LONG TEXT ANALYSIS USING SLICED RECURRENT NEURAL NETWORKS WITH BREAKING POINT INFORMATION ENRICHMENT

Bo Li[1,2,⋆], *Zehua Cheng*[1,⋆], *Zhenghua Xu*[1,†], *Wei Ye*[2,†], *Thomas Lukasiewicz*[3], *Shikun Zhang*[2]

[1]State Key Laboratory of Reliability and Intelligence of Electrical Equipment,
Hebei University of Technology, Tianjin, China
[2]National Engineering Research Center for Software Engineering, Peking University, Beijing, China
[3]Department of Computer Science, University of Oxford, Oxford, United Kingdom
zhenghua.xu@hebut.edu.cn, wye@pku.edu.cn

ABSTRACT

Sliced recurrent neural networks (SRNNs) are the state-of-the-art efficient solution for long text analysis tasks; however, their slicing operations inevitably result in long-term dependency loss in lower-level networks and thus limit their accuracy. Therefore, we propose a breaking point information enrichment mechanism to strengthen dependencies between sliced subsequences without hindering parallelization. Then, the resulting BPIE-SRNN model is further extended to a bidirectional model, BPIE-BiSRNN, to utilize the dependency information in not only the previous but also the following contexts. Experiments on four large public real-world datasets demonstrate that the BPIE-SRNN and BPIE-BiSRNN models always achieve a much better accuracy than SRNNs and BiSRNNs, while maintaining a superior training efficiency.

Index Terms— Long Text Analysis, RNN, SRNN

1. INTRODUCTION

Recurrent neural networks (RNNs) have been witnessed to achieve a superior performance in many natural language processing (NLP) tasks, such as machine translation [1], question answering [2, 3], and natural language inference [4]. The success of applying RNNs in NLP tasks mainly comes from their capability to pass the information in one step of the networks to the next, making it possible to process each word based on the textual information of previous words. However, RNNs suffer from the vanishing (and exploding) gradient problem, which makes it difficult to discover long-distance dependencies between words [5]. Long short-term memory networks (LSTMs) [6, 7] and gated recurrent units (GRUs) [8] are proposed to remedy this problem using a gate mechanism.

However, traditional RNNs, LSTMs, and GRUs all suffer from a training efficiency problem in long text analysis tasks: with increasing text length, the training time of these models also increases dramatically, which greatly limits their application in real-world scenarios. One existing solution is to skip some information, simulating a human's daily reading habits, such as fixation and saccades [9, 10]; however, this will result in the loss of textual information. Another solution is to introduce convolutional components into the recurrent structure [11, 12, 13]; but the convolutional components make it difficult to fully utilize the order information of sequential data. Thus, their performances are usually unsatisfactory.

Sliced recurrent neural networks (*SRNNs*) [14] are the state-of-the-art efficient solution for long text analysis tasks, where input sequences are sliced into multiple subsequences, and GRUs process these subsequences in a parallelized way. Although SRNNs can still discover dependencies between subsequences in higher levels of their networks, the slicing operations still inevitably weaken the models' capability in capturing the long-term dependencies in lower levels.

Consequently, if more dependencies between subsequences can be maintained in lower levels without hindering parallelization, it will not only make lower-level networks capable to get more dependency information, but also help higher-level networks to get more accurate long-term dependencies, which will consequently enhance the models' performance. One natural choice is to use the last few words of the previous subsequence to calculate a more accurate starting state for a subsequence, instead of using a fixed or random starting state. Furthermore, adding previous words to a given subsequence will increase the probability of containing sentence separators in this subsequence; since separators contain dependency information between sentences, they will further improve the representation modeling of given subsequences.

Motivated by this, we first propose a breaking point information enrichment (*BPIE*) mechanism, which strengthens the semantic dependencies between sliced subsequences in SRNNs by adding to each subsequence the last few words of the previous subsequence. The resulting model is called *BPIE-SRNN*. In addition, since people can better understand a given sentence by considering not only the information in previous sentences but also that in following sentences, we further propose to extend the BPIE-SRNN model to a bidirecitonal model, called *BPIE-BiSRNN*, to use the dependency information in not only the previous but also the following contexts. In BPIE-BiSRNNs, subsequences are processed

⋆Co-first authors, contributed to this work equally.
†Corresponding authors.

