current studies are mainly based on professional biochemical literatures written by experts. On the other hand, we are the first to leverage user posts from social media.

2.2 Relation Extraction

RE is a heavily studied area in the Natural Language Processing (NLP) and Information Retrieval (IR) community. Traditionally, it heavily relies on human designed rules and features. Recent years, deep learning RE models [32, 35, 47, 49] based on Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs) have taken up the mainstream since they can achieve state-of-the-art performance in an end-to-end way. Despite that we have reached high performance in a number of RE tasks, there are still many challenges in RE.

One major challenge is MIL. As the same kind of mention can occur several times, mention-pairs (also called "instances") can be divided into different "bags". In most cases, instead of having a set of instances which are individually labeled, we only have a set of labeled bags, each containing many instances. As a result, we need to perform MIL. To do so, most existing methods first compute a representation for each instance **independently** and then get bag-representations by directly *picking out* features from instance-representations [17, 46] in the same bag, or *combining* them by a weighted sum [14, 26]. Some recent studies also use adversarial training and reinforcement learning [42, 48]. On the other hand, our model captures the interactions between different instances **into** their representations, which is a new idea of MIL.

Another challenge lies in the interaction between mention-pairs. As most RE algorithms independently deal with each mention-pair, they are unable to capture these interactions. There are some studies focusing on this challenge: Verga et al. [41] capture the interactions between all mentions, but this is only an implicit way and cannot guarantee that the interaction between mention-pairs can be captured as well. Feng et al. [14] use the attention mechanism to explicitly capture interactions between mention-pairs. However, simply computing a weighted sum lacks effective control. Christopoulou et al. [6] combine mention-pair (A, B) and (B, C) to support (A, C), but this can only capture parts of the interactions. Our model takes a further step by putting mention-pairs into a graph and simultaneously capturing the interactions between them. By contrast, the attention mechanism uses a fully-connected graph (too many edges), while the walk based model only uses some of our sub-graphs (too few edges).

2.3 GNN and GCN

GNN is a general architecture defined according to a graph structure. It is suitable for a number of NLP and IR tasks, e.g. semantic role labeling [28], recommender systems [44], etc. In NLP tasks, GNNs are commonly used for modeling syntactic trees or knowledge graphs. However, **no research** has ever used a GNN to **model mention-pairs in RE**, and we are the first to use a graph model to capture interactions between them as far as we know.

In recent years, a certain type of GNN called GCNs have achieved promising results in many tasks previously dominated by kernel-based methods, graph-based regularization techniques, etc. GCNs are based on spectral graph theory, and the basic idea is defining parameterized filters shared over all locations in the graph. As they are computationally expensive, some researches [10, 21] try to use fast heuristic methods to speed up traditional spectral approaches. Others try to introduce problem-specific specialized GCN architectures [2, 25, 28]. Our model can be viewed as a RE-specific specialized GCN, but it differs from others in many aspects.

3 DATASET

We chose Twitter to build our dataset, since it is one of the most commonly used social media containing billions of tweets. As most tweets do not mention any chemical or disease, we built some prior pairs, each containing a chemical name and a disease name, and only collected tweets that contain at least a prior pair. These pairs were chosen from one of the top healthcare websites, WebMD[4], where one can search for suitable chemicals with a certain disease and vice versa. We crawled 2.4M tweets under the guidance of 905 prior pairs at last. We then performed data cleaning by discarding tweets containing URLs (most of them are advertisements), removing non-textual components (links to images or videos, labels about post time, etc.), filtering out tweets which are no more than ten words and eliminating redundant data. After that, three native speakers were hired to pick up task-relevant tweets, mark out chemical, disease mentions and assign each CD-bag a label from {"better", "maintain", "worse", "none"}. We ensured that each tweet was labeled by at least two different people. If there were disagreements, we added another person and adopted the opinions of the majority. CD-bags with "none" labels are called "N-bags", and those tweets which only contain N-bags as "N-tweets" in contrast with "E-tweets", which contains at least a relation. The counts for N-tweets and E-tweets in our dataset are 20K and 2253, respectively.

Since N-tweets took the most part in our dataset, we used all E-tweets and randomly sampled the same number of N-tweets as evaluation dataset for experiments. It contains 4506 tweets with an average length of 27 words, and mentions 400 chemicals and 128 diseases in total. There are 5883 CD-bags and 6620 CD-instances in this dataset (10.6% bags contain more than one instances), and the distribution of bag labels is 45%-none, 37%-better, 34%-maintain and 4%-worse. As we can see, "better" labels are far more than "worse" labels, which is in accordance with the common sense that chemicals work for the most time. However, "worse" labels are more important for discovering side effects. As mentioned above, relations are not independent with each other: if a tweet contains more than one relations, there is a probability of 98.8% that some of them share the same chemical or disease.

It is worth mentioning that our dataset significantly differs from others based on biochemical literatures, e.g. the CDR dataset. People tend to use colloquial expressions on social media, e.g. "Claritin saved my life", "Insomnia sucks", while literatures written by experts are quite formal. In addition, user posts on social media are much shorter. Despite all these differences, interactions across relations are widespread in both kinds of datasets. By the way, although data from social media are noisier, they are much easier to acquire compared with biochemical literatures.

4 ARCHITECTURE

Given a tweet in which chemical and disease mentions are marked out, our model first constructs a graph by regarding CD-instances as nodes and connecting them by multiple types of edges, as shown in Figure 1. After that, it embeds each node and performs Graph-Based Information Transfers Over Time (GITOT) to capture their interactions (especially those for MIL). Finally, it picks up certain node-representations in each bag to predict the final bag-label. The architecture of our model can be viewed from Figure 3.

4.1 Graph Construction from CD-Instances

We segment each tweet into a word-sequence $W = [w_1, w_2, ..., w_n]$ and further use C, $\mathcal{D}$ to denote the **positions** of all chemical and disease mentions[5] in W, respectively. In other words, each $c \in C$ is an integer between $[1, n]$ which can be used to index a chemical mention $w_c = W[c]$. Similarly, $d \in \mathcal{D}$ is the index of a disease mention $w_d = W[d]$. If two mentions w, w' are the same type of chemical or disease, we define $w \overset{\text{type}}{=} w'$.

As mentioned above, we regard CD-instances as nodes connected by multiple types of edges. To decide which type of edge should be used between two CD-instances (w_c, w_d) and $(w_{c'}, w_{d'})$ ($c, c', d, d' \in \{1, 2, ..., n\}$), there are four different cases:

(1) $w_c \overset{\text{type}}{=} w_{c'}, w_d \overset{\text{type}}{=} w_{d'}$. In this case, two CD-instances come from the same CD-bag. We use a B-edge (B means Bag) to connect them, e.g. for nodes 1 and 3 in Figure 1.

[4]https://www.webmd.com

[5]For the few mentions that contain multiple words, e.g. "chromic pain". We regard them as a single word, which is a simple but effective trick.

