Phar-LSTM: a pharmacological representation-based LSTM network for drug–drug interaction extraction

Mingqing Huang[1,2], Zhenchao Jiang[2] and Shun Guo[2]

[1] School of Software Engineering, Shenzhen Institute of Information Technology, Shenzhen, Guangdong, China
[2] Shenzhen Institute of Advanced Technology, Chinese Academy of Sciences, Shenzhen, Guangdong, China

ABSTRACT

Pharmacological drug interactions are among the most common causes of medication errors. Many different methods have been proposed to extract drug–drug interactions from the literature to reduce medication errors over the last few years. However, the performance of these methods can be further improved. In this paper, we present a Pharmacological representation-based Long Short-Term Memory (LSTM) network named Phar-LSTM. In this method, a novel embedding strategy is proposed to extract pharmacological representations from the biomedical literature, and the information related to the target drug is considered. Then, an LSTM-based multi-task learning scheme is introduced to extract features from the different but related tasks according to their corresponding pharmacological representations. Finally, the extracted features are fed to the SoftMax classifier of the corresponding task. Experimental results on the DDIExtraction 2011 and DDIExtraction 2013 corpuses show that the performance of Phar-LSTM is competitive compared with other state-of-the-art methods. Our Python implementation and the corresponding data of Phar-LSTM are available by using the DOI 10.5281/zenodo.8249384.

Subjects Bioinformatics, Data Mining and Machine Learning

Keywords Pharmacological representation, Long short-term memory, Multi-task learning, Drug–drug interaction extraction

Submitted 7 February 2023
Accepted 15 November 2023
Published 14 December 2023

Corresponding author
Shun Guo, shun.guo1@siat.ac.cn

Academic editor
Dezső Módos

Additional Information and Declarations can be found on page 18

DOI **10.7717/peerj.16606**

INTRODUCTION

Identifying unknown drug interactions is of great benefit for the early detection of adverse drug reactions. In Europe and the USA, adverse drug reactions cause about 300,000 deaths annually (*Zhang, Leng & Liu, 2020*). A Drug–Drug Interaction (DDI) is a situation in which the effects of one drug are changed by the presence of another drug, and it is an important subset of adverse drug reactions (*Brown & Winterstein, 2019*; *Lin et al., 2022*; *Cao et al., 2021*; *Karbownik et al., 2020*). Therefore, detecting DDIs from the biomedical literature can be of great benefit for public health safety.

DDI extraction tasks can be typically divided into coarse-grained tasks and fine-grained tasks. A coarse-grained task aims to predict whether a pair of target drugs has a DDI, whereas a fine-grained task further distinguishes the specific type of the DDI. To address the DDI extraction problem, several platforms, such as the DDIExtraction 2011 (coarse-grained task) (*Segura-Bedmar, Martínez Fernández & Sánchez Cisneros, 2011*) and

DDIExtraction 2013 (fine-grained task) (*Segura-Bedmar, Martínez Fernández & Herrero Zazo, 2013*) challenges have been proposed for evaluating the DDI extraction performance of different methods.

In recent years, various methods (*Asada, Miwa & Sasaki, 2017*; *Björne, Kaewphan & Salakoski, 2013*; *Bobić, Fluck & Hofmann, 2013*; *Bokharaeian & Díaz, 2013*; *Chowdhury & Lavelli, 2013*; *Sánchez Cisneros, 2013*; *Hailu, Hunter & Cohen, 2013*; *Huang et al., 2017*; *Jiang, Gu & Jiang, 2017*; *Qian et al., 2022*; *Liu et al., 2016*; *Rastegar-Mojarad, Boyce & Prasad, 2013*; *Sahu & Anand, 2018*; *Kim et al., 2015*; *Thomas et al., 2013*; *Zhao et al., 2016*; *Chen et al., 2016b*; *Chen et al., 2016a*; *Chen et al., 2020*; *Peng et al., 2022*) have been developed for DDI extraction. These studies can be roughly divided into two periods: The support vector machine (SVM) period and the deep learning period.

Before 2016, most methods were based on SVMs and focused on feature engineering and kernel crafting (*Björne, Kaewphan & Salakoski, 2013*; *Bobić, Fluck & Hofmann, 2013*; *Bokharaeian & Díaz, 2013*; *Chowdhury & Lavelli, 2013*; *Sánchez Cisneros, 2013*; *Hailu, Hunter & Cohen, 2013*; *Rastegar-Mojarad, Boyce & Prasad, 2013*; *Thomas et al., 2013*). For example, FBK-irst (*Chowdhury & Lavelli, 2013*) is a two-stage method that employs a hybrid kernel to detect DDIs and then assign each of the DDIs to one of the four types, wherein the hybrid kernel makes use of shallow linguistic information, a syntactic tree, and manually defined features. *Kim et al. (2015)* proposed a two-stage method based on a linear SVM that used rich features, such as a word feature, word-pair feature, parse-tree feature, and noun phrase constrained on coordination feature. NLLSS (*Chen et al., 2016b*) predicts potential synergistic drug combinations by integrating various types of information, including known synergistic drug combinations, drug-target interactions, and drug chemical structures, thereby enhancing treatment efficacy and reducing the need for high drug dosages to mitigate toxicity. *Chen et al. (2016a)* explored the future directions of network-based drug discovery and the network approach for personalized drug discovery by summarizing databases and web servers involved in drug-target identification and drug discovery processes. One main limitation of these methods is that their performance is largely dependent on the choice of the features.

After 2016, many deep learning-based methods were proposed to automatically extract the feature representations instead of manual feature engineering. Convolutional neural networks (CNNs) and Long Short-Term Memory networks (LSTMs) have been extensively applied by researchers. Representative CNN-based methods include naïve CNN (*Liu et al., 2016*), two-stage syntactic CNN (*Zhao et al., 2016*), and Attention CNN (*Asada, Miwa & Sasaki, 2017*). With respect to LSTM-based methods, many different models have been proposed, such as joint AB-LSTM (*Sahu & Anand, 2018*), two-stage LSTM (*Huang et al., 2017*), Skeleton-LSTM (*Jiang, Gu & Jiang, 2017*), and Attentive LSTM (*Qian et al., 2022*). By reviewing four experimental techniques utilized in recent years to search for small-molecule inhibitors of miRNAs, as well as three distinct models for predicting small molecule-miRNA associations from various perspectives, *Chen et al. (2020)* explored significant publicly accessible databases and web servers containing experimentally validated or potential associations. DAESTB (*Peng et al., 2022*) introduces a cutting-edge computational method for predicting associations between small molecules and miRNAs.

This innovative approach integrates small molecule–small molecule similarity, miRNA–miRNA similarity, and known small molecule–miRNA associations into a high-dimensional feature matrix, leveraging a deep autoencoder and a scalable tree boosting model. Generally, these deep learning-based methods achieve higher performance than traditional SVM-based methods while requiring fewer handcrafted features. Many of these methods adopt the embedding strategy (*i.e.,* map the text information to high-dimensional vectors) to obtain the latent features from the biomedical literature, and this has been proved to be helpful in improving the DDI extraction performance. For instance, AB-LSTM (*Sahu & Anand, 2018*) uses word and position embedding, and two-stage LSTM (*Huang et al., 2017*) combines word embedding with part of speech tag embedding in the model. SCNN (*Zhao et al., 2016*) proposed a syntax word embedding strategy, in which information about the position and part of speech features was taken into account. However, these embedding strategies typically ignore the information associated with the target drug, which would be conductive for more accurate extraction of DDIs.