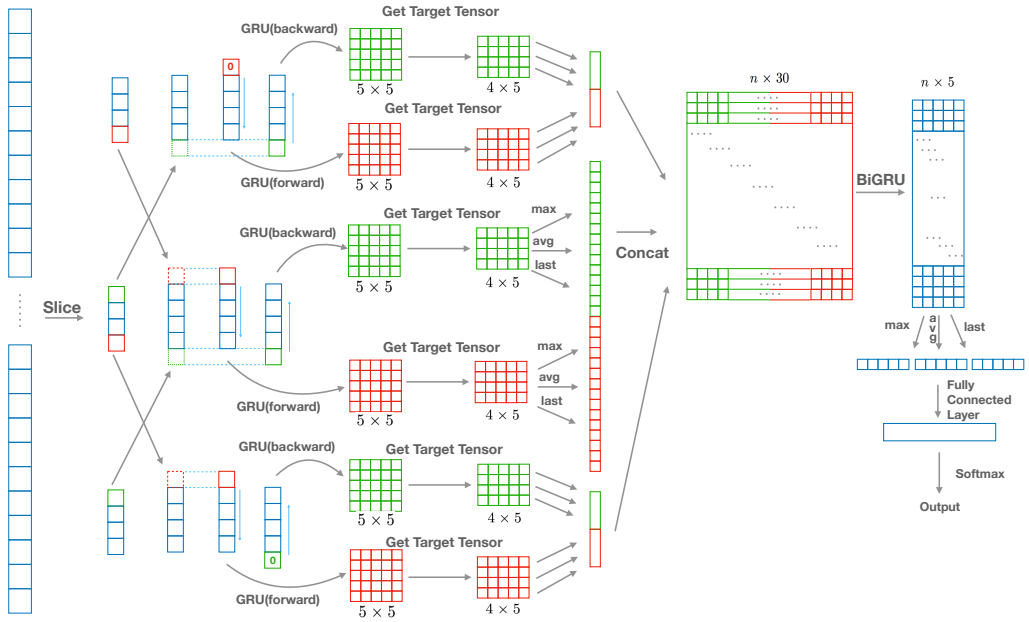

Fig. 1. Procedure of BPIE-BiSRNNs with subsequence length $z = 4$, hyperparameter $m = 1$, and hidden states of GRU $h = 5$

both forward and backward in GRUs, and the last (resp., first) few words of the previous (resp., next) subsequence are added to given subsequences in forward (resp., backward) processes for information enrichment. Extensive experimental studies on four large public real-world datasets demonstrate the superior performance of BPIE-SRNNs and BPIE-BiSRNNs on long text analysis tasks: BPIE-SRNNs and BPIE-BiSRNNs always achieve a much better accuracy than SRNNs and BiSRNNs, while maintaining a superior training efficiency.

2. BPIE-BISRNN

In this section, we introduce the details of the BPIE-BiSRNN model; BPIE-SRNNs are a simplified version of BPIE-BiSRNNs, where all backward operations are removed. The overall working of BPIE-BiSRNNs is shown as pseudocode in Algorithm 1 and also depicted in Fig. 1.

In BPIE-BiSRNNs, given a long text sequence S (after padding) consisting of L words (i.e., $S = [x_0, x_1, \ldots, x_{L-1}]$), we first slice S into n subsequences $S = [s_0, s_1, \ldots, s_{n-1}]$ with an equal length $z = L/n$ (line 1 in Algorithm 1).