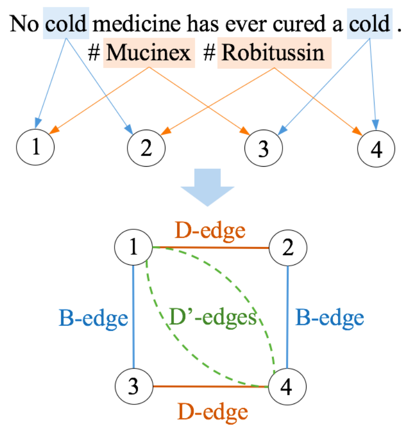

Figure 1: Graph construction from CD-instances. There are two CD-bags: instances (1, 3) and (2, 4).

(2) $w_c \neq w_{c'}, w_d \overset{\text{type}}{=} w_{d'}$. In this case, two CD-instances have the same kind of disease. There are two possibilities:
- $d = d'$, which means two pairs share exactly the same disease mention. We use a D-edge (means Disease) to connect them, e.g. for nodes 1 and 2 in Figure 1.
- $d \neq d'$. This time we use a D'-edge to connect the two nodes, e.g. for nodes 1 and 4 in Figure 1.

In other words, if two nodes are connected by a D-edge, there is a syntactic intersection (sharing the same disease mention); if a D'-edge, there is only a semantic interaction.

(3) $w_c \overset{\text{type}}{=} w_{c'}, w_d \neq w_{d'}$. This is similar with (2). We add a C-edge (means Chemical) or C'-edge, respectively.

(4) $w_c \neq w_{c'}, w_d \neq w_{d'}$. In this case, there is no edge between the two nodes.

After these steps, we construct a graph $G = \{\mathcal{V}, \mathcal{E}\}$, where each node $v_i \in \mathcal{V}$ corresponds with a CD-instance (w_c, w_d) $(c \in \mathcal{C}, d \in \mathcal{D})$, and $e_{ij} \in \mathcal{E}$ is the edge we added between nodes v_i and v_j. Besides, $|\mathcal{V}| = |\mathcal{C}| \times |\mathcal{D}|$ ($|\cdot|$ returns the size of a set).

4.2 Node Embedding Layers

After getting $G = \{\mathcal{V}, \mathcal{E}\}$, we need to embed each node $v_i \in \mathcal{V}$ to perform GITOT, as shown in Figure 2. We first map the tweet $W = [w_1, w_2, ..., w_n]$, a sequence of words, into a sequence of vectors $X = [\boldsymbol{x}_1, \boldsymbol{x}_2, ..., \boldsymbol{x}_n]$ according to node (w_c, w_d). Each $\boldsymbol{x}_k$ $(k = \{1, 2, .., n\})$ is the concatenation of three embeddings of w_k:

- A word embedding $\boldsymbol{q}_{w_k} \in \mathbb{R}^{l_q}$, which captures the syntactic and semantic meanings of w_k under the idea of distributed representations [3]. The CBOW model [31] is used to pre-train a word embedding matrix and look up for $\boldsymbol{q}_{w_k}$. Word embeddings will be fine-tuned during the training process.
- Two position embeddings $\boldsymbol{p}^c_{w_k}, \boldsymbol{p}^d_{w_k} \in \mathbb{R}^{l_p}$. The distances between words and mentions are useful for RE, as proved by many researches. Similar to Zeng et al. [47], we create two position embedding matrices $\boldsymbol{P}_c, \boldsymbol{P}_d$ which are randomly initialized. The relative distances between w_k and chemical mention w_c, disease mention w_d, i.e. $(k - c)$ and $(k - d)$, are transformed into real-valued vectors $\boldsymbol{p}^c_{w_k}, \boldsymbol{p}^d_{w_k}$ by looking up the position embedding matrices, respectively. The position embedding matrices are parameters of our model.

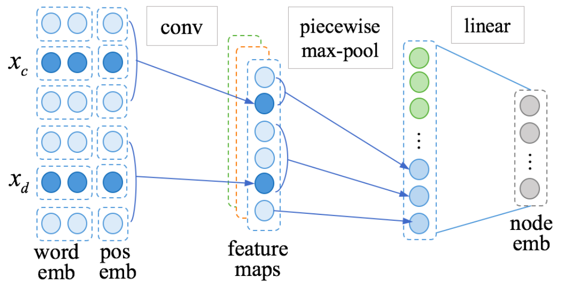

Figure 2: Architecture of a "node embedding unit".

By concatenating $\boldsymbol{p}^c_{w_k}, \boldsymbol{p}^d_{w_k}$ and $\boldsymbol{q}_{w_k}$, we get $\boldsymbol{x}_k \in \mathbb{R}^{l_x}$ $(l_x = l_q + 2 \times l_p)$ corresponding to w_k. In this way, each tweet is turned into a vector-sequence $X = [\boldsymbol{x}_1, \boldsymbol{x}_2, ..., \boldsymbol{x}_n]$, and further regarded as the input of a convolution layer which extracts features by sliding a window (also called mask or kernel) of length l_w over the input sequence. The output of the k-th window is:

$$f(k) = ReLU(\boldsymbol{W}_m \otimes X_{[k-l_w+1:k]} + b_m) \tag{1}$$

where $ReLU$ stands for Rectified Linear Unit, $\boldsymbol{W}_m \in \mathbb{R}^{l_x \times l_w}, b_m \in \mathbb{R}$ are parameters of the mask, $X_{[k-l_w+1:k]}$ denotes the concatenation between l_w vectors from sequence X, and $\otimes$ stands for convolution operation that first performs element-wise matrix multiplication and then sums up all the elements in the product matrix. The outputs of all windows form a feature map $\boldsymbol{f}$. To make $\boldsymbol{f} \in \mathbb{R}^n$, the same padding is adopted, i.e. zero values are padded if a window slides outside of boundaries $(k - l_w + 1 < 1$ or $n < k)$. Besides, multiple filters are also used to get a set of feature maps $\mathcal{F} = \{\boldsymbol{f}\}$.

Instead of using a max-pooling layer and directly picks up the max element from each feature map, the piecewise max-pooling is adopted, which first divides $\boldsymbol{f} \in \mathbb{R}^n$ into three segments according to positions c, d and then selects the max element from each segment. We concatenate the piecewise pooling results on every $\boldsymbol{f} \in \mathcal{F}$ and use a fully-connected layer to get $\boldsymbol{o} \in \mathbb{R}^{l_o}$.