In this article, we present a novel pharmacological representation-based long short-term memory network, named Phar-LSTM, for DDI extraction. The main contributions of this article are summarized as follows:

(1) A newly defined embedding strategy is proposed to extract pharmacological representations from the biomedical literature by combining word embedding with target drug related information embedding (*e.g.*, embedding the degree of the correlation between the word and the target drug, the relative position information of the target drug for each word) in our model.
(2) An LSTM-based multi-task learning scheme is introduced to jointly tackle the related tasks of DDI extraction (*i.e.,* determine whether the given document contains a DDI and identify the specific DDI types) and capture the common features that would benefit both tasks.
(3) We explore the DDI extraction performance of the models with 10 different LSTM variants.
(4) Experiments on the DDIExtraction 2011 and DDIExtraction 2013 corpuses were conducted to evaluate the performance of the proposed method, and the results show that our method outperforms other state-of-the-art methods on both datasets.

MATERIALS AND METHODS

The overall process of our Phar-LSTM method is composed of three parts (illustrated in Fig. 1): (1) Extracting the pharmacological representations from the datasets, which consists of different but related tasks according to the newly defined embedding strategy; (2) taking the pharmacological representations as the input, and extracting the common features of the related tasks through the LSTM-based multi-task learning scheme; (3) the shared features are fed to the corresponding classifier for each task, and the classification results are regarded as the final output.

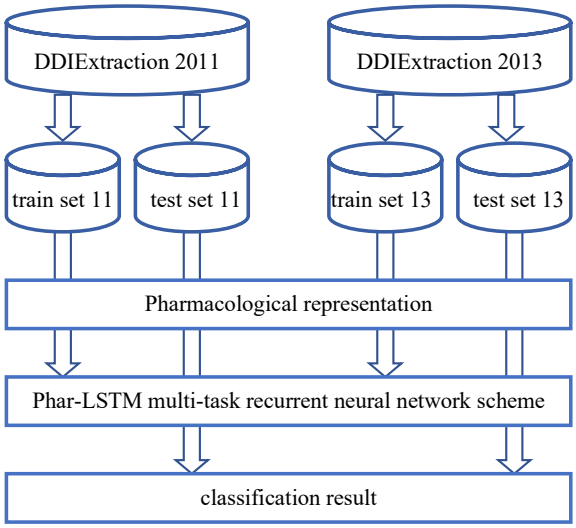

Figure 1 **Overall processing flow of our Phar-LSTM scheme.**
Full-size DOI: 10.7717/peerj.16606/fig-1

Datasets

Text mining and natural language processing have recently benefitted the pharmacological industry. The DDIExtraction 2011 (*Segura-Bedmar, Martínez Fernández & Sánchez Cisneros, 2011*) and DDIExtraction 2013 (*Segura-Bedmar, Martínez Fernández & Herrero Zazo, 2013*) challenge tasks are held to promote the research of DDI extraction by providing benchmark datasets and enabling researchers to compare their methods fairly. The DDIExtraction 2011 challenge focuses on the binary classification of DDIs, that is, deciding whether the given document contains DDIs. For the DDIExtraction 2013 challenge, researchers must identify five DDI types: ADVICE, EFFECT, MECHANISM, INT, and NEGATIVE, which can be considered to be a multi-class classification problem.

The DDIExtraction 2011 dataset includes 579 documents about 14,949 drugs from DrugBank. These DrugBank documents contain rich chemical and pharmaceutical information. There are 5,806 sentences containing 3,160 DDIs (binary) in the DDIExtraction 2011 dataset. The DDIExtraction 2013 dataset has 784 documents from DrugBank and 233 abstract documents from MedLine, with a total of 5,021 DDIs (five specific DDI types). We selected task 9.2 as the testing dataset. More details of the datasets, including the training and testing information, can be found in *Segura-Bedmar, Martínez Fernández & Sánchez Cisneros (2011)* and *Segura-Bedmar, Martínez Fernández & Herrero Zazo (2013)*.

Embedding based pharmacological representation

To obtain useful pharmacological information from the biomedical literature, we present a newly defined embedding strategy to convert the raw input (biomedical documents) into high-dimensional vectors, that is, the pharmacological representations. It should be noted that a major difference between DDI extraction and other natural language processing tasks is that the two target drug entities in the DDI instance should be fully considered since

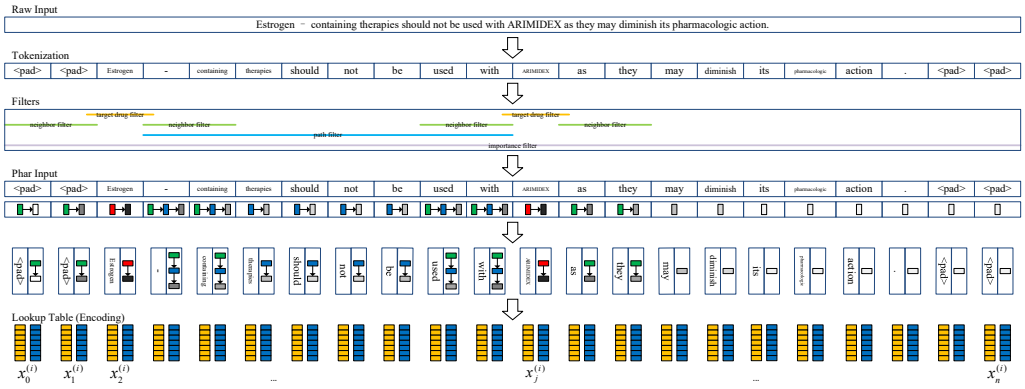

Figure 2 **Pharmacological representation framework.**

Full-size DOI: 10.7717/peerj.16606/fig-2

the target drug pair contains important pharmacological information. For a document containing n drugs, there are C_n^2 DDI candidates. A document may contain more than one DDI instance, and all DDI candidates in the same document are expected to differ from each other. A common way to represent a DDI is "drug blinding", that is, replacing the two target drugs with "drug1" and "drug2", and the other drugs in the document are represented as "durg0" (*Liu et al., 2016*). However, this drug blinding strategy may discard some valuable pharmacological information contained in the target drugs (*e.g.*, the distinguishing information between different target drugs).

The pharmacological representation framework is shown in Fig. 2. First, we tokenized the documents into token sequences. It should be noted that different from the drug blinding strategy, the target drugs are not replaced with the words "drug1" and "drug2". Therefore, more pharmacological information is extracted. The t th token unit was mapped to the $x_t^{(\text{token})}$ using the word embedding strategy. We transformed each token into a d dimensional vector through random encoding as inspired by *Wieting & Kiela (2019)*, wherein the values for each dimension are in $\left[-1/\sqrt{d}, 1/\sqrt{d}\right]$. In practice, we set $d = 400$ in our experiments.

Second, to extract the target drug related information, a set of filters are introduced to obtain the corresponding information in four aspects from the token sentence. The target drug filter determines whether a token is a target drug or not (1: True; 0: False), and the neighbor filter determines whether a token is a neighbor of the target drug (1: True; 0: False). Whether a token exists between a pair of target drugs (1: True; 0: False) is determined by the path filter. The importance filter measures the degree of the token associated with the nearest target drug. The closer the distance, the higher the degree (*i.e.*, more importance is attached).

We define the metric $I = 1/(r+1)^2$, where r is the distance from the token to the nearest target drug (*e.g.*, if the token is the target drug, $r = 0$; if the token is located near to the target drug, $r = 1$). Similar to the vector space model (*Salton, Wong & Yang, 1975*), we characterize the target drug related information of the t th token as a four-dimensional vector $x_t^{(\text{Phar})}$.

