Procesamiento del Lenguaje Natural, Revista nº 64, marzo de 2020, pp. 69-76 recibido 30-11-2019 revisado 14-01-2020 aceptado 16-01-2020

One stage versus two stages deep learning approaches for the extraction of drug-drug interactions from texts

Comparando enfoques deep learning en una fase y en dos fases para extraer interacciones farmacológicas de texto

Antonio Miranda-Escalada[1], **Isabel Segura-Bedmar**[2]
[1]Barcelona Supercomputing Center, Barcelona, Spain
[2]Universidad Carlos III de Madrid, Leganés, Spain
antonio.miranda@bsc.es, isegura@inf.uc3m.es

Abstract: Drug-drug interactions (DDI) are a cause of adverse drug reactions. They occur when a drug has an impact on the effect of another drug. There is not a complete, up to date database where health care professionals can consult the interactions of any drug because most of the knowledge on DDI is hidden in unstructured text. In last years, deep learning has been succesfully applied to the extraction of DDI from texts, which requires the detection and later classification of DDI. Most of the deep learning systems for DDI extraction developed so far have addressed the detection and classification in one single step. In this study, we compare the performance of one-stage and two-stage architectures for DDI extraction. Our architectures are based on a bidirectional recurrent neural network layer composed of Gated Recurrent Units. The two-stage system obtained a 67.45 % micro-average F1 score on the test set.
Keywords: Relation Extraction, Drug-drug interaction, Recurrent Neural Network, Gated Recurrent Unit

Resumen: Las interacciones farmacológicas (DDI) son una de las causas de reacciones adversas a medicamentos. Ocurren cuando una medicina interfiere en la acción de una segunda. En la actualidad, no existe una base de datos completa y actualizada donde los profesionales de la salud puedan consultar las interacciones de cualquier medicamento porque la mayor parte del conocimiento sobre DDIs está oculto en texto no estructurado. En los últimos años, el aprendizaje profundo se ha aplicado con éxito a la extracción de DDIs de los textos, lo que requiere la detección y posterior clasificación de DDIs. La mayoría de los sistemas de aprendizaje profundo para extracción de DDIs desarrollados hasta ahora han abordado la detección y clasificación en un solo paso. En este estudio, comparamos el rendimiento de las arquitecturas de una y dos etapas para la extracción de DDI. Nuestras arquitecturas se basan en una capa de red neuronal recurrente bidireccional compuesta de Gated Recurrent Units (GRU). El sistema en dos etapas obtuvo un puntaje F1 promedio de 67.45 % en el dataset de evaluación.
Palabras clave: Extracción de relaciones, interacciones farmacológicas, Redes neuronales recurrentes, Gated Recurrent Unit

1 Introduction

One of the causes of adverse drug reactions (ADR) is the wrong combination of different drugs. That is, when one drug influences the effect of another. This is known as a drug-drug interaction (DDI). DDIs may have a positive effect on human health. Nevertheless, many DDIs may trigger adverse drug reactions, which can cause health problems and increase healthcare costs.

Despite the efforts in reporting ADRs, such as the monitoring system maintained by the World Health Organization, there is not a single up-to-date database where clinicians can look for all the known DDIs of a drug. Current databases have varying update frequencies, being some of them up to 3 years (Segura-Bedmar, 2010). In addition, most of the information available about DDIs is unstructured, written in natural language in

ISSN 1135-5948. DOI 10.26342/2020-64-8

scientific articles, books and reports.

In this scenario, a challenge has been identified: an automatic system to extract DDIs from biomedical literature is needed to create complete and up-to-date databases with information about DDIs for healthcare professionals. These databases would contribute to reduce adverse drug reactions.

