A Deep Learning Approach for Sentence Classification of Scientific Abstracts

Sérgio Gonçalves[1], Paulo Cortez[1], and Sérgio Moro[2]

[1] ALGORITMI Centre, Dep. Information Systems, University of Minho, Portugal
a72886@alunos.uminho.pt, pcortez@dsi.uminho.pt
[2] Instituto Universitário de Lisboa (ISCTE-IUL), ISTAR-IUL, Lisboa, Portugal
sergio.moro@iscte-iul.pt

Abstract. The classification of abstract sentences is a valuable tool to support scientific database querying, to summarize relevant literature works and to assist in the writing of new abstracts. This study proposes a novel deep learning approach based on a convolutional layer and a bi-directional gated recurrent unit to classify sentences of abstracts. The proposed neural network was tested on a sample of 20 thousand abstracts from the biomedical domain. Competitive results were achieved, with weight-averaged precision, recall and F1-score values around 91%, which are higher when compared to a state-of-the-art neural network.

Keywords: Bi-directional Gated Recurrent Unit, Sentence Classification, Scientific Articles, Text Mining, Deep Learning

1 Introduction

In the last decades, there has been a rise in the number of scholarly publications [14]. For instance, around 114 million of English scholarly documents were accessible on the Web in 2014 [9]. Such volume makes it difficult to quickly select relevant scientific documents. Scientific abstracts summarize the most important elements of a paper and thus those are valuable sources for filtering the most relevant papers during a literature review process [1].

The classification of scientific abstracts is a particular instance of the sequential classification task, considering there is a typical order in the classes (e.g., the 'Objective' label tends to appear after the 'Background'). This classification transforms unstructured text into a more information manageable structure [6]. This is acknowledged by the Emerald publisher, which requires all submissions to include a structured abstract [4]. In effect, the automatic classification of abstract sentences presents several advantages. It is a valuable tool for general scientific database querying (e.g., using Web of Science, Scopus). Also, it can assist in manual [11] or text mining [15] systematic literature review processes, as well as other bibliometric analyses. Moreover, it can help in the writing of new paper abstracts [13].

In this study, we present a deep learning neural network architecture for the sequential classification of abstract sentences. The architecture uses a word embedding layer, a convolutional layer, a bi-directional Gated Recurrent Unit (GNU) and a final concatenation layer. The proposed deep learning model is compared with a recently proposed bi-directional Long Short-Term Memory (LSTM) based model [6], showing an interesting performance on a large 20K abstract corpus that assumes five sentence classes: 'Background', 'Objectives', 'Methods', 'Results' and 'Conclusions'. This paper is organized as follows. First, the related work is introduced in Section 2. Next, the abstract corpus and methods are described in Section 3. Then, the experimental results are presented and analyzed in Section 4. Finally, the main conclusions are discussed in Section 5.

2 Related Work

As pointed out in [6], most sequential sentence classification methods are based on 'shallow' methods (e.g., naive Bayes, Support Vector Machines (SVM)) that require a manual feature engineering based on lexical (e.g., bag of words, n-grams), semantic (e.g, synonyms), structural (e.g., part-of-speech tags) or sequential (e.g., sentence position) information. The advantage of using deep learning is that the neural networks do not require such manual design of features. Also, deep learning often achieves competitive results in text classification [8].

Regarding abstract sentence classification, this topic has been scarcely researched when compared to other text classification tasks (e.g., sentiment analysis). The main reason for this reduced attention is the restricted availability of publicly datasets. In 2010 [2], the manual engineering approach was used to set nine features (e.g., bi-grams) and train five classifiers (e.g., SVM) that were combined to classify four main elements of medical abstracts. In 2013 [13], a private corpus with 4550 abstracts from different scientific fields was collected from ScienceDirect. The abstract sentences were manually labeled into four categories: 'Background', 'Goal', 'Methods' and 'Results'. The authors also used the conventional manual feature design approach (e.g., n-grams) and a transductive SVM. More recently, in 2017 [5], a large abstract corpus was made publicly available. Using this dataset, a deep learning model, based on one bi-directional LSTM, was proposed for a five class sentence prediction, outperforming four other approaches (e.g., n-gram logistic regression, multilayer perceptron) [6].