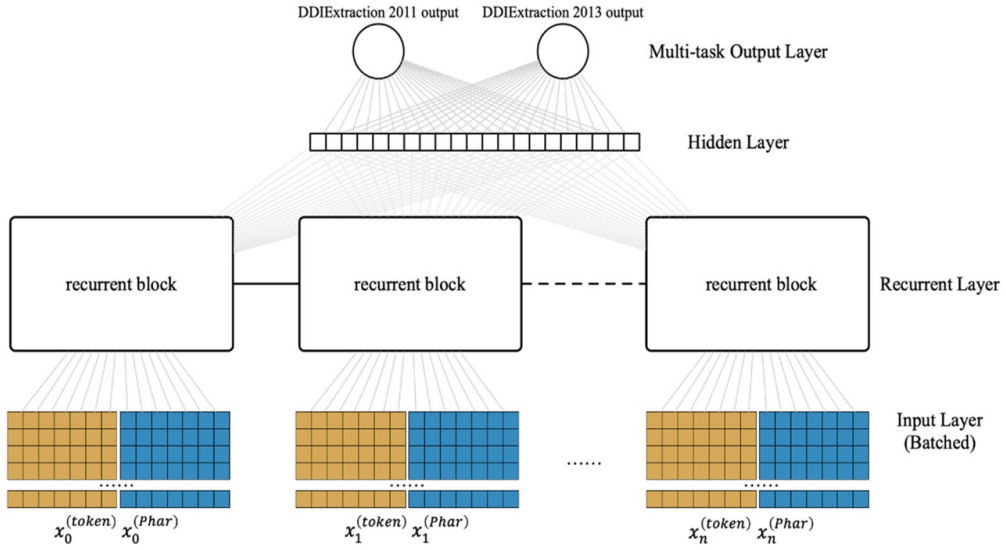

Figure 3 **Architecture of our LSTM-based multi-task learning.**
Full-size DOI: 10.7717/peerj.16606/fig-3

Finally, a given document $D = (w_0, w_1, \ldots, w_m)$ with m words is represented by $D^{(\text{input})} = ((x_0^{(\text{token})}, x_0^{(\text{Phar})}), (x_1^{(\text{token})}, x_1^{(\text{Phar})}), \ldots, (x_n^{(\text{token})}, x_n^{(\text{Phar})}))$, which we call pharmacological representation. Usually, n is larger than m because a document may be tokenized by splitting using punctuation, such as "-" and ".".

LSTM-based multi-task learning

Most previous studies tackled two tasks (*i.e.,* the coarse-grained task and fine-grained task) separately. Because multi-task learning may learn the common features of the related tasks that would benefit each task (*Caruana, 1997*), here, we present an LSTM-based multi-task learning scheme as shown in Fig. 3 (flowing from the bottom up). The input layer converts the raw input into the pharmacological representations. For the recurrent layer, we adopt a special Recurrent Neural Network (RNN) structure (LSTM) (*Van Houdt, Mosquera & Nápoles, 2020*), which can store the previous information for a long time in data processing. Note that each recurrent block of the recurrent layer can be assigned a different LSTM variant, which is illustrated in Fig. 4. The multi-task output layer feeds the common features extracted from the hidden layer into the corresponding SoftMax classifiers. We pretrain the parameters of the neural network(except the multi-task output layer) and then fine-tune in the classification stage.

The Phar-LSTM block contains four gates and a cell state, whose variation that we use here (Fig. 4A) is formulated as

$$z_t = \left(H_{t-1}, x_t^{(\text{token})}\right), \tag{1}$$

$$P_t = \left(g\left(W^P x^{(\text{Phar})} + b^P\right), z_t\right), \tag{2}$$

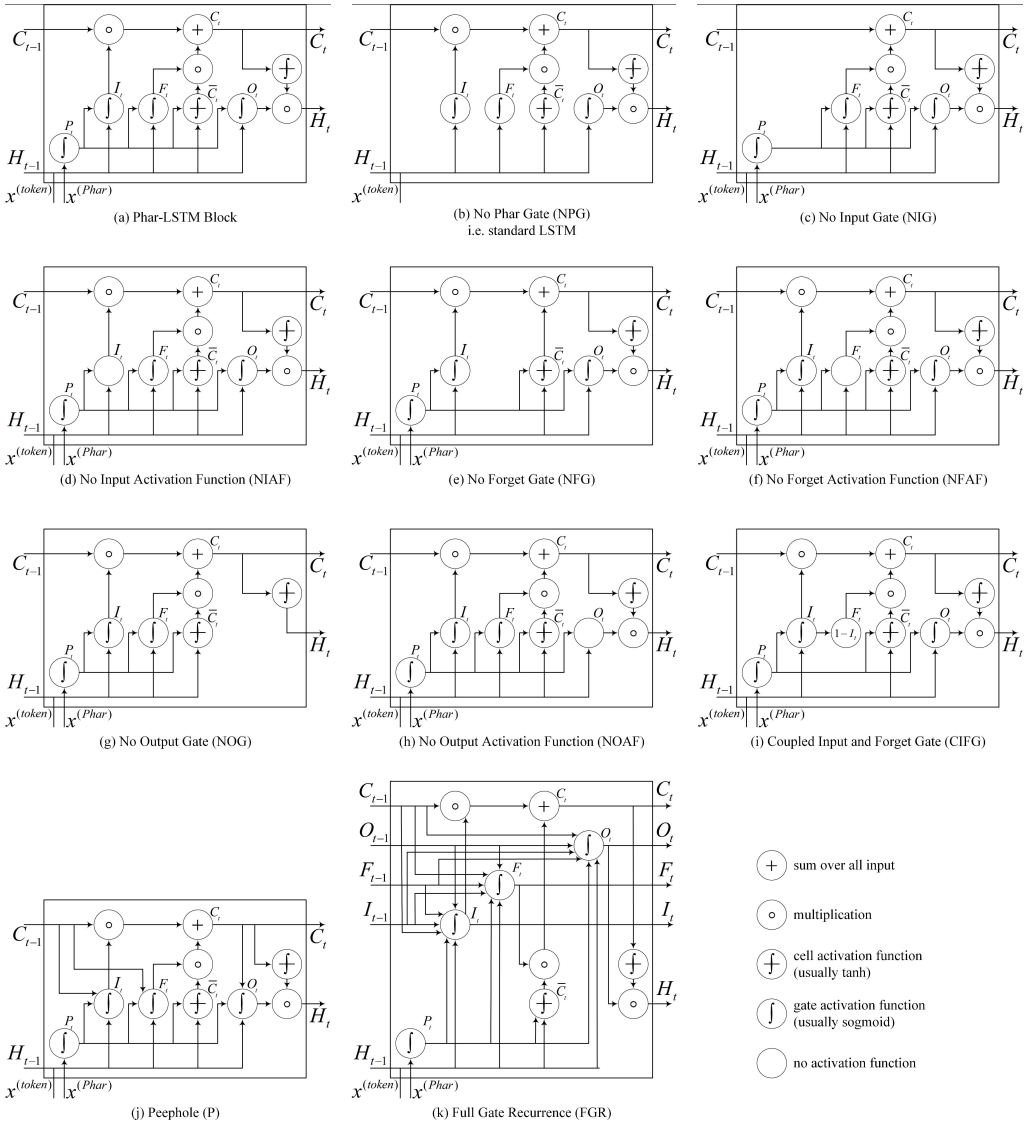

Figure 4 **(A–K) Details of the 10 different LSTM variants.**

Full-size DOI: 10.7717/peerj.16606/fig-4

$$C_t = g\left(W^I P_t + b^I\right) \circ C_{t-1} + g\left(W^F P_t + b^F\right) \circ h\left(W^C p_t + b^C\right), \tag{3}$$

$$O_t = g\left(W^O P_t + b^O\right), \tag{4}$$

$$H_t = O_t \circ h(C_t). \tag{5}$$

In the above, $x_t^{(\text{token})}$ and $x_t^{(\text{Phar})}$ represent the corresponding vectors (*i.e.*, the pharmacological representation) generated by the t th token using the embedding strategy

(see *Embedding based pharmacological representation*). W^P, W^C, W^I, W^F, and W^O are the weight matrices for the pharmacological gate, the cell state, the input gate, the forget gate, and the output gate separately, and b^P, b^C, b^I, b^F, and b^O, respectively, are the corresponding bias units. The functions g and h are activation functions. The sigmoid function is usually used as g for the four gates, and the tangent function is typically used as h for the cell state. The $\circ$ denotes point-wise multiplication.