The DDI corpus (Herrero-Zazo et al., 2013) as well as the shared tasks DDIExtraction 2011 (Segura Bedmar, Martinez, and Sánchez Cisneros, 2011) and 2013 (Segura-Bedmar, Martínez, and Herrero-Zazo, 2013) have undoubtedly contributed to the progress of NLP research for the extraction of DDI from texts. Since then, the popularity of this task has rapidly increased over the past few years and a considerable number of papers devoted to the topic is published every year. Early systems (Segura-Bedmar, Martinez, and de Pablo-Sánchez, 2011; Thomas et al., 2013; Chowdhury and Lavelli, 2013; Abacha et al., 2015; Kim et al., 2015) were based on linguistic features combined with classical machine learning algorithms such as SVM (with F-measures around 67%)(Kim et al., 2015). Recently, deep learning methods have triggered a revolution in NLP, demonstrating tremendous success in numerous NLP tasks. The task of DDI extractions has not been oblivious to this revolution. Various deep learning architectures based on Convolutional Neural Network (CNN) and Recurrent Neural Network (RNN) have been explored to this task in the last five years, achieving state-of-the-art performance (Dewi, Dong, and Hu, 2017; Sun et al., 2018).

In most of these architectures, the detection and classification of DDIs are carried out in one only stage. The recent deep learning architectures have achieved state-of-art results (around 80% of micro-F1), but with long training times because of their deep architectures. Our hypotehsis is that performing first the detection and then the classification could give better results and reduce the training time. The detection step could significantly decrease the number of instances to classify, and thereby, reducing the training time of the classification task. Our goals of this work are two-fold: (1) to study if a two-stage arquictecture could give better results than an one-stage one, and (2) to test if the use of Gated Recurrent Units (GRUs)(Cho et al., 2014), instead of using Long Short-Term Memory (LSTM) units, can improve the results of DDI extraction. GRUs show better performance on smaller datasets (Chung et al., 2014). Moreover, we also study the effect of different pre-trained word embeddings models (Pyysalo et al., 2013) on the results.

2 State of the Art

This section provides a review of the most recent studies based on deep learning about DDI extraction. Deep CNN models (Dewi, Dong, and Hu, 2017; Sun et al., 2018) have shown the state-of-the-art performance (85% in F1-score).

In addition to CNN, several systems have exploited Recurrent Neural Network (RNN) for the task of DDI extraction. While CNN has proved to be successful in discovering discriminative and meaningful phrases in a text, RNN models are able capable of capturing contextual information and long-term dependencies (Lai et al., 2015), which is very important in our case since the clues about a DDI could appear anywhere in a sentence. Most RNN systems for DDI extraction have used LSTM units, to our knowledge. The standard architecture includes, after preprocessing, an embedding layer, a bidirectional LSTM, a pooling layer and a softmax layer that retrieves a probability for each DDI type. Some systems incorporate a dense layer before the softmax layer, as it also happened in some CNN-based systems.

Zheng et al. (2017) used a RNN and obtained 77.3% F1-score. Input attention mechanism was used only on the word embedding vectors. Also, a part of speech (POS) tag embedding and position embeddings were concatenated to the word. Unlike the most systems for DDI extraction based on deep learning methods, there was not pooling layer between the bidirectional LSTM and the softmax layer.

Despite most works employ LSTM units, the work of Yi et al. (2017) proposed a system utilizing GRU. The system concatenated a word embedding with two position embeddings. Then, embedding vectors were fed into a bidirectional GRU layer. The output of the bi-GRU layer was transformed by an attention-pooling layer, and later by another attention layer. Finally, a softmax layer returned the DDI class probabilities. An F1 score of 72.23% on the DDI corpus was re-

ported.

All these systems have in common that the detection and classification of DDIs are performed in one only step. This is the first work that compares one-stage and two-stages deep learning architectures for DDI extraction.

3 Approach

This section details the corpus employed together with the two approaches tested. The first solution, named one-stage architecture, detects and classifies drug-drug interactions employing one single GRU. On the other hand, the second approach, named two-stage architecture, utilizes two steps: one GRU to detect the DDIs and a second GRU to classify them. In addition, the common preprocessing for both solutions is also described.