In this paper, we propose a different deep learning architecture, mainly composed by a convolutional layer and a bi-directional GRU layer to classify the sentences from abstracts, which uses word embeddings instead of character embeddings. By taking into consideration the position of the sentences, as well as encoding contextual information on the vector of each sentence, we expect that the proposed architecture can potentially achieve better results when compared with the study by [6].

3 Materials and Methods

3.1 Abstract Corpus

We adopted the abstract corpus first analyzed by [5], which sets the baseline for comparison purposes. The corpus includes open access papers from the PubMed biomedical database and related with Randomized Controlled Trials (RCT). The sentences were classified by the authors of the articles into the five standardized labels.

The full corpus has a total of 200K abstracts. A smaller subset, with 20K most recent abstracts, was also made available for a faster experimentation of sequential sentence classification methods. Considering the 20K subset was used in the work of [6], we also adopt the same dataset, to facilitate the experimental comparison. Table 1 presents the class frequencies and train, validation and test split sizes. This is an unbalanced dataset, with most sentences being related with 'Methods' or 'Results' (around 30%).

Table 1. Class distribution and train, validation and test sizes (PubMed 20K corpus).

	Background	**Objective**	**Methods**	**Results**	**Conclusions**
#sentences	28,797	18,548	79,214	77,507	36,321
percentage	12.0%	7.7%	33.0%	32.2%	15.1%

	Train	**Validation**	**Test**
#abstracts	15.0K	2.5K	2.5K
#sentences	180.0K	30.0K	30.0 K

3.2 Neural Networks Models

In the last years, there has been remarkable developments in deep learning [8]. Architectures such as Convolutional Neural Network (CNN), LSTM and GRU have obtained competitive results in several competitions (e.g., computer vision, signal and natural language processing).

The CNN is a network mainly composed by convolutional layers. The purpose of the convolutional layers is to extract features that preserve relevant information from the inputs [12]. To obtain the features, a convolutional layer receives a matrix as input, to which a matrix with a set of weights, known as a filter, is applied using a sliding window approach and, at each of the sliding window steps, a convolution is calculated, resulting in a feature. The size of the filter is a relevant hyperparameter.

Although CNNs have been widely used in computer vision, they can also be used in in sentence classification [10]. The use of convolutional layers enables the extraction of features from a window of words, which is useful because word embeddings alone are not able to detect specific nuances, such as double negation, which is important for sentiment classification. The width of the filter, represented by h, determines the length of the n-grams. The number of filters is also a hyperparameter, making it possible to use multiple filters with varying lengths [10]. The filters are initialized with random weights and, during the training of the network, the weights are learned for the specific task of the network, through backpropagation. Since each filter produces its own feature map, there is a need to reduce the dimensionality caused by using multiple filters. A sentence can be encoded as a single vector by applying a max pooling layer after the convolutional layer, which takes the maximum value for each position, from all the feature maps, keeping only the most important features.

Recurrent Neural Networks (RNN) are relevant for sequential data, such as the words that appear in a sentence. Consider the words $(x_1, ..., x_t)$ from a given sentence (sequence of words). The hidden state s_t of the word x_t depends on the hidden state s_{t-1}, which in turn is the hidden state of the word x_{t-1} and, for this reason, the order in which words appear over the sequence also influence the various hidden states of the RNN.

The LSTM network is a particular RNN that uses an internal memory to keep information between distant time steps to model long-term dependencies of the sequence. It uses two gating mechanisms, update gate and forget gate, which controls what information should be updated into the memory, and what information should be erased from the memory, respectively. The GRU [3] was recently introduced and it can be used as an alternative to the LSTM model. The GRU uses a reset and update gate, which are able to control how much information should be kept from previous time steps. Both GRU and LSTM are solutions that help mitigate the vanishing gradient problem of conventional RNNs.

A deep learning model was used in [6] for abstract sentence classification. The model uses character embeddings that are then concatenated with word embeddings and used as input for a bi-directional LSTM layer, which outputs a sentence vector based on those hybrid embeddings. The sentence vector is used to predict the probabilities of the labels for that sentence. The authors also use a sequence optimization layer, which has the objective of optimizing the classification of a sequence of sentences, exploiting existing dependencies between labels.