Other LSTM variants

Different structures of LSTM may influence the results; therefore, we studied the DDI extraction performance of the models with different LSTM variants. The LSTM variants can be derived by modifying the gates, activation function, and connections. The derived 10 variants are shown in Fig. 4, and the details are as follows:

(1) No pharmacological gate (NPG)

Graves & Schmidhuber (2005) originally proposed the LSTM, which is also known as vanilla LSTM. Phar-LSTM can be transformed to the vanilla LSTM by removing the pharmacological gate:

$$C_t = g\left(W^I z_t + b^I\right) \circ C_{t-1} + g\left(W^F z_t + b^F\right) \circ h\left(W^C z_t + b^C\right). \tag{6}$$

(2) No input gate (NIG)

By removing the input gate, we obtain a lighter version of C_t.C_t conveys less information to the next node:

$$C_t = C_{t-1} + g\left(W^F P_t + b^F\right) \circ h\left(W^C P_t + b^C\right). \tag{7}$$

(3) No Input Activation Function (NIAF)

By removing the activation function of I_t, we obtain a "wilder" version of I_t since I_t is no longer confined to $[-1, 1]$ by the sigmoid function:

$$C_t = \left(W^I P_t + b^I\right) \circ C_{t-1} + g\left(W^F P_t + b^F\right) \circ h\left(W^C P_t + b^C\right). \tag{8}$$

(4) No forget gate (NFG)

Gers, Schmidhuber & Cummins (2000) first proposed a variant of LSTM by adding a forget gate, which enabled the LSTM to better forget the history information. By removing the forget gate, we obtain a lighter version of C_t. The C_t of NFG can remember more information because the function of the forget gate is to restrain the useless information from persisting in the history.

$$C_t = g\left(W^I P_t + b^I\right) \circ C_{t-1} + h\left(W^C P_t + b^C\right). \tag{9}$$

(5) No forget activation function (NFAF)

Similar to the NIAF, we obtain a "wilder" version of F_t by removing the activation function of F_t:

$$C_t = g\left(W^I P_t + b^I\right) \circ C_{t-1} + \left(W^F P_t + b^F\right) \circ h\left(W^C P_t + b^C\right). \tag{10}$$

(6) No output gate (NOG)

Similar to the NIG, by removing the output gate, we obtain a lighter version of H_t, and C_t conveys less information to the next node:

$$H_t = h(C_t). \tag{11}$$

(7) No output activation function (NOAF)

Similar to the NIAF, by removing the activation function of O_t, we obtain a "wilder" version of O_t because O_t is no longer confined to $[-1, 1]$ by the sigmoid function:

$$O_t = W^O P_t + b^O. \tag{12}$$

(8) Coupled input and forget gate (CIFG)

Instead of separately calculating what should be forgotten and what should be inputted as new information, the CIFG combines the two steps. The CIFG forgets only when inputting something in its place and inputs new values to the state only when forgetting something older:

$$C_t = g\left(W^I P_t + b^I\right) \circ C_{t-1} + \left(1 - g\left(W^I P_t + b^I\right)\right) \circ h\left(W^C P_t + b^C\right). \tag{13}$$

(9) Peephole (P)

Gers & Schmidhuber (2000) argued that the cell state should control the gates in order to learn precise timings. Therefore, we add connections from the cell to the gates in Phar-LSTM, which are named as Peephole, to make precise timings easier to learn:

$$C_t = g\left(W^I (P_t, C_{t-1}) + b^I\right) \circ C_{t-1} + g\left(W^F (P_t, C_{t-1}) + b^F\right) \circ h\left(W^C P_t + b^C\right), \tag{14}$$

$$O_t = g\left(W^O (P_t, C_t) + b^O\right). \tag{15}$$

(10) Full gate recurrence (FGR)

The LSTM (*Van Houdt, Mosquera & Nápoles, 2020*) consists of cell state and input and output gates and does not include the forget gate and peephole connections. A hybrid of real-time recurrent learning (*Robinson & Fallside, 1987*) and backpropagation through time (*Werbos, 1988*) is used for training. In this case, only the gradient of the cell state was propagated back, and the gradient for the other recurrent connections was truncated. FGR means that all the gates received recurrent inputs from the previous time step:

$$I_t = g\left(W^I \begin{bmatrix} P_t, \\ I_{t-1}, \\ F_{t-1}, \\ O_{t-1}, \\ C_{t-1} \end{bmatrix} + b^I\right), \tag{16}$$

$$F_t = g\left(W^F \begin{bmatrix} P_t, \\ I_{t-1}, \\ F_{t-1}, \\ O_{t-1}, \\ C_{t-1} \end{bmatrix} + b^F\right), \tag{17}$$

$$O_t = g\left(W^O \begin{bmatrix} P_t, \\ I_{t-1}, \\ F_{t-1}, \\ O_{t-1}, \\ C_{t-1} \end{bmatrix} + b^O\right). \tag{18}$$

Classification and training

We use the SoftMax classifier for classification. Let k denote the number of DDI types. The output $o \in R^{|k|}$ is the probabilities of each class to which S belongs.

$$y = \text{argmax}\left(\frac{1}{\exp(WH_n + b)} \begin{bmatrix} \exp(W_1 H_n + b) \\ \exp(W_2 H_n + b) \\ \cdots \\ \exp(W_k H_n + b) \end{bmatrix}\right). \tag{19}$$

We use the cross-entropy (*De Boer et al., 2005*) cost function and ridge regularization (*Hoerl & Kennard, 1970*) as the optimization objective. For the i th instance, $y^{(i)}$ denotes the output. The cross-entropy cost is

$$J = -\left(\sum_{j=1}^{k} 1\{y^{(i)} = j\} \log \frac{\exp(W_j H_n + b)}{\sum_{l=1}^{k} \exp(WH_n + b)}\right) + \frac{\lambda}{2}\|W\|^2, \tag{20}$$

where $1\{\bullet\}$ is the indicator function, such that $1\{\text{a true statement}\} = 1$ and $1\{\text{a false statement}\} = 0$. We optimize the parameters of the objective function J with Rmsprop (*Chowdhury & Lavelli, 2011*), which is a variant of mini-batch stochastic gradient descent. During each training step, the gradient of J is calculated. Then, all the parameters are adjusted according to the gradient. After the end of training, we have a model that is able to predict two drugs' interactions when a sentence about these drugs is given.

RESULTS AND DISCUSSION

To evaluate the performance of our method for DDI extraction, extensive experiments are conducted to compare the Phar-LSTM approach with different variants and other state-of-art methods on the DDIExtraction 2011 and DDIExtraction 2013 datasets. The setup of the experiments is designed to be as simple as possible to make the comparisons fair.

Evaluation metrics

In this section, we describe the evaluation metrics used in our experiments. For the DDIExtraction 2011 and DDIExtraction 2013 corpuses, Precision (P), Recall (R), F-score (F), and Accuracy (Acc) are widely used as the evaluation metrics (*Asada, Miwa & Sasaki, 2023*). Since the DDIExtraction 2013 corpus is a multi-class classification problem, we adopt the micro-average and macro-average strategy to score the overall performance on the five classes.