3.1 DDI Corpus

The DDI corpus (Herrero-Zazo et al., 2013) is the standard corpus employed in DDI extraction from texts and as such is used within this work. It contains sentences mentioning pharmaceutical substances extracted from texts describing DDIs from DrugBank database (Wishart, 2017) and scientific abstracts from Medline. In total, the DDI corpus contains 33,502 DDI instances and it is highly unbalanced: 85% of instances belong to the negative class -no DDI is present- and 15% of instances belong to the positive class -a DDI is described. DDIs are further divided into 4 categories:

- *mechanism*: it is used for pharmacokinetic mechanisms (e.g. *Grepafloxacin may inhibit the metabolism of theobromine*).
- *effect*: it is used for descriptions of effects (e.g. *In uninfected volunteers, 46% developed rash while receiving SUSTIVA and clarithromycin*) or pharmacodynamic mechanisms (e.g. *Chlorthalidone may potentiate the action of other antihypertensive drugs*).
- *advise*: it contains DDIs written as advise (e.g. *Ocupress should be used with caution in patients (...)*).
- *int*: this type is used when no extra information is provided (e.g. *Interaction of clindamycin and gentamicin in vitro*).

The corpus was randomly divided into two datasets for training and test. In our study, we assume that the drug mentions are already provided. Thus, we focus on the tasks of detection and classification of DDIs from texts.

3.2 Models

Preprocessing is common to both systems proposed. First, instances were transformed to lowercase. Then, punctuation signs were removed. After that, drug blinding was performed. That is, the two drugs involved in the interaction are respectively substituted by "DRUGA" and "DRUGB", and other drugs mentioned in the sentence are substituted by "DRUGN".

After these steps, sentences are tokenized (split into separate tokens). As a means of having all sentences of equal length, padding was applied after tokenization. A maximum length of 89 was established, since the longest instance in the training dataset of the DDI corpus contains 89 words.

Finally, position flags are added to every token. They indicate the distance of each token to the two drugs involved in the potential interaction (named "DRUGA" and "DRUGB"). Note that, for the tokens "DRUGA" and "DRUGB", one of the position flags is zero, since the distance to theirselves is zero.

After preprocessing, every instance is represented as a matrix composed by n tokens. This representation is known as the embedding layer, where each token is represented by a vector that concatenates one word embedding and two position embeddings. We explored the use of two different word embedding models. The first one was a pre-trained word2vec 200-dimensional model, which was trained with biomedical texts taken from PubMed and PubMed Central (PMC). The second pre-trained word embedding model was trained using PubMed, PMC and Wikipedia. So, unlike the first model, it contains information about general knowledge texts. Both embeddings were developed by Pyysalo et al. (2013); the corpus to train the first one contained 5.5B tokens and for the second one 7B tokens. Each position embedding codifies the distance from the word to one of the interacting drugs in the DDI instance. Position embeddings were randomly initialized to 32-dimensional vectors from the uniform distribution U(0,0.01). Then, at this point, every instance, S, is a

matrix of vectors, $\overrightarrow{g_i}$ such as:

$$S = [\overrightarrow{g_1}, ..., \overrightarrow{g_N}],$$
$$\overrightarrow{g_i} = \overrightarrow{w_i}||\overrightarrow{e1_i}||\overrightarrow{e2_i} \quad (1)$$

3.3 one-stage approach for DDI extraction

First, we describe the first approach for DDI extraction, where the detection and classification tasks are performed in one single step. Once an instance has been preprocessed and represesented by the embedding layer described above, the one-stage system classifies it as either belonging to the *negative* class or to one of the four positive DDI types of the DDI corpus (*effect*, *mechanism*, *advise* or *int*). It employs four layers to classify the instances: embedding, bidirectional GRU, max pooling and output layer (figure 1).