3.3 Proposed Architecture

The proposed word embedding, convolutional and bi-directional GRU (Word-BiGRU) architecture is shown in Figure 1. We assume that each abstract has i sentences $(S_1, ..., S_i)$ and each individual sentence has n words $(x_1^1, ..., x_n^i)$, where x_n^i is the n^{th} word from the i^{th} sentence. The various words from the sentences are mapped to their respective word embeddings, and those embedding are used to create a sentence matrix $E \in R^{m \times d}$, where d equals to the dimensionality of the embeddings. We use word embeddings pre-trained on English Wikipedia, provided by Glove (with $d = 200$) [16].

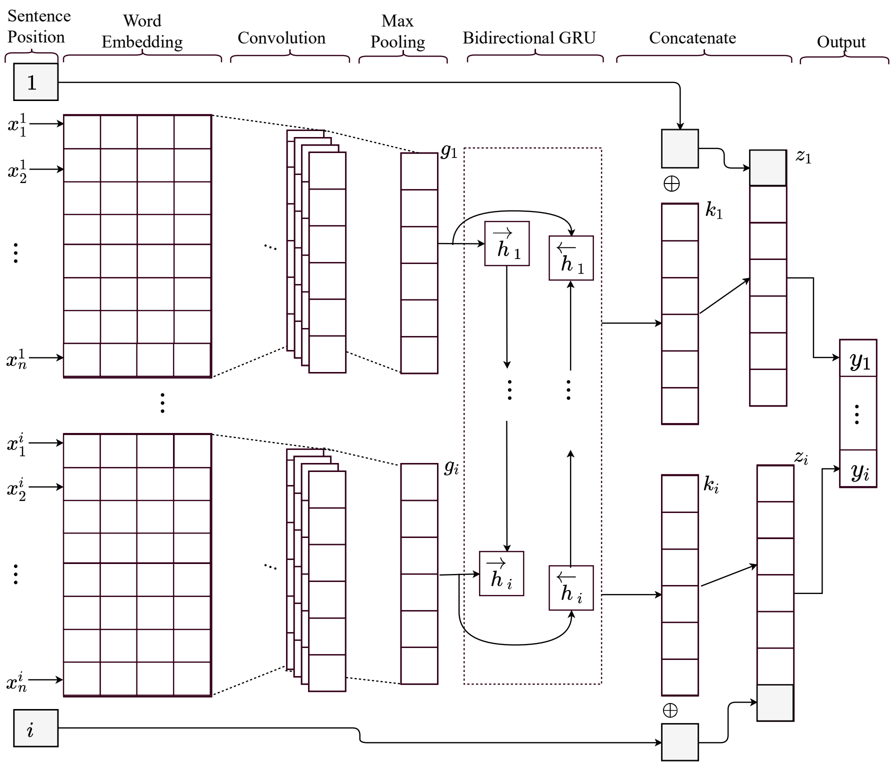

Fig. 1. Schematic of the proposed Word-BiGRU deep learning architecture.

Then, a convolutional layer is used with a sliding window approach that extracts the most important features from the sentences. Let $E \in R^{m\times d}$ denote the sentence matrix, $w \in R^{h\times d}$ a filter, and $E[i : j]$ the sub-matrix from row i to j. The single feature o_i is obtained using:

$$o_i = w * E[i : i + h - 1] \; . \tag{1}$$

In this study, we use a filter with a size of $h = 5$. To add nonlinearity to the output, an activation function applied to every single feature. For the feature o_i, it is obtained by:

$$c_i = f(o_i + b); \tag{2}$$

where f is the activation function and b is the bias. We use ReLU as the activation function in our model because it tends to present a faster convergence [7].

