Learning local and global contexts using a convolutional recurrent network model for relation classification in biomedical text

Desh Raj and **Sunil Kumar Sahu** and **Ashish Anand**
Department of Computer Science and Engineering
Indian Institute of Technology Guwahati
Guwahati, India

Abstract

The task of relation classification in the biomedical domain is complex due to the presence of samples obtained from heterogeneous sources such as research articles, discharge summaries, or electronic health records. It is also a constraint for classifiers which employ manual feature engineering. In this paper, we propose a convolutional recurrent neural network (CRNN) architecture that combines RNNs and CNNs in sequence to solve this problem. The rationale behind our approach is that CNNs can effectively identify coarse-grained local features in a sentence, while RNNs are more suited for long-term dependencies. We compare our CRNN model with several baselines on two biomedical datasets, namely the i2b2-2010 clinical relation extraction challenge dataset, and the SemEval-2013 DDI extraction dataset. We also evaluate an attentive pooling technique and report its performance in comparison with the conventional max pooling method. Our results indicate that the proposed model achieves state-of-the-art performance on both datasets.[1]

1 Introduction

Relation classification is the task of identifying the semantic relation present between a given pair of entities in a piece of text. Since most search queries are some forms of binary factoids (Agichtein et al., 2005), modern question-answering systems rely heavily upon relation classification as a preprocessing step (Fleischman et al., 2003; Lee et al., 2007). Accurate relation classification also facilitates discourse processing and precise sentence interpretations. Hence, this task has witnessed a great deal of attention over the last decade (Mintz et al., 2009; Surdeanu et al., 2012).

In the biomedical domain, in particular, extracting such tuples from data may be essential for identifying protein and drug interactions, symptoms and causes of diseases, among others. Further, since clinical data tends to be obtained from multiple (and diverse) information sources such as journal articles, discharge summaries, and electronic patient records, relation classification becomes a more challenging task.

To identify relations between entities, a variety of lexical, syntactic, or pragmatic cues may be exploited, which results in a challenging variability in the type of features used for classification purpose. Due to this variability, a number of approaches have been suggested, some of which rely on features extracted from POS tagging, morphological analysis, dependency parsing, and world knowledge (Kambhatla, 2004; Santos et al., 2015; Suchanek et al., 2006; Mooney and Bunescu, 2005; Bunescu and Mooney, 2005). Deep learning architectures have recently gathered much interest because of their ability to conveniently extract relevant features without the need of explicit feature engineering. For this reason, a number of convolutional and recurrent neural network models (Zeng et al., 2014; Xu et al., 2015b) have been used for this task.

In this paper, we propose a model that uses recurrent neural networks (RNNs) and convolutional neural networks (CNNs) in sequence to learn global and local context, respectively. We refer to this as CRNN, following the naming convention used in (Huynh et al., 2016). We argue that in order for any classification task to be effective, the

[1]The code for the can be found at: `https://github.com/desh2608/crnn-relation-classification`.

regression layer must see a complete representation of the sentence, i.e., both short and long-term dependencies must be appropriately represented in the sentence embedding. This argument forms the basis of our approach. In a deep learning framework, since the complete information available to the classifier at the top-level is obtained through manipulation of the sentence embedding itself, the task of relation classification essentially emulates other popular objectives such as text classification and sentiment analysis if the representation for the entity types are integrated in the sentence. Although our proposed model uses RNNs and CNNs in sequence, it is only two layers deep, as opposed to the very deep architectures proposed earlier (Conneau et al., 2016). This simplicity allows for intuitive understanding of each level of the model, while still learning a sufficiently complex representation of the input sentence.

In addition to local and global contexts, we also experiment with attention for relation classification. Although attention as a concept is relatively well-known, especially in computational neuroscience (Itti et al., 1998; Desimone and Duncan, 1995), it became popular only recently with applications to image captioning and machine translation (Xu et al., 2015a; Vinyals et al., 2015; Bahdanau et al., 2014). Attention has also been employed to some success in relation classification tasks (Wang et al., 2016a; Zhou et al., 2016a). In our experiments, we use an attention-based pooling strategy and compare the results with those obtained using conventional pooling methods. Our model variants are accordingly named **CRNN-Max** and **CRNN-Att**, depending upong the pooling scheme used.

Our model is distinctive in that it does not rely upon any linguistic feature for relation classification. In domains such as biomedicine, texts may not always be written in syntactically/grammatically correct form. Furthermore, lack of necessary training data may not provide good feature extractors such as those in generic domains. Hence, we explored only models without any extra features. Of course, adding other features such as part-of-speech taggers or dependency parsers, if they are available easily, may improve the performance further. Our key contributions in this paper are as follows:

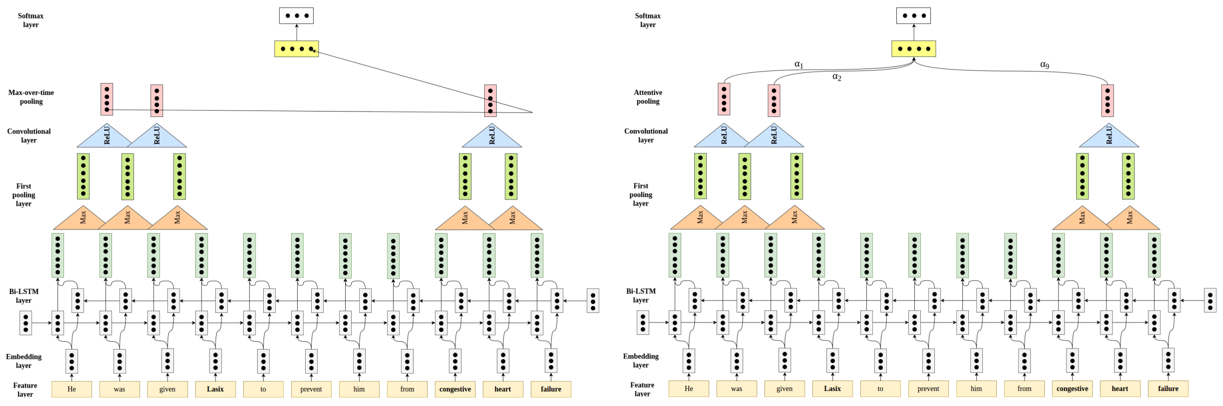
- We propose and validate a two-layer architecture comprising RNNs and CNNs in sequence for relation classification in biomedical text. Our model's performance is comparable to the state-of-the-art on two benchmark datasets, namely the i2b2-2010 clinical relation extraction challenge, and the SemEval-2013 DDI extraction dataset, without any need for handcrafted features.
- We analyze and discuss why such a model effectively captures short and long-term dependencies in a sentence, and demonstrate why this representation facilitates classification.
- We evaluate an attention-based pooling technique and compare its performance with conventional pooling strategies.
- We provide evidence to further the argument in favor of using RNNs to obtain regional embeddings in a sentence.

2 Related Research

CNNs have been effectively employed in NLP tasks such as text classification (Kim, 2014), sentiment analysis (Dos Santos and Gatti, 2014), relation classification (Zeng et al., 2014; Nguyen and Grishman, 2015b), and so on. Similarly, RNN models have also been used for similar tasks (Johnson and Zhang, 2016). The improved performance of these models is due to several reasons:

1. Pretrained word vectors are used as inputs for most of these models. These embeddings capture the semantic similarity between words in a global context better than one-hot representations.
2. CNNs are capable of learning local features such as short phrases or recurring n-grams, similar to the way they provide translational, rotational and scale invariance in vision.
3. RNNs utilize the word order in the sentence, and are also able to learn the long-term dependencies.

These observations amply motivate a model which captures both short-term and long-term dependencies using a combination of CNNs and RNNs to form a robust representation of the sentence. Earlier, researchers have proposed RCNN models that compute "regional embeddings" using a CNN at the first level, and these embeddings



(a) CRNN-Max **(b)** CRNN-Att

Figure 1: Architecture of the proposed models. For representation purpose, the following configuration has been used: $d = n_O = 3$, $f_1 = f_2 = 2$, $n_c = 4$, and $|C| = 3$.

are then fed into an RNN layer which uses sequence information to generate the sentence representation (Huynh et al., 2016; Wang et al., 2016b; Chen et al., 2017; Nguyen and Grishman, 2015a). These models are similar to ones that have also been employed to some success for visual recognition (Donahue et al., 2015). However, such models are still limited because the RNN may "forget" features that occurred in the past if the sequence is very long.

We solve this problem by obtaining the output of the RNN at each time step (or word), and then pooling small phrases. This method of using a "recurrent+pooling" module for regional embedding is inspired from (Johnson and Zhang, 2016), who showed that for text categorization, embeddings of text regions, which can convey higher-level concepts than single words in isolation, are more useful than word embeddings. We also experiment with attention-pooling to integrate weighted features from discontinuous regions in the sentence.

3 Proposed Method

Given a sentence S with marked entities e_1 and e_2, belonging to entity types t_1 and t_2, respectively, and a set of relation classes $C = \{c_1, \ldots, c_m\}$ we formulate the task of identifying the semantic relation as a supervised classification problem, i.e., we learn a function $f : (\mathbf{S}, E, T) \rightarrow C$, where $\mathbf{S}$ is the set of all sentences, E is the set of entity pairs, and T denotes the set of entity types. Our training objective is to learn a joint representation of the sentence and the entity types, such that a softmax regression layer predicts the correct label. To learn such an embedding, we propose a two-layer neural network architecture consisting of a "recurrent+pooling" layer and a "convolutional+pooling" layer in sequence. This architecture is diagrammatically described in Fig. 1, and the remainder of this section explains each of the layers in detail.

3.1 Embedding layer

The only features we use from S are the words themselves. The vector representation of these words is obtained using the GloVe method (Pennington et al., 2014).

Pre-trained word vectors are used for the word embeddings and the words not present in the embeddings list are initialized randomly. All the word vectors are updated during training.

3.2 Recurrent layer

RNN is a class of artificial neural networks which utilizes sequential information and maintains history through its intermediate layers (Graves et al., 2009). We use long short-term memory (LSTM) based model (Hochreiter and Schmidhuber, 1997), which uses memory and gated mechanism to compute the hidden state. In particular we use a bidirectional LSTM model (Bi-LSTM) similar to the ones used in (Graves, 2013; Huang et al., 2015).

Let $h_l^{(t)}$ and $h_r^{(t)}$ be the outputs obtained from the forward and backward direction of the LSTM at time t. Then the combined output is given as

$$z^{(t)} = h_l^{(t)} : h_r^{(t)}, \quad z^{(t)} \in \mathbb{R}^{n_O}. \qquad (1)$$

where : denotes the concatenation operation. We obtain the output at each word and pass it to the first pooling layer.