After the embedding layer, a bidirectional recurrent neural network with GRUs ensues. We decided to use GRU with 512 units per direction. The output vectors from both directions, $\overrightarrow{h_{f,i}}$ and $\overrightarrow{h_{b,i}}$, are concatenated:

$$\overrightarrow{h_{f,i}} = \overrightarrow{GRU}(\overrightarrow{g_i}),$$
$$\overleftarrow{h_{b,i}} = \overleftarrow{GRU}(\overrightarrow{g_i}),$$
$$\vec{h_i} = \overrightarrow{h_{f,i}}||\overrightarrow{h_{b,i}},$$
$$S = [\overrightarrow{h_1}, ..., \overrightarrow{h_N}] \quad (2)$$

Then, the instance tensor, S, is input into a max pooling layer, in which only the timestep of highest magnitude for each feature is kept.

$$\overrightarrow{q_i} = max(h_i^i, ..., h_N^i),$$
$$S = [\overrightarrow{q_1}, ..., \overrightarrow{q_N}] \quad (3)$$

Finally, the instance vector, S, is input into a softmax layer with five output units. It contains 5 neurons employing softmax activation functions. Every neuron returns the probability that the instance belongs to one of the five possible classes, the four DDI types defined in the DDI corpus and the non-DDI type. Therefore, a 5-dimensional output vector represents the confidence the system assigns to each of the five classes. The class with higher confidence is selected as the predicted class.

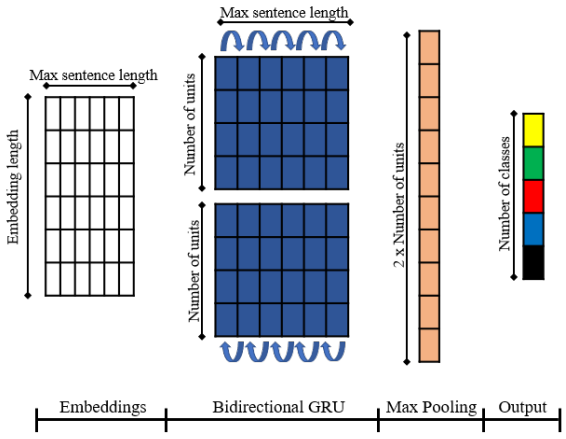

Figure 1: one-stage architecture

3.4 two-stage approach for DDI extraction

Unlike the one-stage system, in the two-stage system, there is a first step to tag each instance as either *negative* or *positive*. Then, instances classified as negative are ruled out. The second step deals with the classification of the positive instances into one of the four DDI corpus categories: *mechanism*, *advise*, *effect* or *int* (figure 2). We describe in more detail each stage below.

The first stage (named detector) also employs four layers to detect the DDI instances: an embedding layer, a bidirectional GRU, a max pooling and a softmax layer with two outputs neurons: one neuron represents the probability for the positive class (the instance is a DDI) and the other neuron stores the probability for the negative class (the instance is not a DDI).

The second stage, named as Classifier, only considers those instances that were classified as positive by the previous step, ruling out the rest of instances. The architecture of this stage is very similar to the previous ones, however, we also introduce, after the max pooling layer, a 64-unit fully connected layer with ReLU activation function, because its integration has shown better performance in combination with complex CNN or RNN architectures (Mohamed, Hinton, and Penn, 2010; Sainath et al., 2015). The output layer contains 4 neurons employing softmax activation functions, one per each DDI type. The higher-probability class is selected as the predicted DDI type.