4.3 Information Transfer Layers

4.3.1 Bag Annotations. Before performing GITOT based on node embeddings $\boldsymbol{o}_i \in \mathbb{R}^{l_o} (i = 1, 2, ..., |\mathcal{V}|)$, we find it useful to add additional "bag annotations", a number of one-hot vectors denoting which bag the node belongs to. For the example in Figure 1, there are two CD-bags: (cold, Mucinex) containing CD-instances v_1, v_3 and (cold, Robitussin) containing v_2, v_4. In this case, we append a one-hot vector $[1, 0]^T$ after $\boldsymbol{o}_1$, $\boldsymbol{o}_3$; and another $[0, 1]^T$ after $\boldsymbol{o}_2$, $\boldsymbol{o}_4$. Generally, if there are l_b bags, each node embedding $\boldsymbol{o} \in \mathbb{R}^{l_o}$ is turned into $\boldsymbol{h}^{(0)} \in \mathbb{R}^{l_h}$, where $l_h = l_o + l_b$. Intuitively, we enable nodes to share information by adding different edges, i.e. "drag" them together. By using bag annotations, we distinguish nodes from different CD-bags, i.e. "push" them away. Experiments show that bag annotations are useful for MIL.

4.3.2 GITOT. We now describe the information transfer process, which can be divided into two steps. At time-step t $(t \geq 1)$:

(1) Each node receives information $\boldsymbol{a}_i^{(t)}$ from its neighbors;
(2) Each node uses $\boldsymbol{a}_i^{(t)}$ to update $\boldsymbol{h}_i^{(t-1)}$ and gets $\boldsymbol{h}_i^{(t)}$

To perform step (1), we assign five matrices $\boldsymbol{W}_B, \boldsymbol{W}_C, \boldsymbol{W}_{C'}, \boldsymbol{W}_D, \boldsymbol{W}_{D'} \in \mathbb{R}^{l_h \times l_h}$ and vectors $\boldsymbol{b}_B, \boldsymbol{b}_C, \boldsymbol{b}_{C'}, \boldsymbol{b}_D, \boldsymbol{b}_{D'} \in \mathbb{R}^{l_h}$ to each of the five edge-types B, C, C', D, D' described in Section 4.1, respectively (all the matrices and vectors are parameters of our model). A function $type(e_{ij})$ is further defiened to return the edge-type of a given edge e_{ij}. This function is used to fetch different kinds of matrices and vectors, i.e. in Figure 1, e_{12} is a D-edge, so $\boldsymbol{W}_{type(e_{12})} = \boldsymbol{W}_D$, $\boldsymbol{b}_{type(e_{12})} = \boldsymbol{b}_D$. Similarly, $\boldsymbol{W}_{type(e_{13})} = \boldsymbol{W}_B$, $\boldsymbol{b}_{type(e_{13})} = \boldsymbol{b}_B$. We use $\mathcal{N}(v_i)$ to denote all neighbors of node v_i. In other words, $\mathcal{N}(v_i) = \{v_j\}$ where v_j is adjacent to v_i. In this way, step (1) can be written as:

$$\boldsymbol{a}_i^{(t)} = \sum_{v_j \in \mathcal{N}(v_i)} \boldsymbol{W}_{type(e_{ij})} \boldsymbol{h}_j^{(t-1)} + \boldsymbol{b}_{type(e_{ij})} \tag{2}$$

which means the information transferred from v_j to v_i is controlled by the type of e_{ij}. If no edge exists, there is no transfer.

After getting $\boldsymbol{a}_i^{(t)}$ for node v_i, we can perform step (2) by gated-update:

$$\begin{aligned}
\boldsymbol{z}_i^{(t)} &= \sigma(\boldsymbol{W}_z[\boldsymbol{a}_i^{(t)}; \boldsymbol{h}_i^{(t-1)}]) \\
\boldsymbol{r}_i^{(t)} &= \sigma(\boldsymbol{W}_r[\boldsymbol{a}_i^{(t)}; \boldsymbol{h}_i^{(t-1)}]) \\
\tilde{\boldsymbol{h}}_i^{(t)} &= tanh(\boldsymbol{W}_h[\boldsymbol{a}_i^{(t)}; \boldsymbol{r}_i^{(t)} \odot \boldsymbol{h}_i^{(t-1)}]) \\
\boldsymbol{h}_i^{(t)} &= \boldsymbol{z}_i^{(t)} \odot \boldsymbol{h}_i^{(t-1)} + (1 - \boldsymbol{z}_i^{(t)}) \odot \tilde{\boldsymbol{h}}_i^{(t)}
\end{aligned} \tag{3}$$

where $\odot$ means element-wise multiplication, and the "update gate" $\boldsymbol{z}_i$ and "reset gate" $\boldsymbol{r}_i$ controls how much new information should be added and old information should be forgotten. As we will explain later, the gated-update mechanism is used to preserve original contextual information when new information about node-interactions is added. By the way, using the gate-update mechanism is inspired by its success in RNN [7, 16].

By iteratively performing the information transfer process, a sequence of representations $H_i = [\boldsymbol{h}_i^{(0)}, \boldsymbol{h}_i^{(1)}, ..., \boldsymbol{h}_i^{(T)}]$ is generated for each node v_i. To get a final representation $\boldsymbol{h}_i^{out}$, we adopt the "mean over time" scenario which is efficient for dealing with recurrent outputs [38]. In other words, $\boldsymbol{h}_i^{out} = \boldsymbol{h}_i^{avg}$ where $\boldsymbol{h}_i^{avg} = \sum_{t=1}^{T} \boldsymbol{h}_i^{(t)}$. As we will discuss in Section 5.6, the mean over time scenario can get the same level of performance compared with more complicated scenarios in a simple and elegant way.

4.4 Output Layers

To get bag-labels, we first compute node-predictions by applying a Multi-Layer Perceptron (MLP) followed by a $softmax$ layer on each node-representation $\boldsymbol{h}_i^{out}$:

$$\hat{\boldsymbol{y}}_i = softmax(MLP(\boldsymbol{h}_i^{out})) \tag{4}$$

where $\hat{\boldsymbol{y}}_i \in \mathbb{R}^4$ is a distribution over four labels. We regard the max probability in the distribution, $\hat{y}_i{}^{max} = \max(\hat{\boldsymbol{y}}_i)$, as its confidence. For all nodes in the same bag, we choose the node-prediction with the largest confidence as bag-prediction $\hat{\boldsymbol{y}}_{bag}$, and use the cross entropy between $\hat{\boldsymbol{y}}_{bag}$ and gold standard annotation $\boldsymbol{y}$ as loss function to train the model.