3.3 First pooling layer

The recurrent layer generates word-level embeddings that incorporate information from the past and future context. Sometimes the word itself may not be important for the sentence representation, and in such cases, it may be better to extract the most important features from short phrases using a pooling technique. If f_1 denotes the length of the filter used for pooling, and $(z_1, \ldots, z_m)$ is the sequence of vectors obtained from the previous layer, then

$$p = (p_1, p_2, \ldots, p_{m-f_1+1}), \tag{2}$$

where $p_i \in \mathbb{R}^{n_O}$ is given as

$$p_i = \max_{1 \leq j \leq f_1} [z_{i+j}], \tag{3}$$

i.e. the maximum among all vectors z_{i+1} to z_{i+f_1}.

3.4 Convolutional layer

We apply convolution on p to get local features from each part of the sentence (Collobert and Weston, 2008). Consider a convolutional filter parametrized by weight vector $w_c \in \mathbb{R}^{n_O * f_2}$, where f_2 is the length of filter. Then the output sequence of convolution layer would be

$$h_c^i = f(w_c \cdot p^{i:i+f_2-1} + b_c), \tag{4}$$

where $i = 1, 2, \ldots, m - f_1 - f_2 + 2$, $\cdot$ is dot product, f is the rectifier linear unit (ReLU) function ($f(x) = \max\{0, x\}$), and $b_c \in \mathbb{R}$ is the bias term. The parameters w_c and b_c are shared across all convolutions $i = 1, 2, \ldots, m - f_1 - f_2 + 2$. On applying n_c such filters, we obtain an output matrix $H_c \in \mathbb{R}^{n_c \times (m-f_1-f_2+2)}$.

3.5 Second pooling layer

The output of the convolutional layer is of variable length ($m - f_1 - f_2 + 2$), since it depends on the length m of the input sentence. To obtain fixed length global features for the entire sentence, we apply pooling over the entire sequence. For this, we experiment with two different pooling schemes based on which our model has two variations, namely CRNN-Max and CRNN-Att.

3.5.1 Max pooling over time

Max pooling over time (Collobert and Weston, 2008) takes the maximum over the entire sentence, with the assumption that all the relevant information is accumulated in that position. Since the input to this layer are the local convolved vectors, this strategy essentially extracts the most important features from several short phrases. The output is then given as

$$z_{pool} = \max_{1 \leq i \leq (m-f_1-f_2+2)} [h_c^i], \tag{5}$$

where $z_{pool} \in \mathbb{R}^{n_c}$ is the dimension-wise maximum over all h_c^i's.

3.5.2 Attention-based pooling

A max pooling scheme may fail when important cues are distributed across different clauses in the sentence. We solve this problem by using an attention-based pooling scheme, which obtains an optimal feature dimension-wise by taking weighted linear combinations of the vectors. These weights are trained using an attention mechanism such that more important features are weighed higher (Bahdanau et al., 2014; Yang et al., 2016; Zhou et al., 2016b). The attention mechanism produces a vector α of size $m - f_1 - f_2 + 2$, and the values in this vector are the weights for each phrase obtained from the convolutional layer feature vectors.

$$\begin{aligned} H_{att} &= \tanh(W_1^{\alpha} H_c) \\ \alpha &= Softmax({W_2^{\alpha}}^T H_{att}) \\ z_{att} &= \alpha H_c^T \end{aligned} \tag{6}$$

Here, H_c is the matrix of CNN output vectors, $W_1^{\alpha}, W_2^{\alpha} \in \mathbb{R}^{n_c \times n_c}$ is the parameter matrix, $\alpha \in \mathbb{R}^{m-f_1-f_2+2}$ are the attention weights, and $z_{att} \in \mathbb{R}^{n_c}$ is the output of the pooling layer. The attention weights are a function of the input sentence, and hence α is different for every sentence.

3.6 Fully connected and softmax

To obtain a classifier over the extracted global features, we use a fully connected layer consisting of $|C|$ nodes, where C is the set of all possible relation classes, followed by a softmax layer to generate a probability distribution over the set of all possible labels. The final output is given as

$$p(c_i|x) = Softmax(W_i^o z + b_i^o), \tag{7}$$

where W^o and b^o are the weight and bias parameters, and z may be either z_{pool} or z_{att}, depending on the second pooling layer scheme. The predicted output y' is obtained as

$$y' = \arg\max_{c_i \in C} p(c_i|x). \tag{8}$$

Class	Train size	Test size
TrCP	436	108
TrAP	2131	532
TrWP	109	26
TrIP	165	41
TrNAP	140	34
TeRP	2457	614
TeCP	409	101
PIP	1776	443
None	44588	11146
Total	**52211**	**13045**

Table 1: Number of training and testing instances for each relation type in the i2b2 dataset.