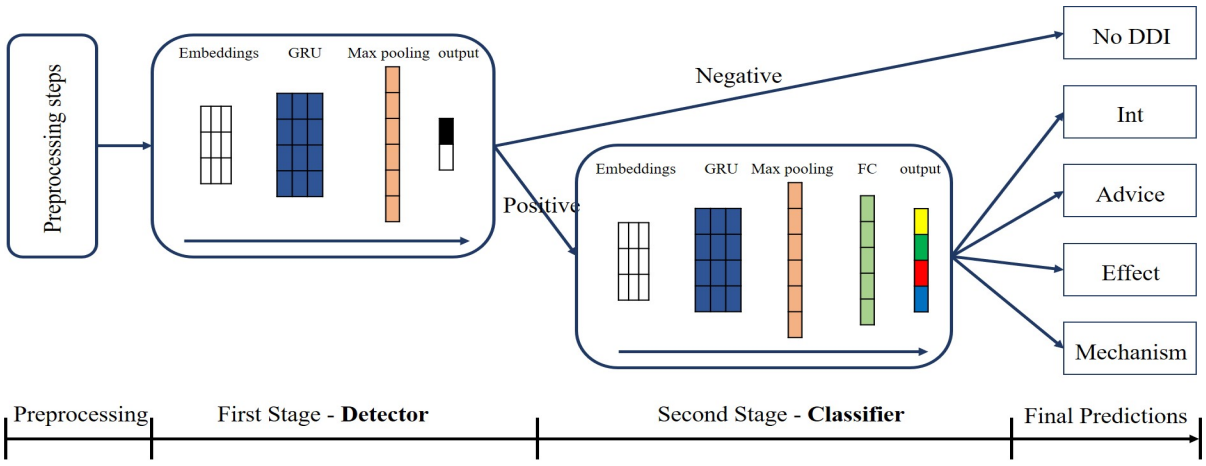

Figure 2: two-stage architecture

3.5 Networks training details

While training the Classifier, class weights were used when computing the loss function for the classes *mechanism*, *advise* and *effect*. This increased the importance on the loss computation of the underrepresented classes and partially solved the class imbalance problem of the dataset. Finally, for training this system, only positive instances were used.

To further deal with class imbalance, negative instance filtering was applied using the rules defined by Kavuluru, Rios, and Tran (2017). In addition, naïve oversampling was employed in the training of the one-stage system and the Detector stage. For the different layers, weights were always initialized according to the Glorot Initializer (Glorot and Bengio, 2010).

In the recurrent layer, we apply always the so-called naïve dropout (Zaremba, Sutskever, and Vinyals, 2015). The dropout probability in the GRU layer and in the fully connected layer was chosen to be always 0.5. The decision was heuristic, since intra-class variability was higher than inter-class variability for the different dropout probabilities tested.

Early Stopping was applied with a patience of 5 epochs. In the Detector stage, F1 score of the positive class was monitored. In the Classifier stage and in the one-stage system, the monitored quantity was micro-average F1 score of the classes *mechanism*, *advise* and *effect*. Class *int* was not taken into account since its number of instances is small (196 in training test, while there are 1687 *effect* instances, for instance).

The optimizer chosen was Adagrad with initial learning rate 0.01. The loss selected was the cross-entropy loss. The mini-batch size was always 50 and the number of epochs was always 25.

4 Results and Discussion

As baseline, we propose the system described in (Suárez-Paniagua and Segura-Bedmar, 2018), which exploited a CNN with max pooling operations, obtaining an F1 of 64.56%.

Table 4 shows the results for both approaches: one-stage versus two-stages architectures. As described above, we aim to compare two different pre-trained word embeddings models to initialize our networks. Both pre-trained models are described in Pyysalo et al. (2013). The first model (from now on, we call it as biomedical) was trained only using biomedical literature such as PubMed and PMC, while the second one (from now on general) was trained also using general texts taken from Wikipedia.

From the results in Table 4, it is seen that the use of GRU is superior to the CNN architecture, since both compared architectures (one-step and two-step) show greater performance than the baseline (Suárez-Paniagua and Segura-Bedmar, 2018), even using general domain word embeddings.

In the one-stage approach, the biomedical pre-trained word embedding models provides slightly better results than using the general pre-trained word embedding model (see Table 4).