To obtain the F-score, the contingency table (or confusion matrix) is built first, in which each row of the matrix represents the instances in a predicted class and each column represents the instances in an actual class. The contingency table enables us to obtain the true positive (TP), false positive (FP), false negative (FN), and true negative (TN). Based on that, the precision, recall, F-score, and accuracy can be defined as follows:

$$P = \frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FP}}, \tag{21}$$

$$R = \frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FN}}, \tag{22}$$

$$F = 2 \times \frac{P \times R}{P+R}, \tag{23}$$

$$\mathrm{Acc} = \frac{\mathrm{TP}+\mathrm{TN}}{\mathrm{TP}+\mathrm{TN}+\mathrm{FP}+\mathrm{FN}}. \tag{24}$$

For the DDIExtraction 2013 corpus, there are five P, R, and F-score values for each class since there are five different classes (the five DDI types: ADVICE, EFFECT, MECHANISM, INT, and NEGATIVE). Each DDI type is evaluated separately. Moreover, to measure the overall performance, two commonly used metrics, *i.e.,* micro-averaged F-score (CLA) and macro-averaged F-scores (MAVG), are calculated. The CLA is calculated by constructing a global contingency table and then calculating the precision and recall, and the MAVG is calculated by first calculating the precision and recall for each type and then taking the average of those results.

To evaluate the scalability of our method, we propose a metric to evaluate the performance gap of the models between the two corpuses under the assumption that if a model has good performance in scalability, it would not only have a high average F-score but also have less variance. For instance, the F-scores of model A are 0.65 and 0.65 on the DDIExtraction 2011 and 2013 corpuses, respectively, and those of model B are 0.60 and 0.70. Although the average F-scores of model A and model B are both 0.65, model A would be considered to have better scalability than model B. Based on this, the metric can be defined using the 1-standard deviation of F-scores as

$$1-\sigma = 1-\sqrt{\frac{1}{2}\left(F_{2011}-\frac{F_{2011}+F_{2013}}{2}\right)^2+\frac{1}{2}\left(F_{2013}-\frac{F_{2011}+F_{2013}}{2}\right)^2}. \tag{25}$$

To evaluate the consistency, the training process for 200 epochs of each learning model is shown as a boxplot, and Welch's t-test at a significance level of alpha $= 0.05$ was used to determine whether the mean test set performance of a learning model was significantly different from that of Phar-LSTM.

To evaluate the reproducibility, we first ran the training process for 10 times using different random seeds and obtained the boxplot of the overall training process to show the gap between each run. Based on the boxplot, we further calculated the sum of the variance of the F-scores of each epoch and the sum of the standard deviation of the F-scores of each epoch to measure the differences among the 10 runs.

Table 1 Scalability comparison of different methods on the DDIExtraction 2011 dataset (the bold indicating the best value on the corresponding metric).

Method	Evaluation metrics							
	TP	FP	FN	TN	P	R	F	Acc
WBI (*Thomas et al., 2013*)	543	354	212	5,914	0.6045	0.7192	0.6574	0.9194
LIMSI-FBK (*Björne et al., 2011a*)	532	376	223	5,895	0.5859	0.7046	0.6398	0.9147
FBK-HLT (*Chowdhury et al., 2011*)	529	377	226	5,894	0.5839	0.7007	0.6370	0.9142
Uturku (*Björne, Kaewphan & Salakoski, 2013*)	520	376	235	5,895	0.5804	0.6887	0.6299	0.9130
LIMSI-CNRS (*Segura-Bedmar, Martínez Fernández & Sánchez Cisneros, 2011*)	490	398	265	5,873	0.5518	0.6490	0.5965	0.9056
BNBNLEL (*Segura-Bedmar, Martínez Fernández & Sánchez Cisneros, 2011*)	420	**266**	335	**6,005**	0.6122	0.5563	0.5829	0.9145
Skeleton-LSTM (*Jiang, Gu & Jiang, 2017*)	550	320	205	5951	0.6322	0.7285	0.6769	0.9253
Phar-LSTM	**559**	311	**196**	5,960	**0.6425**	**0.7404**	**0.6880**	**0.9278**

Hyperparameter settings

Based on previous research and experience, the Phar-LSTMs were trained by an RMSprop optimizer with a loss function of cross entropy and a learning rate of 0.001. Dropout layers were added to each of the embedding layers and hidden layers with a ratio of 0.2. For each run, the number of training epochs were 200 and the batch sizes were 32. All the experiments were run on GeForce GTX-1080 and took 9.3 h on average to complete.

Scalability

To evaluate the scalability of our method, experiments were conducted on both the DDIExtraction 2011 and DDIExtraction 2013 datasets. Before 2013, most studies were evaluated on the DDIExtraction 2011 dataset. After 2013, most research has focused on evaluating the methods on the DDIExtraction 2013 dataset. As far as we know, few methods (*Björne, Kaewphan & Salakoski, 2013*; *Jiang, Gu & Jiang, 2017*; *Thomas et al., 2013*) have been evaluated on both datasets.

We first compared the performance of our scheme with the traditional methods as well as the deep learning-based method on the DDIExtraction 2011 dataset. The results are shown in Table 1. The traditional methods are typically based on manually extracted features or kernels. For example, *Björne et al. (2011b)* leveraged many syntactic-based features, including tokens, dependency types, POS tags, text, and stems. Similarly, *Thomas et al. (2013)* combined an all-path-graph kernel, a shallow linguistic kernel, and a k-band shortest path spectrum kernel, which were all derived from syntactic analysis. Other methods such as FBK-HLT (*Chowdhury et al., 2011*) and LIMSI-FBK (*Björne et al., 2011a*) used either features or kernels or both.

These features and kernels are highly dependent on third-party tools such as syntactic parsing, which makes the method sensitive to the quality of the parsing results and the expertise of researchers in designing features or kernels. Therefore, although the heuristic idea of using features and kernels can be helpful to other researchers, the models themselves may not have good scalability.

Table 2 Scalability comparison of Phar-LSTM with other methods on the DDIExtraction 2013 dataset (the best value on each metric is highlighted in bold).

Method	Evaluation metrics						
	NEG	MEC	EFF	ADV	INT	MAVG	CLA
FBK-irst (*Chowdhury & Lavelli, 2013*)	**0.8**	0.679	0.628	0.692	**0.547**	0.648	0.651
NIL_UCM (*Bokharaeian & Díaz, 2013*)	0.588	0.515	0.489	0.613	0.427	0.535	0.517
SCAI (*Bobić, Fluck & Hofmann, 2013*)	0.683	0.441	0.440	0.559	0.021	0.448	0.452
UC3M (*Sánchez Cisneros, 2013*)	0.676	0.480	0.547	0.575	0.500	0.534	0.529
UCOLORADO SOM (*Hailu, Hunter & Cohen, 2013*)	0.504	0.361	0.311	0.381	0.333	0.407	0.334
Uturku (*Björne, Kaewphan & Salakoski, 2013*)	0.696	0.582	0.600	0.630	0.507	0.587	0.594
UWM-TRIADS (*Rastegar-Mojarad, Boyce & Prasad, 2013*)	0.599	0.446	0.449	0.532	0.421	0.472	0.470
WBI (*Thomas et al., 2013*)	0.736	0.602	0.604	0.618	0.516	0.588	0.599
Kim (*Kim et al., 2015*)	0.775	0.693	0.662	0.725	0.483	–	0.670
CNN (*Liu et al., 2016*)	–	–	–	–	–	–	0.698
Attention-CNN (*Asada, Miwa & Sasaki, 2017*)	–	0.695	0.681	0.773	0.455	–	0.691
One-stage SCNN (*Zhao et al., 2016*)	–	–	–	–	–	–	0.670
Two-stage SCNN (*Zhao et al., 2016*)	–	–	–	–	–	–	0.686
SVM+LSTM (*Huang et al., 2017*)	–	**0.738**	**0.720**	0.715	0.549	0.690	–
Skeleton-LSTM (*Jiang, Gu & Jiang, 2017*)	0.795	0.725	0.701	0.788	0.484	0.707	0.714
AB-LSTM (*Sahu & Anand, 2018*)	–	0.681	0.683	0.697	0.542	0.650	–
Joint AB-LSTM (*Sahu & Anand, 2018*)	–	0.723	0.655	**0.803**	0.441	0.655	–
Phar-LSTM	0.795	0.726	0.699	0.789	0.482	**0.708**	**0.716**

It can be observed from Table 1 that the Phar-LSTM scheme achieved the best performance, with the F-score of 0.6880. Another deep learning-based method, Skeleton-LSTM (*Jiang, Gu & Jiang, 2017*) (F-score: 0.6769), also performed significantly better than other traditional methods, which illustrates the superiority of deep learning-based methods for the coarse-grained task of DDI extraction.