4 Experiments

4.1 Datasets

We have used 2 datasets for experimentation, namely the i2b2-2010 clinical relation extraction challenge dataset (Sun et al., 2013), and the SemEval-2013 DDI extraction dataset (Segura Bedmar et al., 2013).

i2b2-2010 relation extraction

This dataset contains sentences from discharge summaries collected from three different hospitals and have 8 relation types: *treatment caused medical problems* (**TrCP**), *treatment administered medical problem* (**TrAP**), *treatment worsen medical problem* (**TrWP**), *treatment improve or cure medical problem* (**TrIP**), *treatment was not administered because of medical problem* (**TrNAP**), *test reveal medical problem* (**TeRP**), *test conducted to investigate medical problem* (**TeCP**), and *medical problem indicates medical problem* (**PIP**). If a sentence has more than two entities, we make an instance for each pair. Since only 170 of the 394 original training documents and 256 of the 477 testing documents were available for download, we combined all the training and testing instances, and then split it in a 80:20 ratio for training and test sets respectively. The statistics of the dataset are described in Table 1.

SemEval 2013 DDI extraction

This dataset contains annotated sentences from two sources, Medline abstracts (biomedical research articles) and DrugBank database (documents written by medical practitioners). The dataset is annotated with following four kinds of interactions: *advice* (opinion or consultation related to the simultaneous use of the two drugs), *effect* (effect of the DDI together with pharmacodynamic effect or mechanism of interaction), *mechanism* (pharmacokinetic mechanism), and *int* (drug interaction without any other information). Dataset provides the training and test instances by sentences. Similar to i2b2 relation extraction dataset if a sentence has more than two drug names, all possible pairs of drugs in the sentence have been separately annotated, such that a single sentence having multiple drug names leads to separate instances of drug pairs and corresponding interaction. Statistics of the dataset (along with negative instance filtering, discussed in Section 4.1.1) is shown in Table 2.

Class	Train		Test	
	Before	**After**	**Before**	**After**
Mechanism	1318	1264	302	302
Effect	1685	1620	360	360
Advice	826	820	221	221
Int	189	140	96	96
None	23756	12651	4737	3046
Total	**4018**	**3844**	**979**	**979**

Table 2: Number of training and testing instances for each relation type in the DDI extraction dataset.

4.1.1 Preprocessing

As a preprocessing step, we replace the entities in the i2b2 dataset with the corresponding entity types. For instance, the sentence: "He was given *Lasix* to prevent him from *congestive heart failure*." was converted to: "He was given *TREATMENT_A* to prevent him from *PROBLEM_B*." Similarly, for the DDI extraction dataset, the two targeted drug names are replaced with DRUG-A and DRUG-B respectively, and other drug names in the same sentence are replaced with DRUG-N. Further, all numbers were replaced with the keyword *NUM*. Similar to the earlier studies (Sahu and Anand, 2017; Liu et al., 2016; Rastegar-Mojarad et al., 2013), negative instances were filtered from training sets.

4.2 Implementation details

Pretrained 100-dimensional word vectors in the embedding layer are obtained using the GloVe method (Pennington et al., 2014) trained on a corpus of PubMed open source articles (Muneeb et al., 2015), and are updated during the training process. We use both l_2 regularization and dropout (Srivastava et al., 2014) techniques for regularization. Dropout is applied only on the output of the second pooling layer, and it prevents co-adaptation of hidden units by randomly dropping few nodes. After tuning the hyperparameters on a validation set (20% of training set), the values of 0.01 (0.001) and 0.7 (0.5) were found optimal for the regularization parameter and dropout for the i2b2 (DDI extraction) dataset, respectively. We use Adam technique (Kingma and Ba, 2014) to optimize our loss function, with a learning rate of 0.01. For all the models, n_O and n_C were tuned on the validation set, and values of 200 and 100 were found to be optimal. Hyperparameters of baseline methods were taken from the values suggested in the respective papers. Entire neural network parameters and feature vectors are updated while training. We have implemented the proposed model in Python language using the Tensorflow package (Abadi et al., 2016). We experiment with different filter sizes for f_1 and f_2 and discuss the results in Section 5.1.

4.3 Baseline methods

We compare our models with 5 methods that have earlier been used for relation classification to satisfactory results. These baselines were selected for one of the following three purposes.

Feature-based methods

We selected a feature-based SVM classifier (Rink et al., 2011) that uses several handcrafted features such as distance of word from entities, POS tags, chunk tags, etc., to compare whether our models were able to outperform classifiers with rigorous feature engineering. It is to be noted that we use our own implementation of the SVM classifier (using the scikit-learn (Pedregosa et al., 2011) library), using features as described in (Sahu et al., 2016).

Single-layer neural networks

We selected a multiple-filter CNN with max-pooling (Sahu et al., 2016) and an LSTM model with max and attentive pooling (Sahu and Anand, 2017). In Section 5.5, we compare our models with these single layer models to justify using a combination of RNN and CNN to learn long-term and short-term dependencies, respectively. To observe the effect of the network model independent of the feature set, we use only the word embeddings as features for each of these models. Further, we used the same hyperparameters as mentioned in the respective papers.

Recurrent convolutional neural network

This model, inspired from (Wang et al., 2016b), obtains regional embeddings using a convolutional layer. These are then fed into a recurrent layer and a single output is obtained after traversing the entire sequence. We compare our models with this RCNN model to observe the effect of obtaining outputs at every word, as opposed to at the end of the sequence.

$f_1 \backslash f_2$	**2**	**3**	**4**	**5**	**6**
1	59.97	58.96	59.30	59.18	60.03
2	59.84	56.69	60.89	**62.45**	61.03
3	60.46	61.77	58.85	57.34	59.81

Table 3: Average F1 scores on varying filter sizes f_1 and f_2 in the CRNN-Att model for i2b2 dataset.