Only the precision of the *int* class is better using the general pre-trained word embedding model. However, this class is the least represented in the corpus: there are 189

	One-S. (biomedical)			One-S. (general)			Two-S. (biomedical)			Two-S. (general)		
	P	R	F1	P	R	F1	P	R	F1	P	R	F1
Int	0.65	0.33	0.44	0.67	0.31	0.43	0.68	0.38	0.48	0.53	0.34	0.45
Advise	0.78	0.75	0.77	0.75	0.71	0.73	0.74	0.79	0.76	0.69	0.80	0.74
Effect	0.68	0.69	0.69	0.66	0.68	0.67	0.64	0.70	0.67	0.61	0.70	0.65
Mechanism	0.72	0.66	0.69	0.72	0.60	0.66	0.68	0.66	0.67	0.66	0.73	0.69
Micro-average	0.71	0.66	**0.69**	0.70	0.62	**0.66**	0.67	0.67	**0.67**	0.64	0.70	**0.67**
Macro-avg.	0.71	0.61	**0.65**	0.70	0.59	**0.62**	0.69	0.63	**0.65**	0.62	0.64	**0.63**

Table 1: Comparative results for one-stage and two-stages systems for DDI extraction

int instances in the training set and 96 in the test set, while there are 1687 and 360 *effect* instances. Therefore, results from of class are less informative of general classifier behaviour.

Finally, the McNemar test (McNemar, 1947) of homogeneity was performed to compare the results of the system using both word embeddings. This test assesses whether the proportion of errors of two classifiers is statistically significant. In this case, it is, since the p-value was smaller than 0.05 (0.02). Therefore, in the case of the one-stage approach, using a word embedding trained with texts from the domain of knowledge of the problem appears to be the optimum choice.

Table 4 also compares the results obtained with the two different word embeddings models for the two-stage approach. The general pre-trained word embedding model obtains a micro-average F1 score of 66.73%, slightly smaller than the metric obtained with the biomedical pre-trained model.

Both word embedding models result in similar performances. However, there are a few differences whose discussion ensue. First, the recall of the *mechanism* class is 7 points higher using the domain-combined word embedding: there are less *mechanism* instances misclassified as other classes. This effect is not observed in the other positive classes. Therefore, there may be a difference in the way information is encoded in *mechanism* sentences and it is affected by the use of a word embedding that incorporates information from domains of knowledge different from the biomedical one.

Second, there are 14 points of difference in the precision for the *Int* class. The reason is a 209% increment in the *negative* instances misclassified as *Int*. However, the total number of *Int* instances is extremely low: the 209% increment corresponds to a change from 11 to 23 misclassified instances.

Finally, the McNemar test of homogeneity has been performed and according to the resulting p-value, p=0.054, the difference in the proportion of errors is on the edge of being significant using the two word embeddings.

We now compare the performance of the one-stage and two-stage approaches. If we use the biomedical pre-trained word embedding model for both approaches, the micro-average F1 score of the one-stage System is 68.54% on the test set, while the two-stage approach achieves an F1 score of 67.45%.

The one-stage approach is superior for all classes (except for *Int*) on the relevant metric, F1 score. Nonetheless, the two-stage system is superior in recall for classes *Int*, *advise* and *effect*. Then, in the one-stage approach, there were more false negatives, while in the two-stage approach, there were more false positives. This phenomenon may be a consequence of the two-stage approach. In it, the first stage (detector) rules out *negative* instances. However, its performance is not perfect, and *negative* instances may be misclassified as *positive* instances, which are entered to the second stage, the classifier. There, these *negative* instances are tagged as *mechanism*, *advise*, *effect* or *Int*. And therefore, the number of false positives increases for the positive classes. An example of this phenomenom is shown in Table 4, sentences 2 and 3.