Next, we take the various features maps obtained from the convolutional layer, and feed them into a max pooling layer to encode the most important features extracted by the convolutional layer into a single vector representation that can be used by the next layers. Let $g_1, ..., g_i$ denote several vectors, each one encoding a particular sentence of the abstract. The vectors are then fed to bi-directional GRU layer, where the hidden states for each time step are calculated. We will use $\odot$ to denote the Hadamard Product, while using W and U to denote weight matrices of the GRU layer. Let h_{i-1} be the hidden state of the previous sentence from the same abstract, the candidate hidden state $\tilde{h}_i$ for the current sentence is given by:

$$\tilde{h}_i = \tanh(W_h g_i + U_h(r_i \odot h_{i-1}) + b_h) \ . \tag{3}$$

The reset gate $r_i \in [0, 1]$ has the purpose of controlling how much information of the past hidden state, h_{t-1} will be kept. Let σ be the sigmoid activation function. The reset gate r_i is calculated by:

$$r_i = \sigma(W_r g_i + U_r h_{i-1} + b_r) \ . \tag{4}$$

To control how much new information will be stored in the hidden state, an update gate $z_i \in [0, 1]$ is used, given by:

$$z_i = \sigma(W_z g_i + U_z h_{i-1} + b_z) \ . \tag{5}$$

The hidden state h_i, which is the hidden state of the sentence i, is obtained by:

$$h_i = z_i \odot \tilde{h}_i + (1 - z_i) \odot h_{i-1} \ . \tag{6}$$

Since we use a bi-directional GRU layer, there is a forward pass and a backward pass. The hidden states resulting from the forward pass are:

$$(\overrightarrow{h_1}, ..., \overrightarrow{h_i}) \ . \tag{7}$$

where h_i is the hidden state of the i^{th} sentence of the abstract. Similarly, the hidden states resulting from the backward pass are:

$$(\overleftarrow{h_1}, ..., \overleftarrow{h_i}) \ . \tag{8}$$

By using a bi-directional GRU, we want to capture contextual information about each sentence of the abstract, by taking into consideration the sentences that appear before and after it. For the i^{th} sentence of the abstract, the individual vector k_i, which encodes the sentence with contextual information captured using the bi-directional GRU layer, is obtained by concatenating ($\oplus$ operator) the forward and backward hidden states:

$$k_i = [\overrightarrow{h_i} \oplus \overleftarrow{h_i}] \ . \tag{9}$$

Each encoded sentence k_i is then concatenated with an integer value indicating the position of that sentence in the abstract, resulting in z_i:

$$z_i = [k_i \oplus i] \ . \tag{10}$$

Finally, a softmax layer is used, such that the outputs can be interpreted as class probabilities.

3.4 Evaluation

Classification accuracy is often measured using a confusion matrix, which maps predicted versus desired labels. From this matrix, several metrics can be computed, such as: [17]: Precision, Recall, F1-score. For a class c, these metrics are obtained using:

$$\begin{array}{ll} \text{Precision}_c & = \frac{TP_c}{TP_c+FP_c} \\ \text{Recall}_c & = \frac{TP_c}{TP_c+FN_c} \\ \text{F1-score}_c & = 2 \times \frac{\text{Precision}_c \times \text{Recall}_c}{\text{Precision}_c + \text{Recall}_c} \;. \end{array} \tag{11}$$

where TP_c, FP_c, FN_c denote the number of true positives, false positives and false negatives for class c.

To combine all five class results into a single measure, we adopt two aggregation methods: macro-averaging and weight-averaging. The macro-averaging computes first the metric (e.g., Precision using Equation 11) for each class and then averages the overall result. The weight-averaging is computed in a similar way except that each class metric is weighted proportionally to its prevalence in the data. In [6] only the weight-averaging method was used.

For comparison purposes, we adopt the same train, validation and test sets used in [6] (Table 1). When fitting the deep learning architecture, we adjusted different combinations of its main hyperparameters, namely: the number of filters (128 or 256) in the convolutional layer and the number of units ($\in \{25, 50, 75, 100\}$) in the bi-directional GRU Layer. The validation set was used to select the best configuration, when monitoring the macro-averaging Precision metric. In the test set comparison, we computed all classification metrics.

4 Results

The deep learning models were trained on the p2.xlarge instance from Amazon Elastic Compute Cloud, which has an Intel Xeon E5-2686 v4 2.30 GHz, Nvidia Tesla K80 and 61 GB of RAM. The experiments were implemented in Python using the Keras and Scikit packages. The selected hyperparameters (using validation metrics) are shown in Table 2. Figure 2 shows the normalized confusion matrix of the proposed model. The matrix confirms that a very good classification was achieved, in particular for the 'Methods', 'Conclusions' and 'Results' labels and that correspond to the most frequent classes.