5 Results and Discussion

5.1 Effect of filter sizes f_1 and f_2

We experiment with various combinations of filter sizes f_1 and f_2 on the i2b2 dataset using our CRNN-Att model. Since f_1 denotes the size of the first pooling filter, it essentially represents the amount of information present in a regional embedding that is fed into the convolutional layer. If f_1 is too small ($f_1 = 1$, i.e., no pooling), embeddings from seemingly unimportant words may get through, and if it is large ($f_1 \geq 3$), individual embeddings may get pooled such that a few words dominate the majority of regions. For the filter size f_2 in the convolutional layer, a mid-range value (4 to 6) was found to work well. This may be because this layer learns to identify short phrases which are usually of this length. These observations were common for both datasets. The F1 scores for various combinations of filter sizes on the i2b2 data are shown in Table 3. In the remaining experiments, we choose (f_1,f_2) = (2,5) for both our model variants.

5.2 Initialization and tuning of word embeddings

The only feature used in our models is the word vectors for every word in the sentence. We perform several experiments on the i2b2 data to observe the effect of word vector initialization and update on the model performance. The results are summarized in Table 5.

Interestingly, the best performing model uses randomly initialized word embeddings that are not updated during training. This is in contrast to earlier studies (Sahu and Anand, 2017; Collobert and Weston, 2008) where pretrained embeddings

Model	**i2b2-2010**			**DDI extraction**		
	Precision	**Recall**	**F1 score**	**Precision**	**Recall**	**F1 score**
SVM (Rink et al., 2011)	**67.44**	57.85	59.31	65.39	40.13	49.74
CNN-Max (Sahu et al., 2016)	55.73	50.08	49.42	68.15	46.58	54.05
LSTM-Max (Sahu and Anand, 2017)	57.54	55.40	55.60	**73.98**	59.96	65.41
LSTM-Att (Sahu and Anand, 2017)	65.23	56.77	60.04	53.43	**64.86**	58.27
RCNN (Wang et al., 2016b)	50.07	45.34	46.47	–	–	–
CRNN-Max	67.91	61.98	**64.38**	72.91	60.88	**65.89**
CRNN-Att	64.62	**62.14**	62.45	69.03	59.04	63.24

Table 4: Comparison of our proposed models CRNN-Max and CRNN-Att, with baselines, on the i2b2-2010 and DDI extraction datasets.

Initialization	**update**	**CRNN-Max**	**CRNN-Att**
Random	Trainable	62.78	61.19
Random	Non-trainable	**64.38**	61.51
PubMed	Trainable	60.60	**62.45**
PubMed	Non-trainable	58.49	59.35

Table 5: Effect of initialization and update of word embeddings in our proposed models, in terms of F1 score, using the i2b2-2010 datset.

Class	**Size**	**SVM**	**CNN**	**LSTM-Max**	**RCNN**	**CRNN-Max**	**CRNN-Att**
TrCP	108	34.90	34.01	35.48	18.30	43.18	**47.66**
TrAP	532	63.48	46.69	58.74	45.15	**67.39**	63.94
TrWP	26	7.41	10.26	0.00	0.00	**16.67**	9.52
TrIP	41	9.09	21.74	0.00	0.00	25.71	**34.48**
TrNAP	34	5.13	15.87	0.00	0.00	**36.36**	18.60
TeRP	614	**80.44**	63.52	73.50	67.01	80.32	76.31
TeCP	101	30.30	27.63	25.20	11.48	39.46	**39.76**
PIP	443	49.44	49.30	51.54	45.05	**58.04**	55.53

Table 6: Classwise performance (in terms of F1 score) of various models on the i2b2 dataset.

usually improved model performances by 3-4%. However, this result aligns with the observations made in (Johnson and Zhang, 2015) and supports the argument for one-hot LSTMs. It may be enlightening to discuss why such a result is obtained.

First, we note that in the formulas for LSTM, e.g., $\mathbf{u}_t = \tanh(\mathbf{W}^{(u)}\mathbf{x}_t + \mathbf{U}^{(u)}\mathbf{h}_{t-1} + \mathbf{b}^{(u)})$, if $\mathbf{x}_t$ is the one-hot representation of a word, the term $\mathbf{W}^{(u)}\mathbf{x}_t$ serves as a word embedding. Thus, a one-hot LSTM inherently includes a word embedding in its computation. Further, a word vector lookup is a linear operation, and hence it may be merged into the LSTM layer itself by multiplying the LSTM weights by the word embedding matrix. This means that the expressive power of an LSTM which uses pretrained vectors is the same as that of one which uses randomly initialized word embeddings. It has also been shown in earlier studies that pretrained embeddings do not improve the performance of networks as the number of layers increases.

Johnson et al. (2015) even argued that the embedding layer can be replaced with a one-hot representation without compromising on the performance. Empirically, inclusion of an embedding layer makes training from scratch more difficult, even with the help of adaptive learning rates. Similar observations have been made regarding CNNs (Kim, 2014; Johnson and Zhang, 2014).

5.3 Comparison with baseline methods

Table 4 shows the results obtained on the i2b2 and DDI extraction datasets using our proposed models, as compared to the baseline methods. Our models outperform the baselines even without the need for explicit feature engineering. It is interesting to note that our CRNN-Max performs better than the CRNN-Att, and a similar result has also been observed earlier in (Sahu and Anand, 2017).