To statistically compare both systems, McNemar test of homogeneity was performed. The resulting p-value, p=0.147, does not allow to say that both systems are significantly different in their error proportion.

Performances of one-stage and two-stage systems are comparable. The one-stage architecture obtains a slightly better performance. This may indicate that one-stage systems are more suitable for DDI extraction.

Some light experimentation showed that both proposed architectures misclassify sentences with more than two drugs mentioned. As seen in example 1 from Table 4, two drugs adjacent in the sentence are not considered to interact, but when one of them is further away, an interaction is wrongly undetected by

Sentence	One-S.	Two-S.	Truth
(1) **DRUGA DRUGB** as well as other DRUGN may affect ... to DRUGN	None	None	None
DRUGN **DRUGA** as well as other DRUGN may affect ... to **DRUGB**	None	effect	effect
(2) the use of **DRUGA** before **DRUGB** to attenuate ... DRUGN has not been studied	None	effect	None
(3) ... drug drug interaction studies between **DRUGA** and **DRUGB** are inconclusive	None	effect	None
(4) **DRUGA** inhibits the glucuronidation of **DRUGB** and could possibly potentiate DRUGN	None	effect	mechanism
DRUGA inhibits the glucuronidation of DRUGN and could possibly potentiate **DRUGB**	effect	None	effect
DRUGN inhibits the glucuronidation of **DRUGA** and could possibly potentiate **DRUGB**	None	None	None

Table 2: Examples of errors in prediction

the one-stage system. In addition, handling with negation and uncertainty are recurrent challenges in NLP systems. Example 4 from Table 4 shows how both architectures commit mistakes when dealing with the construction *could possible potentiate*. A greater corpus could allow the network to reduce those mistakes and others.

5 Conclusions and Future Directions

Most previous studies on DDI extraction usually opt for one-stage architecture. This work proposes a comparison of one-stage versus two-stage architectures. The two-stage approach first detects the positive instances and rules out the negative instances. Then, only the positive instances are classified in a second stage. Both approaches use GRU, a type of recurrent neural network unit.

Results did not show a significant difference in the error distribution of both systems. However, F1 score is slightly higher for the one-stage System than for the two-stage (68.54% vs 67.45%).

Since performances are comparable, this suggests that the use of one-stage architectures is more suitable in deep learning based DDI extractors because of its simpler design. On the other hand, experiments show that the one-stage architecture requires more training time, since more sentences are synthetically created in the oversampling phase and therefore more instances are used during training. The pre-trained word embedding model created from biomedical literature also provides better performance than the general word embedding model.

As future work, we plan to explore hybrid architectures which exploit advantages of both CNN and LSTM models.

Acknowledgments

This work was supported by the Research Program of the Ministry of Economy and Competitiveness - Government of Spain, (DeepEMR project TIN2017-87548-C2-1-R).

References

Abacha, A. B., M. F. M. Chowdhury, A. Karanasiou, Y. Mrabet, A. Lavelli, and P. Zweigenbaum. 2015. Text mining for pharmacovigilance: Using machine learning for drug name recognition and drug–drug interaction extraction and classification. *Journal of biomedical informatics*, 58:122–132.

Cho, K., B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio. 2014. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*.

Chowdhury, M. F. M. and A. Lavelli. 2013. Exploiting the scope of negations and heterogeneous features for relation extraction: A case study for drug-drug interaction extraction. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 765–771.

Chung, J., C. Gulcehre, K. Cho, and Y. Bengio. 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. In *NIPS 2014 Workshop on Deep Learning*.

Dewi, I. N., S. Dong, and J. Hu. 2017. Drug-drug interaction relation extraction with deep convolutional neural networks. In *IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*.

Glorot, X. and Y. Bengio. 2010. Understanding the difficulty of training deep feedforward neural networks. In *13th International Conference on Artificial Intelligence and Statistics (AISTATS)*.