Table 2 shows the results of our scheme in comparison with the baselines on the DDIExtraction 2013 dataset. From Table 2, we can observe that Phar-LSTM achieved the best performance in terms of MAVG and CLA (0.708 and 0.716, respectively). Skeleton-LSTM (*Jiang, Gu & Jiang, 2017*) had similar performance as Phar-LSTM and performed significantly better than other baselines. One posssible reason may be that Skeleton-LSTM and Phar-LSTM both use the end-end-learning framework (*i.e.,* feed the raw input into the neural network and produce the output directly), which would capture some latent features because the features are automatically extracted by the neural network rather than by third-party tools.

To further evaluate the scalability of our scheme, three baselines were chosen for comparing with our defined metric, and the results are shown in Table 3. Note that many methods developed for addressing the coarse-grained DDI extraction task may not be applicable for the fine-grained task of DDI extraction. It can be observed in Table 3 that the Phar-LSTM scheme achieved the highest $(1-\sigma)$ value (0.986) among all methods, which demonstrates the scalability of our method.

Table 3 Scalability comparison of Phar-LSTM with other methods on the two datasets (the bold denoting the best value on the corresponding metric).

Method	Evaluation metrics		
	F_{2011}	F_{2013}	$1-\sigma$
WBI (*Thomas et al., 2013*)	0.6574	0.599	0.9708
Uturku (*Björne, Kaewphan & Salakoski, 2013*)	0.6299	0.594	0.9821
Skeleton-LSTM (*Jiang, Gu & Jiang, 2017*)	0.6769	0.714	0.9810
Phar-LSTM	**0.6880**	**0.716**	**0.9860**

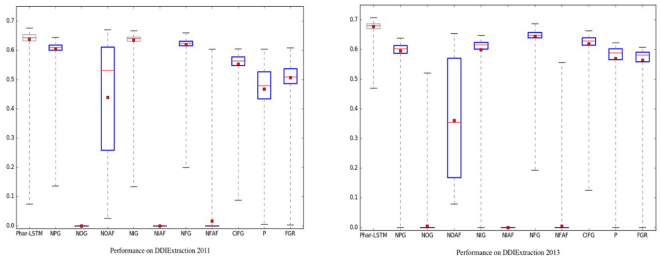

Figure 5 Consistency comparison of Phar-LSTM with 10 variants at a significance level of alpha = 0.05 (Welch's t-test) on the DDIExtraction 2011 dataset and DDIExtraction 2013 dataset.
Full-size DOI: 10.7717/peerj.16606/fig-5

Consistency

To evaluate the consistency of the Phar-LSTM scheme, we compared Phar-LSTM with 10 different variants of LSTM, in which the number of epochs for each variant were set to 200. Welch's t-test at a significance level of alpha = 0.05 was used to determine whether the performance of each variant was significantly different from another. A summary of the results of the different methods with 200 epochs is shown in Fig. 5. The boxplots of the variants that differ significantly from Phar-LSTM are highlighted in blue.

It can be observed from Fig. 5 that Phar-LSTM generally achieved the best performance on both datasets. Moreover, the F-scores of Phar-LSTM for most epochs were relatively stable, which indicates the consistency of our method. Another observation based on Fig. 5 is that removing the output gate (NOG) or the activation functions (NOAF, NIAF, and NFAF) significantly hurt the performance on the two datasets. The ability to output information and the activation of the perceptron appear to be critical for the LSTM architecture. This is probably because the output value of the hidden layers cannot be constrained without the activation function and therefore fails to train the parameters. If the output gate is removed from the LSTM unit, although $x_t^{(\text{token})}$ and $x_t^{(\text{Phar})}$ can be integrated to the hidden layer H_t by C_t, the original information of and $x_t^{(\text{token})}$ and $x_t^{(\text{Phar})}$ is diluted during the calculation of H_t. And then, the final SoftMax regression built on the hidden layer of the last unit captures little information of the input, which leads to the failure of training.

On the contrary, although removing the input gate (NIG) or forget gate (NFG) or coupling them into one gate (CIFG) can decrease the F-score, comparing with the NOG,

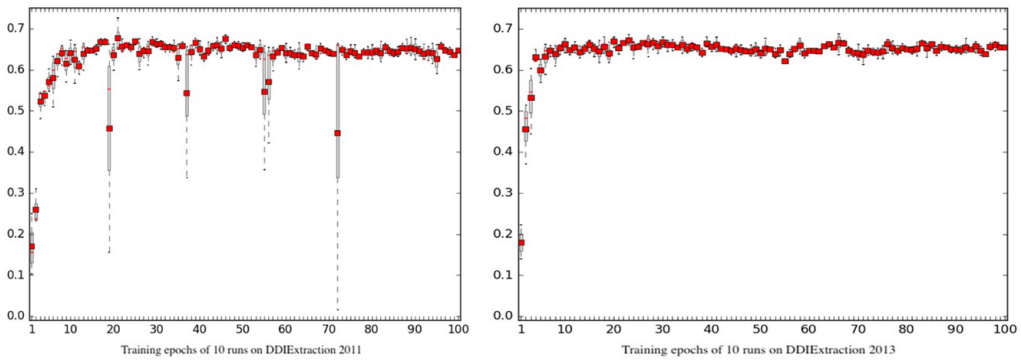

Figure 6 **Reproducibility of Phar-LSTM with different epochs for 10 runs on the DDIExtraction 2011 dataset and DDIExtraction 2013 dataset.**

Full-size DOI: 10.7717/peerj.16606/fig-6

the input information is still integrated through the other gate (*e.g.*, the forget gate for the NIG and the input gate for the NFG). Therefore, the parameters can be trained successfully.

Similarly, removing the pharmacological gate (NPG) generally decreases the F-score more than the NIG, NFG, and CIFG do. This illustrates that the pharmacological gate contains important information to represent the DDI than other gates do. This proves that the Phar-LSTM scheme indeed improved the DDI extraction performance by incorporating the pharmacological gate.

Both adding the Peephole (P) and the full gate recurrence (FGR) decrease the performance while increasing the computational complexity. We generally advise against using them for DDI extraction.

Reproducibility

Due to the random seed mechanism and the implementation of the GPU training architecture, the training process is usually unreproducible. To evaluate the reproducibility of Phar-LSTM, we ran our scheme for many times to check the differences of the outputs. The boxplots of Phar-LSTM's performance for 10 runs with different epochs on the two datasets are shown in Fig. 6, from which we can see that the performances of Phar-LSTM for most epochs are close.

Some specific epochs can be observed for DDIExtraction 2011, such as the 19th, 37th, 55th, 56th, and 72th epochs. For these specific epochs, the performance of different runs differed from each other. However, there were no such specific epochs for DDIExtraction 2013. One possible reason may be the data distribution. DDIExtraction 2011 is smaller, and the annotation strategy is different from that of DDIExtraction 2013. Researchers should be aware of these specific epochs. The best way to check the model is using a validation set. By conducting experiments on the validation set and drawing the boxplot of the learning curve of different runs, researchers can easily find these specific epochs and improve their extraction system.