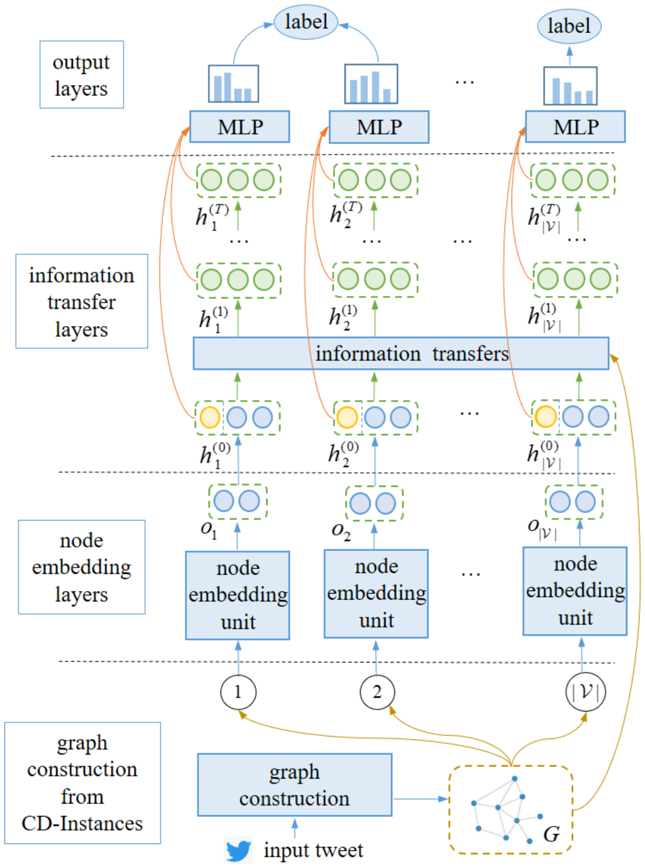

Figure 3: Architecture of our model.

Since each node-representation $\boldsymbol{h}_i^{out}$ is the output of GITOT, they have already captured the interactions between CD-instances, which is important for MIL. To this end, we directly pick out the most confident node-prediction in a bag as outputs. In other words, our model *implicitly* deals with MIL when generating instance-representations. Compared with other models that first deal with each instance independently and then *explicitly* combine instances in the same bag to perform MIL, this is a major difference.

4.5 Comparison with Other GCNs

4.5.1 Similarities. First, we illustrate that our model is a special kind of GCN. For a general GCN, the input is a feature description for every node $\boldsymbol{X}_i \in \mathbb{R}^{N \times D}$ (where N is the number of nodes and D is the number of input features) and a representative description of the graph structure in matrix form $\boldsymbol{A}$ (typically in the form of an adjacency matrix). The output of GCN is another node-level matrix $Z \in \mathbb{R}^{N \times F}$ (where F is the number of output features per node). Every neural network layer of GCN can be written as:

$$\boldsymbol{H}^{(l+1)} = f(\boldsymbol{H}^{(l)}, \boldsymbol{A}) \tag{5}$$

with $\boldsymbol{H}^{(0)} = \boldsymbol{X}$ and $\boldsymbol{H}^{(L)} = Z$, L being the number of layers. For specific GCN models, how f is chosen can be different, but there are always weight-sharing during this process, e.g. Duvenaud et al. [13] used a GCN similar to $f(\boldsymbol{H}^{(l)}, \boldsymbol{A}) = \sigma(\boldsymbol{A}\boldsymbol{H}^{(l)}\boldsymbol{W}^{(l)})$ where $\boldsymbol{W}^{(l)}$ is a weight matrix for the l-th neural network layer and $\sigma(\cdot)$ is a non-linear activation function. Kipf et al. [21] used $f(\boldsymbol{H}^{(l)}, \boldsymbol{A}) = \sigma(\hat{\boldsymbol{D}}^{-\frac{1}{2}}\hat{\boldsymbol{A}}\hat{\boldsymbol{D}}^{-\frac{1}{2}}\boldsymbol{H}^{(l)}\boldsymbol{W}^{(l)})$ with $\hat{\boldsymbol{A}} = \boldsymbol{A} + \boldsymbol{I}$, where $\boldsymbol{I}$ is the identity matrix and $\hat{\boldsymbol{D}}$ is the diagonal node degree matrix of $\hat{\boldsymbol{A}}$. Our model can be viewed as a specific GCN where Equation 5 is expressed by Equation 2 and Equation 3. Each time-step in GITOT corresponds to a specific layer in GCN, and we use the same $\boldsymbol{W}^{(l)}$ for every layer. Besides, parameters in $\boldsymbol{W}^{(l)}$ corresponding to the same edge-type are shared. Finally, $\boldsymbol{A}$ is not only an adjacent matrix, but also contains information about edge-type.

4.5.2 *Differences.*

Next, we illustrate the differences between our model which is **the first GCN in RE**, and other GCNs, especially a certain type called Gated Graph Neural Network (GGNN) [25]. GGNN also distinguishes different types of edges and adopts the gated-update mechanism. It has reached promising results in some NLP tasks [2, 28], but has never been used in RE. Our model differs from other GCNs in many aspects. Some key differences are:

- **The meaning of edges**. In many NLP tasks, edges in a GCN often have specific and explicit meanings, e.g. the graph is constructed from a syntactic tree thus each edge indicates a certain type of syntactic dependency, e.g. "v_i is a subject-verb of v_j". However, edges in our model have obscure and implicit meanings, e.g. "v_i, v_j come from the same CD-bag" does not show how they are interacted. Besides, there are always less types of edges, thus parameter explosion is not a big problem.
- The importance of **preserving initial node-information**. In GGNNs (and most GCNs), the main goal is to capture the structure of a graph. So initial values of each node are always basic features, e.g. a one-hot vector denoting node number [25], word or positional embedding [2], etc, and we expect these features to be transformed through layers into high-level representations. However, when it comes to our model, the initial values of each node contain rich contextual information captured by node embedding units. So there a trade-off between preserving initial contextual features and capturing new interaction information, i.e. we are "mixing" graph features with contextual features instead of "distilling" high-level graph features from low-level features.

Because of these differences, it is a necessity to use gated-update mechanism in our model, which can carefully add new information without forgetting too much contextual information. In addition, instead of directly picking up the top-layer's output, we combine representations in all layers to preserve more contextual information, which does not often appear in other GCNs.

5 EXPERIMENTS

In this section, we perform experiments to evaluate our model. We first introduce baselines, evaluation metrics and experimental details. Then, we compare the performance of different models and analysis the experimental results.

5.1 Experiments Setup

We compare our model with eight typical baselines.

- **Zeng2015** [46] first regarded distant supervised RE as a MIL problem and proposed piecewise CNN to avoid feature engineering. Both of the two ideas have greatly affected a number of successors.
- **Jiang2016** [17] relaxed the at-least-once assumption in Zeng 2015 and employed cross-sentence max-pooling to enable information sharing across different sentences.
- **Lin2016** [26] proposed a model using sentence-level attention mechanism. It dealt with MIL by automatically reducing the attention weights of noisy instances in a bag.
- **Zhou2016** [49] adopted the word-level attention-based bidirectional Long Short-Term Memory Network (LSTM) for supervised RE. We further added sentence-level attention similar to Zeng et al. [46] for MIL, thus a hierarchical attention architecture was adopted.
- **Feng2017** [14] combined piecewise CNN with two memory networks, which can be regarded as a generalized attention mechanism, to capture the importance of each context word for modeling the representation of mentions, as well as the intrinsic dependencies between relations.
- **Chen2017** [5] proposed a framework based on multiple-attention mechanism for aspect sentiment analysis. Intuitively, the effect of a drug "better, worse, maintain, none" can correspond to different sentiments polarities of the patient: "positive (satisfied), negative (displeasure), neutral and none". So we used this model as one of the baselines and added sentence-level attention proposed by Zeng et al. [46] to deal with MIL.
- **Chris2018** [6] was the first model to explicitly consider the interactions between different relations. By walking trough mention-pair (A, B) and (B, C), it can better deal with (A, C). In our task, relations between two chemicals (or diseases) are not considered, so we modified the loss function to adjust it for our task.
- **Verga2018** [41] was the first model trying to deal with all relations simultaneously to capture their interactions. By leveraging the transformer [40] architecture based on self-attention mechanism, it considered the interactions between all mentions, and expected it to implicitly capture the connection between mention-pairs.