Herrero-Zazo, M., I. Segura-Bedmar, P. Martínez, and T. Declerck. 2013. The DDI corpus: An annotated corpus with pharmacological substancesand drug–drug interactions. *Journal of Biomedical Informatics*, 46:914–20.

Kavuluru, R., A. Rios, and T. Tran. 2017. Extracting Drug-Drug interactions with Word and Character-Level Recurrent Neural Networks. In *IEEE International Conference on Healthcare Informatics (ICHI)*.

Kim, S., H. Liu, L. Yeganova, and W. J. Wilbur. 2015. Extracting drug–drug interactions from literature using a rich feature-based linear kernel approach. *Journal of biomedical informatics*, 55:23–30.

Lai, S., L. Xu, K. Liu, and J. Zhao. 2015. Recurrent convolutional neural networks for text classification. In *Twenty-ninth AAAI conference on artificial intelligence*.

McNemar, Q. 1947. Note on the sampling error of the difference between correlated proportions or percentages. *Psychometrika*, 12.

Mohamed, A.-R., G. Hinton, and G. Penn. 2010. Understanding the difficulty of training deep feedforward neural networks. In *Acoustics, Speech and Signal Processing (ICASSP), 2012 IEEE International Conference*.

Pyysalo, S., F. Ginter, H. Moen, T. Salakoski, and S. Ananiadou. 2013. Distributional Semantics Resources for Biomedical Text Processing. In *Proceedings of LBM 2013*.

Sainath, T. N., O. Vinyals, A. Senior, and H. Sak. 2015. Convolutional, long short-term memory, fully connected deep neural networks. In *Acoustics, Speech and Signal Processing (ICASSP), IEEE International Conference*.

Segura-Bedmar, I. 2010. *Application of Information Extraction techniques to pharmacological domain: Extracting drug-drug interactions*. Ph.D. thesis, Department of Computer Science, University Carlos III of Madrid.

Segura-Bedmar, I., P. Martinez, and C. de Pablo-Sánchez. 2011. Using a shallow linguistic kernel for drug–drug interaction extraction. *Journal of biomedical informatics*, 44(5):789–804.

Segura-Bedmar, I., P. Martínez, and M. Herrero-Zazo. 2013. SemEval-2013 task 9 : Extraction of drug-drug interactions from biomedical texts (DDIExtraction 2013). In *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*, pages 341–350, Atlanta, Georgia, USA, June. Association for Computational Linguistics.

Segura Bedmar, I., P. Martinez, and D. Sánchez Cisneros. 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 2011*.

Suárez-Paniagua, V. and I. Segura-Bedmar. 2018. Evaluation of pooling operations in convolutional architectures for drug-drug interaction extraction. *BMC bioinformatics*, 19(8):209.

Sun, X., L. Ma, X. Du, J. Feng, and K. Dong. 2018. Deep Convolution Neural Networks for Drug-Drug Interaction Extraction. In *IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*.

Thomas, P., M. Neves, T. Rocktäschel, and U. Leser. 2013. Wbi-ddi: drug-drug interaction extraction using majority voting. In *Second Joint Conference on Lexical and Computational Semantics (* SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*, pages 628–635.

Wishart, D. e. a. 2017. Drugbank 5.0: a major update to the drugbank database for 2018. *Nucleid Acird Research*, 8.

Yi, Z., S. Li, J. Yu, and Q. Wu. 2017. Drug-drug Interaction Extraction via Recurrent Neural Network with Multiple Attention Layers. In *International Conference on Advanced Data Mining and Applications*.

Zaremba, W., I. Sutskever, and O. Vinyals. 2015. Recurrent Neural Network Regularization. *arXiv preprint arXiv:1409.2329*, 1.

Zheng, W., H. Lin, L. Luo, Z. Zhao, Z. Li, Y. Zhang, Z. Yang, and J. Wang. 2017. An attention-based effective neural model for drug-drug interactions extraction. *BMC Bioinformatics*, 18.