The proposed Word-BiGRU deep learning architecture is compared with two other approaches: a similar model that does not include the bi-directional GRU layer (CNN model), and with the results provided in [6] (Char-BiLSTM). Table 3 shows the test results for each class. Word-BiGRU shows competitive results when compared with Char-BiLSTM. Specifically, it achieves the best Precision and Recall values for three classes and the best F1-scores for all classes. Furthermore, the deep learning model provides the highest classification improvement (11.3 percentage points) for the least frequent class ('Objectives'). The averaged class results are detailed in Table 4. Word-BiGRU provides better results in all metrics when compared with the other models. The improvement ranges:

Table 2. Selected hyperparameters of the proposed model.

Common parameters			
Embedding dimension	200		
Maximum Length	100		
Dropout	0.35		
Loss Function	Categorical Cross-entropy		
Optimizer	Adam		
Convolutional Layer		**Bi-directional GRU Layer**	
Activation function(s)	ReLU	Activation function	Tanh
Filter size	5	Number of units	50
Number of Filters	128		

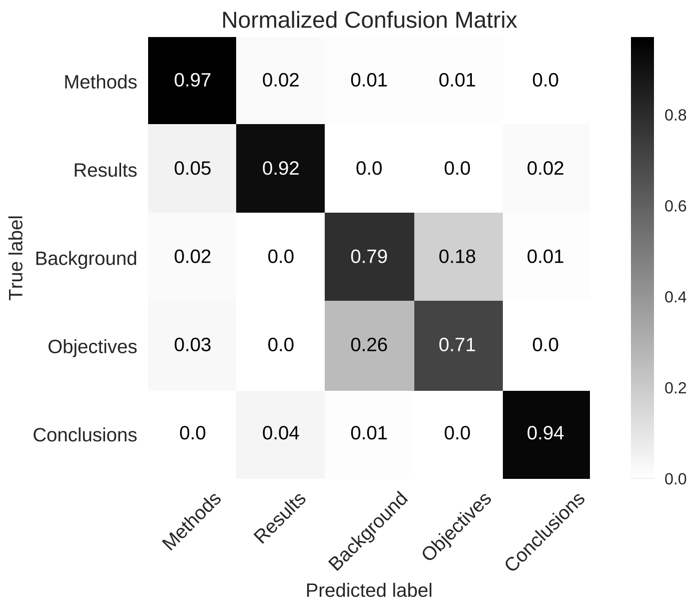

Fig. 2. Normalized confusion matrix.

from 6.0 to 9.1 percentage points, when compared with CNN, confirming the value of the bi-directional GRU layer; and from 0.3 to 3.0 percentage points when compared with Char-BiLSTM. Finally, we note that the Word-BiGRU model requires more computation than the simpler CNN model. On average, the proposed architecture requires 880 seconds per epoch while CNN requires 182 seconds.

Table 3. Test results for each class (in %, best values in **bold**).

		Background	Objective	Methods	Results	Conclusions
Precision	Word-BiGRU	**79.7**	70.5	93.3	**95.9**	**94.2**
	Char-BiLSTM [6]	71.8	**78.2**	**93.7**	94.8	93.5
Recall	Word-BiGRU	78.7	**71.4**	**96.7**	92.3	**94.5**
	Char-BiLSTM [6]	**88.2**	48.1	96.2	**93.1**	92.9
F1-Score	Word-BiGRU	**79.2**	**70.9**	**95.0**	**94.1**	**94.3**
	Char-BiLSTM [6]	79.1	59.6	94.9	93.9	93.2

Table 4. Averaged test results (in %, best values in **bold**).