Since F1 score was widely adopted by prior work, we chose it as evaluation metric (we mainly used micro-F1 because of the imbalanced label distribution). However, evaluation metrics based on the whole dataset were unable to describe the ability of a model to capture the interactions between mention-pairs. Based on the observation that if more CD-instances appear in the same tweet, there are more complicated interactions, we divided the whole dataset into different subsets according to the number of CD-instances in each tweet. We computed micro-F1 on different subsets, denoted by $\mathrm{F1}_k$ where k is the number of CD-instances. When k was large, some subsets contained too few tweets thus F1 values were greatly affected by the result of a single data. To this end, we combined them together and introduced $\mathrm{F1}_{>k}$. Intuitively, it is more difficult for a model to get the same level of $\mathrm{F1}_k$ on larger k.

To get each F1 score, we used the ten-fold cross validation. First, we randomly partitioned our evaluation dataset into ten subsets

Table 2: Experimental Results. The highest F1 value in each column is in bold and that of the baselines is underlined.

Methods	Mechanism	$F1_1$	$F1_2$	$F1_3$	$F1_4$	$F1_{\geq 5}$	F1	$F1_{>1}$
Zeng2015	PCNN + Simple MIL	83.32	68.66	53.92	50.47	21.52	72.79	54.73
Jiang2016	PCNN + Cross Sentence Max-Pool	83.42	69.41	56.54	49.53	22.31	73.11	55.42
Lin2016	PCNN + Sentence Level Attention	<u>**83.96**</u>	69.32	58.82	50.24	22.83	73.64	55.93
Zhou2016	LSTM + Hiarachical Attention	77.88	69.13	58.17	53.07	23.10	69.95	56.35
Chen2017	LSTM + Multiple Attention	77.83	69.89	59.48	52.59	22.57	70.05	56.71
Feng2017	PCNN + Memory Network	83.40	<u>70.55</u>	57.19	52.59	30.71	<u>74.09</u>	58.14
Chris2018	LSTM + Walk Aggregation	79.92	69.60	<u>65.69</u>	<u>55.19</u>	<u>34.12</u>	72.58	<u>59.99</u>
Verga2018	Transformer + CNN	77.23	68.47	58.17	50.00	21.26	69.08	55.10
Ours	GITOT	83.40	**72.92**	**71.24**	**62.03**	**51.97**	**77.31**	**66.87**

with equal size. At each fold, every subset was used exactly once as the test set, and the remaining data were randomly split into 8:1 as training and development sets. In this way, we trained and evaluated our model for ten times and averaged all ten results to get the final F1 scores.

5.2 Details of our model

5.2.1 Hyper-parameters. For embedding layers, we per-trained 128-dimensional word embeddings using the word2vec toolkit[6]. During this process, the NLTK[7] package was used for sentence splitting and word tokenization. To get each $\boldsymbol{x}_i$ which is the input of a node embedding unit, we concatenated the pre-trained word embedding and two 32-dimensional position embeddings. When it comes to the convolution layer, we used filters whose l_x ranged from 1 to 5, and the counts for different filters were 32, 64, 64, 32, 16. After each embedding unit, the dimension of a node embedding was 96. For information transfer layers, we chose $l_h = 128$ and set the depth T to 3. Finally, the MLP on top of the model contained a hidden layer with 32 neurons. Their activation functions were leaky *ReLU*.

5.2.2 Optimization. We used an Adam optimizer [20] with momentums $\beta_1 = 0.9, \beta_2 = 0.99$ and $\epsilon = 10^{-8}$ to minimize the loss function. The learning rate was 0.0003, and we applied dropout [37] between 0.4 and 0.5 on the output of each sub-layer to prevent over-fitting. The model was trained on an Nvidia TITAN X graphics card.

5.3 Results

In Table 2, we report the evaluation results of our methods and all baselines computed by ten-fold cross validation (we divide all baselines into two parts: **Feng2017**, **Chris2018** and **Verga2018** that considers the interactions between mention-pairs, and others that deal with each mention-pair indepently). We use two kinds of F1 scores introduced in Section 5.1:

(1) Overall metrics. F1 is used to evaluate the performance on the whole test set, and $F1_{>1}$ illustrates the model's ability to capture interactions between CD-instances.
(2) Specific metrics. As mentioned above, we divided our test set into different subsets to compute each $F1_k$. When $k \geq 5$, there were too few tweets in each subset so we combined them together to report $F1_{\geq 5}$. In our evaluation dataset, the five subsets contain 3716, 528, 102, 106, 54 tweets and 3716, 1056, 306, 424, 381 CD-instances, respectively.

We can draw some interesting conclusions from Table 2. First, our model has significant advantages. When it comes to overall metrics, it outperforms all baselines by a substantial margin (+3.22% on F1, +6.88% on $F1_{>1}$), especially for $F1_{>1}$. More specifically, under the evaluation of $F1_1$, our method has a similar performance compared with those based on PCNNs (**Zeng2015**, **Jiang2016**, **Lin2016**, **Feng2017**). In this case, there is only one CD-instance thus GITOT does not work at all. When there are more CD-instances in the same tweet, our model gradually shows an increasing advantage (+2.37% in $F1_2$, +5.55% in $F1_3$, +6.84% in $F1_4$, +17.85% in $F1_{\geq 5}$) compared with the baseline which has the best performance, respectively. In conclusion, our model gets the state-of-art performance on this task by better capturing interactions between CD-instances.

Second, PCNN is more powerful to deal with simple interactions, while LSTM is better to capture complicated contextual dependencies. We can divide models in Table 2 into two parts according to how they embed CD-instances. On the one hand, PCNN architecture is used by **Zeng2015**, **Jiang2016**, **Lin2016**, **Feng2017** and **Ours**. On the other hand, **Zhou2016**, **Chen2017** and **Chris2018** choose the bidirectional LSTM to embed different CD-instances. Since **Verga2018** embeds all mentions simultaneously by a transformer and there is no explicit embedding of CD-instances, we do not take it into consideration. As we can see, PCNN-based models get much better performance on $F1_1$ where there are no interactions. However, as more CD-instances occur in the same tweet, the interactions make the contextual dependencies complicated. In this case, the performance of PCNN-based models significantly goes down, while LSTM-based models can often get better performance.