Then, for each subsequence s_k, we add the last m words of the previous subsequence s_{k-1} to the front of s_k, resulting in an information-enriched subsequence for the forward processes, denoted $s_{k\_forward}$; similarly, $s_{k\_backward}$ is obtained for the backward processes by adding the first m words of the next subsequence s_{k+1} (lines 4–5). Formally,

$$s_{k\_forward} = Concat([s_{k-1}[-m:], s_k]), \quad (1)$$

$$s_{k\_backward} = Concat([s_k, s_{k+1}[:m]]), \quad (2)$$

where zero-padded subsequences s_{-1} and s_n are created for the boundary cases, when $k = 0$ and $k = n - 1$ (line 2), and m is a suitable hyperparameter with value less than z.

Algorithm 1: BPIE-BiSRNN(S)

Input: long text sequence S
Output: classification probability $Softmax(p)$

1 $s_0, s_1, \ldots, s_{n-1} \leftarrow slice(S)$
2 $s_{-1} \leftarrow s_n \leftarrow [0] * m$
3 **for** k *in range* 0 *to* $n-1$ **do**
4 $\quad s_{k\_forward} \leftarrow Concat([s_{k-1}[-m:], s_k])$
5 $\quad s_{k\_backward} \leftarrow Concat([s_k, s_{k+1}[:m]])$
6 $\quad \mathbf{O}_{k\_forward} \leftarrow \overrightarrow{GRU}\ (s_{k\_forward})$
7 $\quad \mathbf{O}_{k\_backward} \leftarrow \overleftarrow{GRU}\ (s_{k\_backward})$
8 $\quad \mathbf{O'}_{k\_forward} \leftarrow \mathbf{O}_{k\_forward}[m:]$
9 $\quad \mathbf{O'}_{k\_backward} \leftarrow \mathbf{O}_{k\_backward}[:-m]$
10 $\quad r_{k\_forward} \leftarrow Concat([max(\mathbf{O'}_{k\_forward}), avg(\mathbf{O'}_{k\_forward}), last(\mathbf{O'}_{k\_forward})])$
11 $\quad r_{k\_backward} \leftarrow Concat([max(\mathbf{O'}_{k\_backward}), avg(\mathbf{O'}_{k\_backward}), last(\mathbf{O'}_{k\_backward})])$
12 $\quad r_k \leftarrow Concat([r_{k\_forward}, r_{k\_backward}])$
13 $\mathbf{R} \leftarrow [r_0, r_1, \ldots, r_{n-1}]$
14 $\mathbf{R'} \leftarrow BiGRU(\mathbf{R})$
15 $r' \leftarrow Concat([max(\mathbf{R'}), avg(\mathbf{R'}), last(\mathbf{R'})])$
16 $p \leftarrow Fully\_Connected(r')$
17 return $Softmax(p)$

$s_{k\_forward}$ (resp., $s_{k\_backward}$) is then used as the input of a forward (resp., backward) GRU to obtain a forward (resp., backward) output tensor $\mathbf{O}_{k\_forward}$ (resp., $\mathbf{O}_{k\_backward}$) (lines 6–7). Formally,

$$\mathbf{O}_{k\_forward} = \overrightarrow{GRU}\ (s_{k\_forward}), \quad (3)$$

$$\mathbf{O}_{k\_backward} = \overleftarrow{GRU}\ (s_{k\_backward}), \quad (4)$$

where the size of $\mathbf{O}_{k\_forward}$ and $\mathbf{O}_{k\_backward}$ is $(z+m) \times h$, with h being the number of hidden states in the GRU.

Furthermore, to avoid duplicate computations in the following steps, we drop the outputs originating from the m enriched words, resulting in a target output tensor $\mathbf{O'}_{k\_forward}$ and $\mathbf{O'}_{k\_backward}$ with a size $z \times h$ (lines 8–9). This operation is called *get target tensor* and formally written as

$$\mathbf{O'}_{k\_forward} = \mathbf{O}_{k\_forward}[m:], \tag{5}$$

$$\mathbf{O'}_{k\_backward} = \mathbf{O}_{k\_backward}[:-m]. \tag{6}$$

Then, for each pair of $\mathbf{O'}_{k\_forward}$ and $\mathbf{O'}_{k\_backward}$, we conduct maximum, average, and last tensor operations on their first dimension and concatenate the resulting $1 \times h$ vectors to obtain a 1st-layer feature representation of s_k, denoted r_k, whose size is $1 \times (6 * h)$ (lines 10–12). Furthermore, we merge the representations of all subsequences to a full 1st-layer feature representation of the input long text sequence S, denoted $\mathbf{R}$, whose size is $n \times (6 * h)$ (line 13).