Another finding is that both the climb stages ($0 < \text{epoch} < 10$) of DDIExtraction 2011 and DDIExtraction 2013 blurred. The reason is that the initial random states of the

parameters are different, which may cause the performances to differ during the climb stage. However, after the climb stage, the performances with the following epochs are much closer. This means that the Phar-LSTM scheme can adapt to different initial random seeds.

To further compare the reproducibility of Phar-LSTM with other variants, we summed the variance and the standard deviation of each epoch for the 10 runs. The metrics (variance and standard deviation) can indicate the overall reproducibility. From Table 4, we can see that Phar-LSTM, NOG, NIAF, and NFAF had better scores than the other models, and the conclusion is consistent with Fig. 5. However, from Fig. 5, we can see that the NOG, NIAF and NFAF had poor *F*-scores. Although the results of the three models can be reproduced easily, the value of the three models is low. Phar-LSTM reaches a good balance between high reproducibility and high *F*-score.

CONCLUSIONS AND FUTURE WORK

In this study, we proposed a pharmacological representation-based LSTM network to extract DDIs from the biomedical literature. Different from previous studies, we adopted a new embedding strategy, in which the documents were represented as a sequence of word embeddings and target drug relative information embeddings, called pharmacological representations. An LSTM-based multi-task learning scheme was introduced to extract features of the pharmacological representations from two related DDI extraction tasks (*i.e.,* the coarse-grained task and fine-grained task). Experimental results showed that our scheme outperformed other state-of-the-art methods on both DDIExtraction 2011 (the coarse-grained task) and DDIExtraction 2013 (the fine-grained task). The scalability, consistency, and reproducibility of our scheme were evaluated on both datasets, and the results demonstrated the relatively superior performance of our method in these aspects.

In our forthcoming work, we will address the existing issues to enhance the prediction of DDI events. First, we will extend our method to other biomedical relative extraction tasks, such as protein–protein interaction extraction and chemical–disease interaction extraction. Second, there are insufficient interactions for certain events, and we will explore data augmentation techniques to expand the event dataset.

ACKNOWLEDGEMENTS

We thank LetPub for its linguistic assistance during the preparation of this manuscript.

Table 4 Reproducibility comparison of Phar-LSTM with other methods on DDIExtraction 2013.

Metric	Different approaches									
	Phar-LSTM	**NPG**	**NOG**	**NOAF**	**NIG**	**NIAF**	**NFG**	**NFAF**	**CIFG**	***P***
Variance[2011]	0.2219	0.5625	0.1283	12.7918	0.4409	0.1380	0.3634	0.1752	0.3967	0.6529
Standard deviation[2011]	2.8032	5.6749	1.6493	150.108	4.1166	1.2839	3.4471	2.1577	4.5873	6.3908
Variance[2013]	0.0222	0.03979	0.0146	4.1293	0.0631	0.0164	0.0364	0.0193	0.0413	0.1695
Standard deviation[2013]	1.3934	2.7890	0.7589	48.3608	3.8259	0.7765	1.8223	0.8131	2.4978	3.6656

ADDITIONAL INFORMATION AND DECLARATIONS

Funding

This work was supported by the Shenzhen Basic Research Foundation (No. 20220819134631001), the Characteristic Innovation Projects of Colleges and Universities in Guangdong Province (No. 2023KTSCX326), the Shenzhen Institute of Information Technology (No. SZIIT2022KJ018), and the China Postdoctoral Science Foundation (Nos. 2018M633187 and 2020M672892). The funders had no role in study design, data collection and analysis, decision to publish, or preparation of the manuscript.

Grant Disclosures

The following grant information was disclosed by the authors:
Shenzhen Basic Research Foundation: 20220819134631001.
Characteristic Innovation Projects of Colleges and Universities in Guangdong Province: 2023KTSCX326.
Shenzhen Institute of Information Technology: SZIIT2022KJ018.
China Postdoctoral Science Foundation: 2018M633187, 2020M672892.

Competing Interests

The authors declare there are no competing interests.

Author Contributions

- Mingqing Huang conceived and designed the experiments, performed the experiments, analyzed the data, prepared figures and/or tables, authored or reviewed drafts of the article, and approved the final draft.
- Zhenchao Jiang conceived and designed the experiments, performed the experiments, analyzed the data, prepared figures and/or tables, authored or reviewed drafts of the article, and approved the final draft.
- Shun Guo conceived and designed the experiments, analyzed the data, prepared figures and/or tables, authored or reviewed drafts of the article, and approved the final draft.

Data Availability

The following information was supplied regarding data availability:

The datasets and source codes are available at Zenodo:

Huang Mingqing, Jiang Zhenchao, & Guo Shun. (2023). Data and Codes of "A Pharmacological Representation-based LSTM Network for Drug–Drug Interaction Extraction" [Data set]. Zenodo. https://doi.org/10.5281/zenodo.8249384.

Supplemental Information

Supplemental information for this article can be found online at http://dx.doi.org/10.7717/peerj.16606#supplemental-information.

REFERENCES

Asada M, Miwa M, Sasaki Y. 2017. Extracting drug–drug interactions with attention CNNs. In: *BioNLP 2017*. 9–18.

Asada M, Miwa M, Sasaki Y. 2023. Integrating heterogeneous knowledge graphs into drug–drug interaction extraction from the literature. *Bioinformatics* **39(1)**:btac754 DOI 10.1093/bioinformatics/btac754.

Björne J, Airola A, Pahikkala T, Salakoski T. 2011a. Drug–drug interaction extraction from biomedical texts with SVM and RLS classifiers. In: *Proceedings of the first Challenge task on Drug–Drug Interaction Extraction (DDIExtraction-2011)*. 35–42.

Björne J, Heimonen J, Ginter F, Airola A, Pahikkala T, Salakoski T. 2011b. Extracting contextualized complex biological events with rich graph-based feature sets. *Computational Intelligence* **27(4)**:541–557 DOI 10.1111/j.1467-8640.2011.00399.x.

Björne J, Kaewphan S, Salakoski T. 2013. Uturku: drug named entity recognition and drug–drug interaction extraction using SVM classification and domain knowledge. In: *Proceedings of the 7th International Workshop on Semantic Evaluation (SemEval 2013)*. 651–659.

Bobić T, Fluck J, Hofmann M. 2013. SCAI: extracting drug–drug interactions using a rich feature vector. In: *Second joint conference on lexical and computational semantics (*SEM), Volume 2: proceedings of the seventh international workshop on semantic evaluation (SemEval 2013)*. Stroudsburg: ACL, 675–683.

Bokharaeian B, Díaz A. 2013. NIL_UCM: extracting drug–drug interactions from text through combination of sequence and tree kernels. In: *Second joint conference on lexical and computational semantics (*SEM), Volume 2: Proceedings of the seventh international workshop on semantic evaluation (SemEval 2013)*. Stroudsburg: ACL, 644–650.

Brown JD, Winterstein AG. 2019. Potential adverse drug events and drug–drug interactions with medical and consumer cannabidiol (cbd) use. *Journal of Clinical Medicine* **8(7)**:989 DOI 10.3390/jcm8070989.

Cao W, Yang Q, Zhang W, Xu Y, Wang S, Wu Y, Zhao Y, Guo Z, Li R, Gao R. 2021. Drug–drug interactions between salvianolate injection and aspirin based on their metabolic enzymes. *Biomedicine & Pharmacotherapy* **135**:111203 DOI 10.1016/j.biopha.2020.111203.

Caruana R. 1997. Multitask learning. *Machine Learning* **28(1)**:41–75 DOI 10.1023/A:1007379606734.

Chen X, Guan NN, Sun YZ, Li JQ, Qu J. 2020. MicroRNA-small molecule association identification: from experimental results to computational models. *Briefings in bioinformatics* **21(1)**:47–61.