Third, models that consider the interactions between mention-pairs can often get better results. The performance of **Chris2018** and **Feng2017** are often better than those performing RE on mention-pairs independently (especially in $F1_{>1}$). However, **Verga2018** has rather poor performance. We think the reason is that interactions between mentions are very different from those of mention-pairs in our task. Besides, the difference between **Chris2018** and **Feng2017** is in accordance with the second conclusion: when there are fewer CD-instances in the same tweet, **Feng2017** based on PCNN has

[6]https://code.google.com/archive/p/word2vec/
[7]https://www.nltk.org/

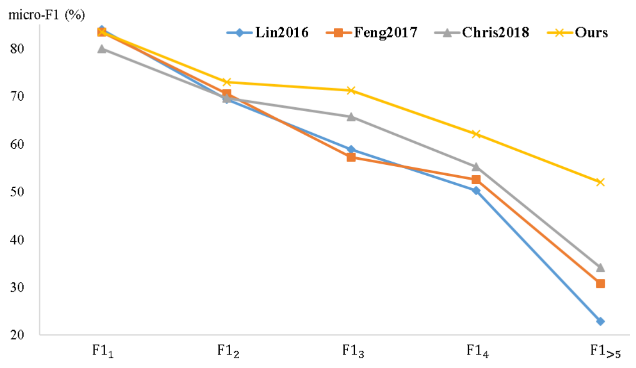

Figure 4: Comparison of our model and strong baselines.

better performance. When there are more CD-instances, i.e. more complicated contextual dependencies, **Chris2018** based on LSTM becomes more powerful.

By the way, for the same model, if there is a larger k, F1$_k$ is always smaller, indicating that the tweets with more CD-instances are harder to deal with, which is in accordance with our intuition.

In Figure 4, we pick three strong baselines and compare them with our model to show these conclusions more clearly:

- **Lin2016**, which has better performance compared with other baselines dealing with CD-instances independently.
- **Feng2017**, which is the most powerful model among PCNN-based methods.
- **Chris2018**, which shows great advantages over all other LSTM-based methods.

As we can see, our model has high performance on almost all evaluation metrics, as introduced in the first conclusion. For the second conclusion, PCNN-based methods (**Lin2016** and **Feng2017**) have high performance when k is small, but they drop sharply as k grows. On the other hand, LSTM-based method (**Chris2018**) has high performance when $k \geq 3$. For the third conclusion, **Feng2017** is more powerful than **Lin2016**, since it takes the interactions between mention-pairs into consideration. Finally, the F1 of all methods goes down as k grows, showing the task is becoming more difficult as node interactions are more complicated. However, our model is much more robust with the change of k.

5.4 Performance on Each Label

As there is an uneven distribution over the four labels, it is necessary to evaluate on different labels. In Table 3, the four columns "none", "better", "maintain" and "worse" show the F1 scores on corresponding labels, respectively. By the way, the counts for each label in our evaluation dataset is 2666, 2179, 818 and 220, respectively. As we can see, it is more difficult for a model to deal with the labels which occur less often (especially the "worse" label). The "macro" column is the average of four F1 scores, which is exactly the macro-F1, and the "weight" column is the weighted average of the four F1 scores, where the weight of each label is decided by its proportion in the test set.

By comparing different rows, we first notice that our model still has clear advantages in most cases. Besides, models based

Table 3: Performance on different labels. (The highest F1 value in each column is in bold and that of the baselines is underlined)

Models	none	better	maintain	worse	macro	weight
Zeng2015	79.28	72.75	60.15	8.96	55.28	71.57
Jiang2016	80.72	73.30	60.61	9.35	55.99	72.50
Lin2016	80.41	74.03	**60.97**	13.95	57.34	72.86
Zhou2016	72.37	72.48	59.42	23.33	56.90	68.78
Chen2017	74.66	72.46	56.79	21.05	56.24	69.35
Feng2017	80.76	73.56	60.02	19.18	58.38	72.91
Chris2018	77.06	75.86	58.95	24.14	59.00	72.12
Verga2018	69.51	75.21	55.46	4.55	51.18	67.24
Ours	**84.90**	**79.04**	56.54	**31.47**	**62.99**	**76.79**

on PCNN are often more powerful on labels which take up large proportions (e.g. the "none" label). However, when it comes to "worse" label which occurs least often, LSTM-based methods can get better results. Finally, models taking the interactions between CD-instances into consideration can often get better performance, which is in accordance with the conclusion drawn from Table 2.

5.5 Ablation Tests

We perform ablation tests to show how B-edges and bag annotations influence the performance of MIL. First, we replace B edges by corresponding C, C', D, D' edges. Second, we remove bag annotations and directly use each node embedding for information transfer. To further show its impact, we use "noisy bag annotations", which means we randomly use a one-hot vector for each node, to train our model. Finally, we train our model based on bag-annotations but use noisy bag annotations during the test stage. The results of ablation tests are shown in Table 4.

As we can see, no matter whether we replace B-edges or remove bag annotations, the performance becomes worse (except F1$_1$ where MIL is not a problem). Besides, if we use noisy bag annotations to train our model, the results are similar to removing bag annotations, indicating that our model automatically ignores noisy information. However, if we only add noisy bag annotations during the test stage, we get the worst performance since the model tries to leverage wrong information from noisy bag annotations in this case.

5.6 Different Scenarios of GITOT

As mentioned above, there is a trade-off in our model between preserving contextual features and capturing interaction information. The depth of GITOT, T, is closely connected with it, since:

- By using larger T, we capture longer-distance interactions since information can propagate further.
- By using smaller T, we better preserve the contextual information captured by node embedding units with less update.

In Figure 5, we show the performance of our model under different T based on two scenarios: "mean" denotes the mean over time scenario where $\boldsymbol{h}_i^{out} = \boldsymbol{h}_i^{avg}$, "last" means we get $\boldsymbol{h}_i^{out}$ by directly

Table 4: Ablation tests.