$\mathbf{R}$ is further passed to a BiGRU to get a full 2nd-layer output of S, denoted $\mathbf{R'}$, and then maximum, average, and last tensor operations are conducted again on $\mathbf{R'}$ to get the full 2nd-layer feature representation r' (lines 14–15). Finally, we send r' into a fully-connected layer and apply softmax operation to get results (lines 16–17). Please note that BPIE-BiSRNNs are set to three layers in this work for clarity; however, by stacking multiple GRUs between the first two layers, BPIE-BiSRNNs can be easily extended to deeper networks.

3. EXPERIMENTS

To show the strength of our proposed BPIE-SRNN model in handling long text analysis tasks and offering a superior performance in both accuracy and efficiency, standard RNNs with GRUs [8] as the recurrent unit and the state-of-the-art solution, SRNNs [14], are selected as the baselines. In addition, the performance of the bidirectional models of GRUs, SRNNs, and BPIE-SRNNs (denoted *BiGRUs*, *BiSRNNs*, and *BPIE-BiSRNNs*, respectively) is further evaluated to show the advantage of applying the bidirectional structure to further enhance the classification accuracies of models. For a fair comparison, the experiments are performed on the same public large online real-world datasets used in [14]. The statistic information of the datasets is shown in Table 1, and the detailed descriptions are as follows:

Rating sentiment datasets: Three subsets of Yelp reviews released by the Yelp Dataset Challenge[1] in the years 2013, 2014, and 2015 are extracted by Yu an Liu [14] to construct three Yelp rating sentiment datasets, denoted *Yelp 2013*, *Yelp 2014*, and *Yelp 2015*, respectively. These datasets contain $468,608$ documents, $670,440$ documents, and $897,835$ documents, respectively, and each document is associated with a rating sentiment label, ranging from 1 to 5 stars (the more, the better). Then, 10% of the documents in each dataset are randomly selected using the same random seed as in [14] to form a test set, and the rest is used as the training set.

Polarity sentiment dataset: Yelp reviews are also used by Zhang et al. [15] to construct a polarity sentiment dataset, denoted *Yelp_P*, where each document is associated with a polarity sentiment label (either positive or negative). This dataset is also used in [14], and the same division of training and test sets as in [15] is adopted.

3.1. Training Details and Code Release

All models are implemented using Keras [16], and trained on a GPU server with an NVIDIA GTX 1080Ti GPU. The training details here are all similar to that in [14], except for the length of subsequences. Specifically, in [14], the input long text sequence is sliced into several subsequences with 32 words in each of them; however, in BPIE-SRNNs, we add to each subsequence the last 5 words of the previous subsequence to enrich the semantic dependency information between sliced subsequences, making the length of each subsequence become 37; thus, to keep a fair comparison in training efficiency, the length of subsequences in SRNNs is also set to 37. Similarly, the length of each subsequence in BiSRNNs is also the same as that in BPIE-BiSRNNs.

To facilitate future research and clarify the details, easy-to-run codes have been released online.[2] Some important training details are as follows: (i) the pre-trained Glove embeddings in [17] are used to initialize the word embeddings; (ii) the batch size is 512; (iii) the recurrent unit is set as GRU, and all its hidden states and fully-connected layers have 64 dimensions; (iv) all models are learned using the Adam optimizer [18] with learning rate $\alpha = 0.001$, the first momentum $\beta_1 = 0.9$, and the second momentum $\beta_2 = 0.999$; and (v) the dropout rate is 0.2.