Class-wise performance analysis

We compare class-wise performance of our models on the i2b2 dataset with some of the baselines, and this is summarized in Table 6. It is evident that performances improve with training size, and from the confusion matrices (not shown here), we found that samples of a lower frequency class were misclassified into a higher frequency class comprising the same entity types. For instance, samples belonging to TrWP (Treatment Worsen medical Problem) were often classified as TrAP (Treatment Administered medical Problem).

5.4 Effect of attention-based pooling

Our CRNN-Att model uses an attention-based technique in the final pooling layer, i.e. it obtains a weighted linear combination of different phrases depending upon their relative importance in the sentence embedding. To confirm this, we visualize attention weights in a CRNN-Att model with $(f_1, f_2) = (1, 3)$, for 5 samples in the i2b2 dataset through a heat map as shown in Figure 2. Since weights are assigned to phrases rather than words, to obtain attention for each word we take the mean

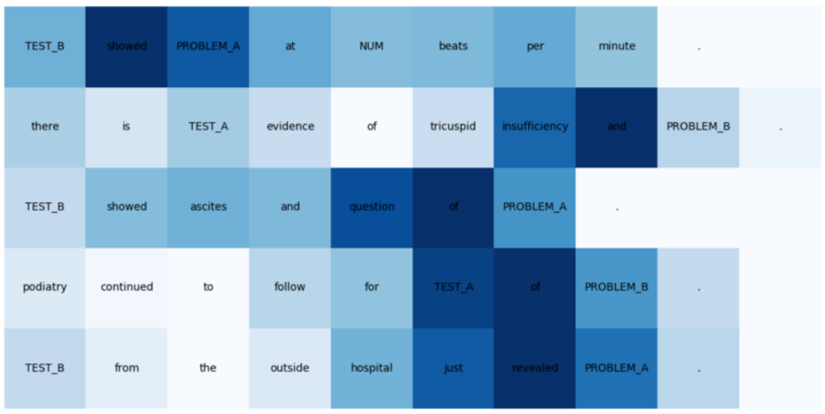


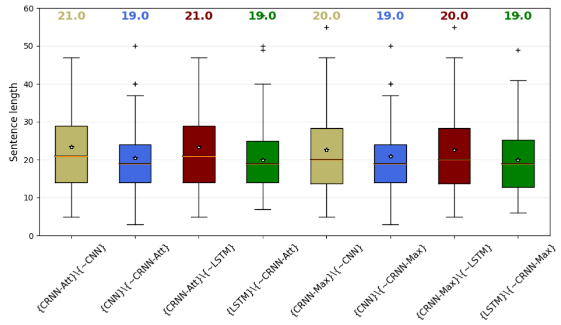
Figure 2: Attention heatmap for 5 sentences selected from the i2b2-2010 dataset. A darker background corresponds to a larger attention weight.

of weights of all phrases that the word is present in. The figure shows that the attentive pooling scheme is able to select important phrases depending upon the classification label. It is evident that the model assigns a higher weight to semantically relevant words such as "showed," "question," and "revealed".

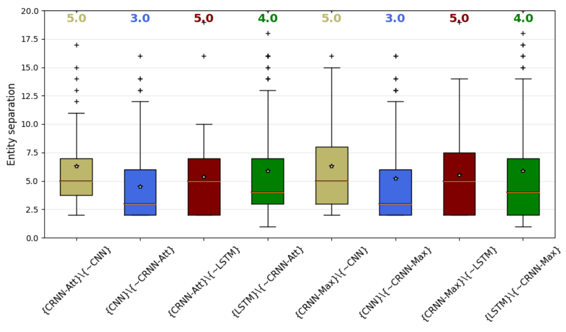
5.5 Long and short term dependencies

We conjecture that our proposed CRNN models perform better than single layer CNNs or RNNs because they capture both local and global contexts efficiently. To confirm our hypothesis, we determine the average sentence lengths and entity separations for several sets of sentences belonging to classes where our models performed well, and for classes where either the CNN model or the LSTM-Max model performed relatively well, for the i2b2-2010 dataset. These results are visualized in the box plots shown in Fig. 3.

From the figure, we note that our models CRNN-Max and CRNN-Att perform significantly better than a CNN model in classifying long sentences with large entity separation, while CNN models work well with shorter sentences where the entities are less separated. This is evident by observing the median and range of lower to upper quartile values in the figure. This confirms our conjecture that our models learn long-term dependencies better than a simple CNN model. Similarly, our proposed models perform better on a larger range of sentence lengths than LSTMs, which may be due to more effective modeling of local contexts.



(a)



(b)

Figure 3: Box plots for distribution of (a) sentence lengths and (b) entity separation for sentence sets. A representation of the form $\{X\}\backslash\{\tilde{}Y\}$ denotes the set of sentences correctly classified by model X but wrongly classified by model Y. The numbers at the top are the median values for each box.

5.6 Effect of linguistic features

The SVM baseline model described earlier consists of the following features obtained for each word in the sentence: word embedding, part-of-speech (POS) tag, chunk tag, distance from first entity, distance from second entity, and entity type. Of these, the entity type feature is already used in our CRNN model in the preprocessing step

Class	Size	SVM	CRNN-Max	#1	#2
TrCP	108	34.90	36.91	11	30
TrAP	532	63.48	68.85	83	93
TrWP	26	7.41	0.00	1	0
TrIP	41	9.09	0.00	2	0
TrNAP	34	5.13	0.00	1	0
TeRP	614	80.44	81.29	69	83
TeCP	101	30.30	36.90	5	14
PIP	443	49.44	60.66	45	110
Total	1899	59.31	63.78	217	330

Table 7: Classwise performance comparison between SVM and CRNN-Max using linguistic features. #1 denotes number of sentences of a class classified correctly by SVM but incorrectly by CRNN-Max; #2 denotes vice-versa.

by replacing the entities with their corresponding types. Furthermore, we have also described experiments with initialization and update of word embeddings.