Scenario	$F1_1$	$F1_2$	$F1_3$	$F1_4$	$F1_{\geq 5}$	F1	$F1_{>1}$
no *B*-edges	**83.45**	71.50	68.30	60.14	47.24	76.49	64.56
no bag-annotations	83.37	71.97	70.26	60.85	48.29	76.75	65.39
noisy bag-annotations (train)	83.07	70.64	66.67	58.73	46.72	75.88	63.54
noisy bag-annotations (test)	82.06	70.35	65.31	55.00	35.94	73.53	60.58
Ours	83.40	**72.92**	**71.24**	**62.03**	**51.97**	**77.31**	**66.87**

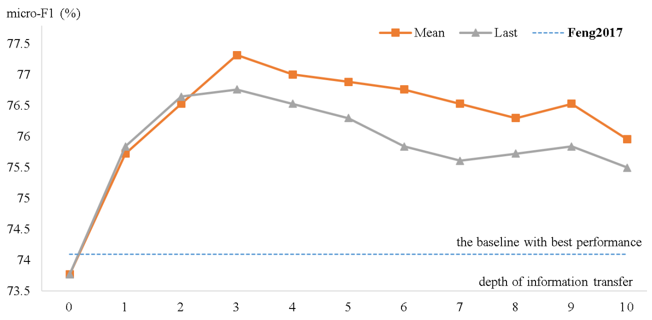

Figure 5: Micro-F1 under different depth of GITOT.

picking out the last term $\boldsymbol{h}_i^{(T)} \in H_i$ (as most GCNs do). By individually looking at each curve, we find that F1 first goes up as T grows from 0, since the model is gradually capturing longer-distance node-interactions without losing too much contextual information. The model can get the best performance when $T = 3$. After that, F1 goes down because too much original information is forgotten. By comparing the two scenarios, we notice that they can get similar performance when $T < 3$. However, as the mean-scenario suffers less from forgetting contextual information, it shows a clear advantage when $T > 2$.

By the way, we can use other ways to get $\boldsymbol{h}_i^{out}$ from H_i. For example, introducing a decay rate λ $(0 \leq \lambda \leq 1)$ and performing $\boldsymbol{h}_i^{out} = \sum_{t=1}^{T} \lambda^{(t-1)} \boldsymbol{h}_i^{(t)}$. We can also regard each H_i as the input of another network (e.g. an LSTM). However, experiments show that the mean over time scenario always reaches high performance in a simple way.

5.7 Comparison of Different Edges

As shown in Equation 2, the information received by node v_i from v_j is controlled by the type of edge e_{ij}. If we use $||\cdot||_2$ to denote the L2-norm, we have:

$$||\boldsymbol{W}\ \boldsymbol{x} + \boldsymbol{b}||_2 \leq ||\boldsymbol{W}||_2 \cdot ||\boldsymbol{x}||_2 + ||\boldsymbol{b}||_2 \qquad (6)$$

indicating that in Equation 2, larger $||\boldsymbol{W}_{type(e_{ij})}||_2$ and $||\boldsymbol{b}_{type(e_{ij})}||_2$ can make the expectation value $E[||\boldsymbol{W}_{type(e_{ij})}\, \boldsymbol{h}_j^{(t-1)} + \boldsymbol{b}_{type(e_{ij})}||_2]$ larger, i.e. more information is expected to transfer from v_j to v_i. To this end, we can use the L2-norm of $\boldsymbol{W}$, $\boldsymbol{b}$ corresponding to each edge-type to reflect the intensity of information transfer.

In Table 5, we show the L2-norm of parameters corresponding to five different edge-types. As we can see, *B*-edges have the largest norm, pointing out that the intensity of information transfer between nodes in the same bag is the strongest as we expected. Besides, C and C' edges have larger norms than D and D' edges since there are 400 kinds of chemicals but only 128 kinds of diseases in our evaluation dataset. Finally, compared with C, D edges, C', D' edges have smaller norms, respectively. This is in accordance with our intuition that the two nodes connected by C', D' edges are not physically interacted.

Table 5: L2-norms of parameters corresponding to different edge-types.

edge-type	B	C	C'	D	D'
W-norm	2.54	2.43	2.23	2.11	2.06
b-norm	0.74	0.62	0.43	0.54	0.26

6 CONCLUSIONS AND FUTURE WORK

In this paper, we have studied how to leverage user posts from social media to harvest drug effectiveness by performing RE between chemicals and diseases. This is of great importance for drug safety, but current surveillance systems which heavily rely on human experts are far from enough. As a resource of this task, we build the first dataset based on social media and compare it with existing datasets coming from biochemical literatures to show its necessity. Since relations between CD-instances are highly interacted and it is not carefully considered by most RE algorithms, we design a model which can be viewed as a specific GCN, to simultaneously deal with all CD-instances and explicitly capture their interactions. Our model also uses a novel idea for MIL, a big challenge in RE. Extensive experiment results prove that our model significantly outperforms previous work by a substantial margin.

For further work, it would be interesting to enlarge the dataset and put our model in real-word use. In this way, we can compare its performance with real-world surveillance systems. Besides, as "worse" labels are more important for drug safety but they always appear least often, finding GCN architectures that are more powerful to deal with "worse" labels is also worth exploration.

ACKNOWLEDGMENTS

This work was supported by National Natural Science Foundation of China (61772036) and Key Laboratory of Science, Technology and Standard in Press Industry (Key Laboratory of Intelligent Press Media Technology). We thank the anonymous reviewers for their helpful comments. Xiaojun Wan is the corresponding author.

REFERENCES

[1] David W Bates, R Scott Evans, Harvey Murff, Peter D Stetson, Lisa Pizziferri, and George Hripcsak. 2003. Detecting adverse events using information technology. *Journal of the American Medical Informatics Association* 10, 2 (2003), 115–128.

[2] Daniel Beck, Gholamreza Haffari, and Trevor Cohn. 2018. Graph-to-sequence learning using gated graph neural networks. *arXiv preprint arXiv:1806.09835* (2018).

[3] Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Jauvin. 2003. A neural probabilistic language model. *Journal of machine learning research* 3, Feb (2003), 1137–1155.

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

[5] Peng Chen, Zhongqian Sun, Lidong Bing, and Wei Yang. 2017. Recurrent attention network on memory for aspect sentiment analysis. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*. 452–461.

[6] Fenia Christopoulou, Makoto Miwa, and Sophia Ananiadou. 2018. A Walk-based Model on Entity Graphs for Relation Extraction. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, Vol. 2. 81–88.

[7] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555* (2014).

[8] Anne Cocos, Alexander G Fiks, and Aaron J Masino. 2017. Deep learning for pharmacovigilance: recurrent neural network architectures for labeling adverse drug reactions in Twitter posts. *Journal of the American Medical Informatics Association* 24, 4 (2017), 813–821.

[9] Hong-Jie Dai, Musa Touray, Jitendra Jonnagaddala, and Shabbir Syed-Abdul. 2016. Feature engineering for recognizing adverse drug reactions from twitter posts. *Information* 7, 2 (2016), 27.

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

[11] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805* (2018).

[12] Jeffrey M Drazen et al. 2007. *Adverse drug event reporting: the roles of consumers and health-care professionals: workshop summary*. Natl Academy Pr.

[13] David K Duvenaud, Dougal Maclaurin, Jorge Iparraguirre, Rafael Bombarell, Timothy Hirzel, Alán Aspuru-Guzik, and Ryan P Adams. 2015. Convolutional networks on graphs for learning molecular fingerprints. In *Advances in neural information processing systems*. 2224–2232.

[14] Xiaocheng Feng, Jiang Guo, Bing Qin, Ting Liu, and Yongjie Liu. 2017. Effective deep memory networks for distant supervised relation extraction. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI*. 19–25.