Chen X, Ren B, Chen M, Wang Q, Zhang L, Yan G. 2016a. NLLSS: predicting synergistic drug combinations based on semi-supervised learning. *PLOS Computational Biology* **12(7)**:e1004975 DOI 10.1371/journal.pcbi.1004975.

Chen X, Yan CC, Zhang X, Zhang X, Dai F, Yin J, Zhang Y. 2016b. Drug–target interaction prediction: databases, web servers and computational models. *Briefings in Bioinformatics* **17(4)**:696–712 DOI 10.1093/bib/bbv066.

Chowdhury MFM, Abacha AB, Lavelli A, Zweigenbaum P. 2011. Two different machine learning techniques for drug–drug interaction extraction. *Challenge Task on Drug-Drug Interaction Extraction* **761**:19–26.

Chowdhury MFM, Lavelli A. 2011. Drug–drug interaction extraction using composite kernels. In: *Challenge Task on Drug-Drug Interaction Extraction pages*. 27–33.

Chowdhury MFM, Lavelli A. 2013. FBK-irst: a multi-phase kernel based approach for drug–drug interaction detection and classification that exploits linguistic information. In: *Proceedings of the 7th International Workshop on Semantic Evaluation (SemEval 2013)*. 351–355.

De Boer PT, Kroese DP, Mannor S, Rubinstein RY. 2005. A tutorial on the cross-entropy method. *Annals of Operations Research* **134(1)**:19–67 DOI 10.1007/s10479-005-5724-z.

Gers FA, Schmidhuber J. 2000. Recurrent nets that time and count. In: *Proceedings of the IEEE-INNS-ENNS International Joint Conference on Neural Networks (IJCNN)*. Piscataway: IEEE, 189–194.

Gers FA, Schmidhuber J, Cummins F. 2000. Learning to forget: continual prediction with LSTM. *Neural Computation* **12(10)**:2451–2471 DOI 10.1162/089976600300015015.

Graves A, Schmidhuber J. 2005. Framewise phoneme classification with bidirectional LSTM and other neural network architectures. *Neural Networks* **18(5–6)**:602–610 DOI 10.1016/j.neunet.2005.06.042.

Hailu N, Hunter L, Cohen KB. 2013. UColorado_SOM: extraction of drug–drug interactions from biomedical text using knowledge-rich and knowledge-poor features. In: *Second joint conference on lexical and computational semantics (*SEM), Volume 2: Proceedings of the seventh international workshop on semantic evaluation (SemEval 2013)*. Stroudsburg: ACL, 684–688.

Hoerl AE, Kennard RW. 1970. Ridge regression: biased estimation for nonorthogonal problems. *Technometrics* **12(1)**:55–67 DOI 10.1080/00401706.1970.10488634.

Huang D, Jiang Z, Zou L, Li L. 2017. Drug–drug interaction extraction from biomedical literature using support vector machine and long short term memory networks. *Information Sciences* **415**:100–109.

Jiang Z, Gu L, Jiang Q. 2017. Drug drug interaction extraction from literature using a skeleton long short term memory neural network. In: *Proceedings of the 2017 IEEE international conference on bioinformatics and biomedicine (BIBM)*. Piscataway: IEEE, 552–555.

Karbownik A, Miedziaszczyk M, Grabowski T, Stanisławiak-Rudowicz J, Jaźwiec R, Wolc A, Grześkowiak E, Szałek E. 2020. *In vivo* assessment of potential for UGT-inhibition-based drug–drug interaction between sorafenib and tapentadol. *Biomedicine & Pharmacotherapy* **130**:110530 DOI 10.1016/j.biopha.2020.110530.

Kim S, Liu H, Yeganova L, Wilbur WJ. 2015. Extracting drug–drug interactions from literature using a rich feature-based linear kernel approach. *Journal of Biomedical Informatics* **55**:23–30 DOI 10.1016/j.jbi.2015.03.002.

Lin S, Wang Y, Zhang L, Chu Y, Liu Y, Fang Y, Jiang M, Wang Q, Zhao B, Xiong Y, Wei D. 2022. MDF-SA-DDI: predicting drug–drug interaction events based on multi-source drug fusion, multi-source feature fusion and transformer self-attention mechanism. *Briefings in Bioinformatics* **23(1)**:bbab421 DOI 10.1093/bib/bbab421.

Liu S, Tang B, Chen Q, Wang X. 2016. Drug–drug interaction extraction *via* convolutional neural networks. *Computational and Mathematical Methods in Medicine* **2016**:6918381 DOI 10.1155/2016/6918381.

Peng L, Tu Y, Huang L, Li Y, Fu X, Chen X. 2022. DAESTB: inferring associations of small molecule–mirna *via* a scalable tree boosting model based on deep autoencoder. *Briefings in Bioinformatics* **23(6)**:bbac478 DOI 10.1093/bib/bbac478.

Qian J, Qiu X, Tan X, Li Q, Chen J, Jiang X. 2022. An Attentive LSTM based approach for adverse drug reactions prediction. *Applied Intelligence* **53(5)**:1–15.

Rastegar-Mojarad M, Boyce RD, Prasad R. 2013. UWM-TRIADS: classifying drug–drug interactions with two-stage SVM and post-processing. In: *Proceedings of the 7th International Workshop on Semantic Evaluation (SemEval 2013)*. 667–674.

Robinson AJ, Fallside F. 1987. *The utility driven dynamic error propagation network*. Cambridge: Cambridge University Press.

Sahu SK, Anand A. 2018. Drug–drug interaction extraction from biomedical texts using long short-term memory network. *Journal of Biomedical Informatics* **86**:15–24 DOI 10.1016/j.jbi.2018.08.005.

Salton G, Wong A, Yang CS. 1975. A vector space model for automatic indexing. *Communications of the ACM* **18(11)**:613–620 DOI 10.1145/361219.361220.

Sánchez Cisneros D. 2013. UC3M: a kernel-based approach to identify and classify DDIs in biomedical texts. In: *Proceedings of the 7th International Workshop on Semantic Evaluation (SemEval 2013)*. 617–621.

Segura-Bedmar I, Martínez Fernández P, Herrero Zazo M. 2013. Semeval-2013 task 9: extraction of drug–drug interactions from biomedical texts (ddiextraction 2013). In: *Proceedings of the Second Joint Conference on Lexical and Computational Semantics*. 341–350.

Segura-Bedmar I, Martínez Fernández P, Sánchez Cisneros D. 2011. The 1st DDIExtraction-2011 challenge task: extraction of drug–drug interactions from biomedical texts. In: *Proceedings of the 1st Challenge Task on Drug–drug Interaction Extraction*. 1–9.

Thomas P, Neves M, Rocktäschel T, Leser U. 2013. WBI-DDI: drug–drug interaction extraction using majority voting. In: *Proceedings of the 7th International Workshop on Semantic Evaluation (SemEval 2013)*. 628–635.

Van Houdt G, Mosquera C, Nápoles G. 2020. A review on the long short-term memory model. *Artificial Intelligence Review* **53(8)**:5929–5955 DOI 10.1007/s10462-020-09838-1.

Werbos PJ. 1988. Generalization of backpropagation with application to a recurrent gas market model. *Neural Networks* **1**(**4**):339–356.

Wieting J, Kiela D. 2019. No training required: exploring random encoders for sentence classification. ArXiv arXiv:1901.10444.

Zhang T, Leng J, Liu Y. 2020. Deep learning for drug–drug interaction extraction from the literature: a review. *Briefings in Bioinformatics* **21**(**5**):1609–1627 DOI 10.1093/bib/bbz087.

Zhao Z, Yang Z, Luo L, Lin H, Wang J. 2016. Drug drug interaction extraction from biomedical literature using syntax convolutional neural network. *Bioinformatics* **32**(**22**):3444–3453 DOI 10.1093/bioinformatics/btw486.