In this section, we add the four other linguistic features in our proposed model to observe its performance in comparison with the SVM model. Table 7 summarizes this comparison.

Although the F1 scores for the models are relatively close, the precision (P) and recall (R) vary significantly: P is 67.44 and 61.00, while R is 57.85 and 67.54, for the SVM and CRNN-Max models, respectively. Our CRNN-Max model, therefore, is more sensitive while the SVM classifier has a higher specificity. Furthermore, it is evident that SVM outperforms our model only on classes with a disproportionately low instance count. We may argue that due to the presence of more features and less number of records, our model gets over-trained only on the larger classes. This problem may then be avoided with better regularization, to achieve even higher performance.

6 Conclusion

In this work, we proposed and evaluated a two-layer architecture comprising recurrent and convolutional layers in sequence to learn global and local contexts in a sentence, which was then used for relation classification. To the best of our knowledge, this is the first attempt at combining CNNs and RNNs in sequence for a relation classification task in biomedical domain. Two variants of the model, namely CRNN-Max and CRNN-Att, were evaluated on the i2b2-2010 dataset and the SemEval 2013 DDI extraction dataset, and max-pooling was found to perform better than attentive pooling. Even though our method employed only word embeddings as input feature, it was able to conveniently outperform state-of-the-art techniques that use extensive feature engineering. Finally, our results indicated that a "recurrent+pooling" layer effectively generates regional embedding without the need for pretrained word vectors. It would be interesting to see whether one-hot word vectors perform better than randomly initialized embeddings. We may also benefit from probing whether tree-based or non-continuous convolutions work as well as our CRNN models for learning long and short term dependencies for relation classification.

References

Martın Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, et al. 2016. Tensorflow: Large-scale machine learning on heterogeneous distributed systems. *arXiv preprint arXiv:1603.04467* .

Eugene Agichtein, Silviu Cucerzan, and Eric Brill. 2005. Analysis of factoid questions for effective relation extraction. In *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, pages 567–568.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473* .

Razvan C Bunescu and Raymond J Mooney. 2005. A shortest path dependency kernel for relation extraction. In *Proceedings of the conference on human language technology and empirical methods in natural language processing*. Association for Computational Linguistics, pages 724–731.

Guibin Chen, Deheng Ye, Erik Cambria, Jieshan Chen, and Zhenchang Xing. 2017. Ensemble application of convolutional and recurrent neural networks for multi-label text categorization. IJCNN.

Ronan Collobert and Jason Weston. 2008. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th international conference on Machine learning*. ACM, pages 160–167.

Alexis Conneau, Holger Schwenk, Loïc Barrault, and Yann Lecun. 2016. Very deep convolutional networks for natural language processing. *arXiv preprint arXiv:1606.01781* .

Robert Desimone and John Duncan. 1995. Neural mechanisms of selective visual attention. *Annual review of neuroscience* 18(1):193–222.

Jeffrey Donahue, Lisa Anne Hendricks, Sergio Guadarrama, Marcus Rohrbach, Subhashini Venugopalan, Kate Saenko, and Trevor Darrell. 2015. Long-term recurrent convolutional networks for visual recognition and description. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. pages 2625–2634.

Cícero Nogueira Dos Santos and Maira Gatti. 2014. Deep convolutional neural networks for sentiment analysis of short texts. In *COLING*. pages 69–78.

Michael Fleischman, Eduard Hovy, and Abdessamad Echihabi. 2003. Offline strategies for online question answering: Answering questions before they are asked. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics-Volume 1*. Association for Computational Linguistics, pages 1–7.

Alex Graves. 2013. Generating sequences with recurrent neural networks. *arXiv preprint arXiv:1308.0850* .

Alex Graves, Marcus Liwicki, Santiago Fernández, Roman Bertolami, Horst Bunke, and Jürgen Schmidhuber. 2009. A novel connectionist system for unconstrained handwriting recognition. *IEEE transactions on pattern analysis and machine intelligence* 31(5):855–868.

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

Zhiheng Huang, Wei Xu, and Kai Yu. 2015. Bidirectional lstm-crf models for sequence tagging. *arXiv preprint arXiv:1508.01991* .

Trung Huynh, Yulan He, Allistair Willis, and Stefan Rüger. 2016. Adverse drug reaction classification with deep neural networks .

Laurent Itti, Christof Koch, Ernst Niebur, et al. 1998. A model of saliency-based visual attention for rapid scene analysis. *IEEE Transactions on pattern analysis and machine intelligence* 20(11):1254–1259.

Rie Johnson and Tong Zhang. 2014. Effective use of word order for text categorization with convolutional neural networks. *arXiv preprint arXiv:1412.1058* .

Rie Johnson and Tong Zhang. 2015. Semi-supervised convolutional neural networks for text categorization via region embedding. In *Advances in neural information processing systems*. pages 919–927.

Rie Johnson and Tong Zhang. 2016. Supervised and semi-supervised text categorization using lstm for region embeddings. *arXiv preprint arXiv:1602.02373* .