[15] Shashank Gupta, Sachin Pawar, Nitin Ramrakhiyani, Girish Palshikar, and Vasudeva Varma. 2017. Semi-Supervised Recurrent Neural Network for Adverse Drug Reaction Mention Extraction. *arXiv preprint arXiv:1709.01687* (2017).

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

[17] Xiaotian Jiang, Quan Wang, Peng Li, and Bin Wang. 2016. Relation extraction with multi-instance multi-label convolutional neural networks. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*. 1471–1480.

[18] JITENDRA Jonnagaddala, Toni Rose Jue, and Hong-Jie Dai. 2016. Binary classification of Twitter posts for adverse drug reactions. In *Proceedings of the Social Media Mining Shared Task Workshop at the Pacific Symposium on Biocomputing, Big Island, HI, USA*. 4–8.

[19] Peter Jüni, Douglas G Altman, and Matthias Egger. 2001. Assessing the quality of controlled clinical trials. *Bmj* 323, 7303 (2001), 42–46.

[20] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

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

[22] Kathy Lee, Ashequl Qadir, Sadid A Hasan, Vivek Datla, Aaditya Prakash, Joey Liu, and Oladimeji Farri. 2017. Adverse drug event detection in tweets with semi-supervised convolutional neural networks. In *Proceedings of the 26th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 705–714.

[23] Haodi Li, Ming Yang, Qingcai Chen, Buzhou Tang, Xiaolong Wang, and Jun Yan. 2018. Chemical-induced disease extraction via recurrent piecewise convolutional neural networks. *BMC medical informatics and decision making* 18, 2 (2018), 60.

[24] Jiao Li, Yueping Sun, Robin J Johnson, Daniela Sciaky, Chih-Hsuan Wei, Robert Leaman, Allan Peter Davis, Carolyn J Mattingly, Thomas C Wiegers, and Zhiyong Lu. 2016. BioCreative V CDR task corpus: a resource for chemical disease relation extraction. *Database* 2016 (2016).

[25] Yujia Li, Daniel Tarlow, Marc Brockschmidt, and Richard Zemel. 2015. Gated graph sequence neural networks. *arXiv preprint arXiv:1511.05493* (2015).

[26] Yankai Lin, Shiqi Shen, Zhiyuan Liu, Huanbo Luan, and Maosong Sun. 2016. Neural relation extraction with selective attention over instances. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Vol. 1. 2124–2133.

[27] Xuezhe Ma and Eduard Hovy. 2016. End-to-end sequence labeling via bi-directional lstm-cnns-crf. *arXiv preprint arXiv:1603.01354* (2016).

[28] Diego Marcheggiani and Ivan Titov. 2017. Encoding sentences with graph convolutional networks for semantic role labeling. *arXiv preprint arXiv:1703.04826* (2017).

[29] John Ernest Marley. 2000. Efficacy, effectiveness, efficiency. (2000).

[30] Graham S May, David L DeMets, Lawrence M Friedman, Curt Furberg, and Eugene Passamani. 1981. The randomized clinical trial: bias in analysis. *Circulation* 64, 4 (1981), 669–673.

[31] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781* (2013).

[32] Thien Huu Nguyen and Ralph Grishman. 2015. Relation extraction: Perspective from convolutional neural networks. In *Proceedings of the 1st Workshop on Vector Space Modeling for Natural Language Processing*. 39–48.

[33] Vassilis Plachouras, Jochen L Leidner, and Andrew G Garrow. 2016. Quantifying self-reported adverse drug events on Twitter: signal and topic analysis. In *Proceedings of the 7th 2016 International Conference on Social Media & Society*. ACM, 6.

[34] Hariprasad Sampathkumar, Xue-wen Chen, and Bo Luo. 2014. Mining adverse drug reactions from online healthcare forums using hidden Markov model. *BMC medical informatics and decision making* 14, 1 (2014), 91.

[35] Cicero Nogueira dos Santos, Bing Xiang, and Bowen Zhou. 2015. Classifying relations by ranking with convolutional neural networks. *arXiv preprint arXiv:1504.06580* (2015).

[36] Amit G Singal, Peter DR Higgins, and Akbar K Waljee. 2014. A primer on effectiveness and efficacy trials. *Clinical and translational gastroenterology* 5, 1 (2014), e45.

[37] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research* 15, 1 (2014), 1929–1958.

[38] Kaveh Taghipour and Hwee Tou Ng. 2016. A neural approach to automated essay scoring. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*. 1882–1891.

[39] Elena Tutubalina and Sergey Nikolenko. 2017. Combination of deep recurrent neural networks and conditional random fields for extracting adverse drug reactions from user reviews. *Journal of Healthcare Engineering* 2017 (2017).

[40] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*. 5998–6008.

[41] Patrick Verga, Emma Strubell, and Andrew McCallum. 2018. Simultaneously Self-Attending to All Mentions for Full-Abstract Biological Relation Extraction. *arXiv preprint arXiv:1802.10569* (2018).

[42] Xiaozhi Wang, Xu Han, Yankai Lin, Zhiyuan Liu, and Maosong Sun. 2018. Adversarial Multi-lingual Neural Relation Extraction. In *Proceedings of the 27th International Conference on Computational Linguistics*. 1156–1166.

[43] Chih-Hsuan Wei, Yifan Peng, Robert Leaman, Allan Peter Davis, Carolyn J Mattingly, Jiao Li, Thomas C Wiegers, and Zhiyong Lu. 2016. Assessing the state of the art in biomedical relation extraction: overview of the BioCreative V chemical-disease relation (CDR) task. *Database* 2016 (2016).

[44] Le Wu, Peijie Sun, Richang Hong, Yanjie Fu, Xiting Wang, and Meng Wang. 2018. SocialGCN: An Efficient Graph Convolutional Network based Model for Social Recommendation. *arXiv preprint arXiv:1811.02815* (2018).

[45] Andrew Yates, Nazli Goharian, and Ophir Frieder. 2015. Extracting Adverse Drug Reactions from Social Media.. In *AAAI*, Vol. 15. 2460–2467.

[46] Daojian Zeng, Kang Liu, Yubo Chen, and Jun Zhao. 2015. Distant supervision for relation extraction via piecewise convolutional neural networks. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. 1753–1762.

[47] Daojian Zeng, Kang Liu, Siwei Lai, Guangyou Zhou, and Jun Zhao. 2014. Relation classification via convolutional deep neural network. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*. 2335–2344.

[48] Xiangrong Zeng, Shizhu He, Kang Liu, and Jun Zhao. 2018. Large scaled relation extraction with reinforcement learning. *Relation* 2 (2018), 3.

[49] Peng Zhou, Wei Shi, Jun Tian, Zhenyu Qi, Bingchen Li, Hongwei Hao, and Bo Xu. 2016. Attention-based bidirectional long short-term memory networks for relation classification. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, Vol. 2. 207–212.