Metric	**Averaged**	**Char-BiLSTM [6]**	**CNN**	**Word-BiGRU**
Precision	Macro-Averaged	86.4	80.7	**86.7**
	Weight-Averaged	90.1	83.6	**90.9**
Recall	Macro-Averaged	83.7	77.6	**86.7**
	Weight-Averaged	89.9	83.5	**90.8**
F1-score	Macro-Averaged	85.0	78.5	**86.7**
	Weight-Averaged	90.0	83.5	**90.8**

5 Conclusions

Abstract sentence classification is a key element to assist in scientific database querying, performing literature reviews and to support the writing of new abstracts. In this paper, we present a novel deep learning architecture for abstract sentence classification. The proposed Word-BiGRU architecture assumes word embeddings, a convolutional layer and a bi-directional Gated Recurrent Unit (GRU). Using a large sentence corpus, related with 20 thousand abstracts from the biomedical domain, we have obtaining high quality classification performances, with weight-average Precision, Recall and F1-score values around 91%. These results compare favourably against a state-of-the-art bi-directional Long Short-Term Memory (LSTM) model. In future work, we wish to enlarge the experimentation of the proposed deep learning architecture to classify abstract corpus from other scientific domains and also to other sequential tasks.

Acknowledgements

This work was supported by COMPETE: POCI-01-0145-FEDER-007043 and FCT Fundação para a Ciência e Tecnologia within the Project Scope: UID/CEC/-00319/2013.

References

1. Atanassova, I., Bertin, M., Larivière, V.: On the composition of scientific abstracts. Journal of Documentation **72**(4), 636–647 (2016)
2. Boudin, F., Nie, J.Y., Bartlett, J.C., Grad, R., Pluye, P., Dawes, M.: Combining classifiers for robust pico element detection. BMC medical informatics and decision making **10**(1), 29 (2010)
3. Cho, K., van Merrienboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., Bengio, Y.: Learning phrase representations using rnn encoder–decoder for statistical machine translation. In: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP). pp. 1724–1734. Association for Computational Linguistics, Doha, Qatar (October 2014)
4. Cornuel, E.: A vision for business schools, vol. 24. Emerald Group Publishing (2005)
5. Dernoncourt, F., Lee, J.Y.: Pubmed 200k rct: a dataset for sequential sentence classification in medical abstracts. In: Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 2). vol. 2, pp. 308–313 (2017)
6. Dernoncourt, F., Lee, J.Y., Szolovits, P.: Neural networks for joint sentence classification in medical paper abstracts. In: Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics. vol. 2, pp. 694–700 (2017)
7. Glorot, X., Bordes, A., Bengio, Y.: Deep sparse rectifier neural networks. In: Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics. pp. 315–323 (2011)
8. Goodfellow, I., Bengio, Y., Courville, A., Bengio, Y.: Deep learning, vol. 1. MIT press Cambridge (2016)
9. Khabsa, M., Giles, C.L.: The number of scholarly documents on the public web. PloS one **9**(5), e93949 (2014)
10. Kim, Y.: Convolutional neural networks for sentence classification. In: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP). pp. 1746–1751 (2014)
11. Kitchenham, B., Brereton, P.: A systematic review of systematic review process research in software engineering. Information and software technology **55**(12), 2049–2075 (2013)
12. LeCun, Y., Kavukcuoglu, K., Farabet, C.: Convolutional networks and applications in vision. Proceedings of 2010 IEEE International Symposium on Circuits and Systems pp. 253–256 (2010)
13. Liu, Y., Wu, F., Liu, M., Liu, B.: Abstract sentence classification for scientific papers based on transductive svm. Computer and Information Science **6**(4), 125 (2013)
14. Michalska-Smith, M.J., Allesina, S.: And, not or: Quality, quantity in scientific publishing. PloS one **12**(6), e0178074 (2017)
15. Moro, S., Cortez, P., Rita, P.: Business intelligence in banking: A literature analysis from 2002 to 2013 using text mining and latent dirichlet allocation. Expert Systems with Applications **42**(3), 1314–1324 (2015)
16. Pennington, J., Socher, R., Manning, C.: Glove: Global vectors for word representation. In: Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP). pp. 1532–1543 (2014)
17. Witten, I., Frank, E., Hall, M., Pal, C.: Data Mining: Practical Machine Learning Tools and Techniques. Morgan Kaufmann, San Franscico, USA, San Francisco, CA, 4th edn. (2017)