Nanda Kambhatla. 2004. Combining lexical, syntactic, and semantic features with maximum entropy models for extracting relations. In *Proceedings of the ACL 2004 on Interactive poster and demonstration sessions*. Association for Computational Linguistics, page 22.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. *arXiv preprint arXiv:1408.5882* .

Diederik Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* .

Changki Lee, Yi-Gyu Hwang, and Myung-Gil Jang. 2007. Fine-grained named entity recognition and relation extraction for question answering. In *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, pages 799–800.

Shengyu Liu, Buzhou Tang, Qingcai Chen, and Xiaolong Wang. 2016. Drug-drug interaction extraction via convolutional neural networks. *Computational and mathematical methods in medicine* 2016.

Mike Mintz, Steven Bills, Rion Snow, and Dan Jurafsky. 2009. Distant supervision for relation extraction without labeled data. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 2-Volume 2*. Association for Computational Linguistics, pages 1003–1011.

Raymond J Mooney and Razvan C Bunescu. 2005. Subsequence kernels for relation extraction. In *Advances in neural information processing systems*. pages 171–178.

TH Muneeb, Sunil Kumar Sahu, and Ashish Anand. 2015. Evaluating distributed word representations for capturing semantics of biomedical concepts. *Proceedings of ACL-IJCNLP* page 158.

Thien Huu Nguyen and Ralph Grishman. 2015a. Combining neural networks and log-linear models to improve relation extraction. *arXiv preprint arXiv:1511.05926* .

Thien Huu Nguyen and Ralph Grishman. 2015b. Relation extraction: Perspective from convolutional neural networks. In *Proceedings of NAACL-HLT*. pages 39–48.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research* 12:2825–2830.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *EMNLP*. volume 14, pages 1532–1543.

Majid Rastegar-Mojarad, Richard D Boyce, and Rashmi Prasad. 2013. UWM-TRIADS: classifying drug-drug interactions with two-stage SVM and post-processing. In *Proceedings of the 7th International Workshop on Semantic Evaluation*. pages 667–674.

Bryan Rink, Sanda Harabagiu, and Kirk Roberts. 2011. Automatic extraction of relations between medical concepts in clinical texts. *Journal of the American Medical Informatics Association* 18(5):594–600.

Sunil Kumar Sahu and Ashish Anand. 2017. Drug-drug interaction extraction from biomedical text using long short term memory network. *arXiv preprint arXiv:1701.08303* .

Sunil Kumar Sahu, Ashish Anand, Krishnadev Oruganty, and Mahanandeeshwar Gattu. 2016. Relation extraction from clinical texts using domain invariant convolutional neural network. *arXiv preprint arXiv:1606.09370* .

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

Isabel Segura Bedmar, Paloma Martínez, and María Herrero Zazo. 2013. Semeval-2013 task 9: Extraction of drug-drug interactions from biomedical texts (ddiextraction 2013). Association for Computational Linguistics.

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

Fabian M Suchanek, Georgiana Ifrim, and Gerhard Weikum. 2006. Combining linguistic and statistical analysis to extract relations from web documents. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, pages 712–717.

Weiyi Sun, Anna Rumshisky, and Ozlem Uzuner. 2013. Evaluating temporal relations in clinical text: 2012 i2b2 challenge. *Journal of the American Medical Informatics Association* 20(5):806–813.

Mihai Surdeanu, Julie Tibshirani, Ramesh Nallapati, and Christopher D Manning. 2012. Multi-instance multi-label learning for relation extraction. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*. Association for Computational Linguistics, pages 455–465.

Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan. 2015. Show and tell: A neural image caption generator. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. pages 3156–3164.

Linlin Wang, Zhu Cao, Gerard de Melo, and Zhiyuan Liu. 2016a. Relation classification via multi-level attention cnns. In *ACL*.

Xingyou Wang, Weijie Jiang, and Zhiyong Luo. 2016b. Combination of convolutional and recurrent neural network for sentiment analysis of short texts. In *Proceedings of the 26th International Conference on Computational Linguistics*. pages 2428–2437.

Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhutdinov, Richard S Zemel, and Yoshua Bengio. 2015a. Show, attend and tell: Neural image caption generation with visual attention. *arXiv preprint arXiv:1502.03044* 2(3):5.

Kun Xu, Yansong Feng, Songfang Huang, and Dongyan Zhao. 2015b. Semantic relation classification via convolutional neural networks with simple negative sampling. *arXiv preprint arXiv:1506.07650* .

Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy. 2016. Hierarchical attention networks for document classification. In *Proceedings of NAACL-HLT*. pages 1480–1489.

Daojian Zeng, Kang Liu, Siwei Lai, Guangyou Zhou, Jun Zhao, et al. 2014. Relation classification via convolutional deep neural network. In *COLING*. pages 2335–2344.

Peng Zhou, Wei Shi, Jun Tian, Zhenyu Qi, Bingchen Li, Hongwei Hao, and Bo Xu. 2016a. Attention-based bidirectional long short-term memory networks for relation classification. In *ACL*.

Peng Zhou, Wei Shi, Jun Tian, Zhenyu Qi, Bingchen Li, Hongwei Hao, and Bo Xu. 2016b. Attention-based bidirectional long short-term memory networks for relation classification. In *The 54th Annual Meeting of the Association for Computational Linguistics*. page 207.