3.2. Main Results

Table 2 depicts the performance of the proposed models, BPIE-SRNNs and BPIE-BiSRNNs, and their corresponding baselines, GRUs, SRNNs, BiGRUs, and BiSRNNs, on four sentiment datasets, *Yelp 2013*, *Yelp 2014*, *Yelp 2015*, and *Yelp_P*, in terms of classification accuracy and training efficiency. Despite some slight differences, due to different subsequence lengths (37 vs. 32) and different kinds of GPUs used (1080Ti vs. 1080), the performance results of GRUs and SRNNs in Table 2 are highly consistent with the results reported in [14], showing the correctness of our experiments.

In Table 2, BPIE-SRNNs (resp., BPIE-BiSRNNs) significantly outperform their corresponding baselines, GRUs and SRNNs (resp., BiGRUs and BiSRNNs), on all four datasets in terms of classification accuracy. This demonstrates that BPIE-SRNNs and BPIE-BiSRNNs successfully overcome the long-term dependency loss problem in the lower levels of SRNNs and BiSRNNs by applying breaking point information enrichment to strengthen the semantic dependencies between sliced subsequences. Furthermore, due to breaking point information enrichment, the training time costs of BPIE-SRNNs and BPIE-BiSRNNs on all datasets are inevitably higher than those of SRNNs and BiSRNNs; however, the increases are always less than 20%, which is minor and acceptable in practice, and the time costs are still much less

[1] https://www.yelp.com/dataset/challenge

[2] Code release link: https://github.com/limberc/BPIE-BiSRNN

Table 1. Dataset information

Dataset	Classes	Documents	Training Samples	Test Samples	Max Words	Average Words
Yelp 2013	5	468,608	421,748	46,860	2,185	130
Yelp 2014	5	670,440	603,396	67,044	2,379	116
Yelp 2015	5	897,835	808,052	89,783	2,372	108
Yelp_P	2	598,000	560,000	38,000	1059	136

Table 2. Classification accuracy and training efficiency of all models on four sentiment datasets

Dataset	Model	Accuracy (%)	Time/Epoch (sec)	Dataset	Model	Accuracy (%)	Time/Epoch (sec)
Yelp 2013	GRUs	66.02	730	*Yelp 2014*	GRUs	70.29	1,253
	SRNNs	66.58	102		SRNNs	70.46	151
	BPIE-SRNNs	**67.30**	121		BPIE-SRNNs	**70.90**	183
	BiGRUs	66.68	1,472		BiGRUs	70.52	2,492
	BiSRNNs	66.72	213		BiSRNNs	70.68	312
	BPIE-BiSRNNs	**68.04**	251		BPIE-BiSRNNs	**71.60**	372
Yelp 2015	GRUs	73.24	1,609	*Yelp_P*	GRUs	96.05	763
	SRNNs	72.93	218		SRNNs	96.07	136
	BPIE-SRNNs	**73.54**	247		BPIE-SRNNs	**96.36**	170
	BiGRUs	73.36	3,176		BiGRUs	96.25	1,532
	BiSRNNs	73.34	440		BiSRNNs	96.16	281
	BPIE-BiSRNNs	**74.12**	501		BPIE-BiSRNNs	**96.84**	344

than those of GRUs and BiGRUs. These results show that BPIE-SRNNs and BPIE-BiSRNNs can achieve a much better accuracy than SRNNs and BiSRNNs in long text analysis tasks, while maintaining a superior training efficiency.

Table 2 also exhibits that the bidirectional models, BiGRUs, BiSRNNs, and BPIE-BiSRNNs, always outperform the single directional models, GRUs, SRNNs, and BPIE-SRNNs, respectively, in terms of classification accuracy on all datasets, and BPIE-BiSRNNs achieve the best accuracy among all six models. This finding proves that besides the dependency information in the previous context, the information in the following context is also beneficial for long text analysis tasks; thus, when the accuracy instead of the time cost is the first priority, it is reasonable to extend BPIE-SRNNs to BPIE-BiSRNNs, to further enhance the accuracy.

Notice also that the accuracy improvements of BPIE-BiSRNNs relative to BPIE-SRNNs are higher than those of BiSRNNs relative to SRNNs on all datasets. It proves that, by using breaking point information enrichment, BPIE-BiSRNNs successfully overcome the long-term dependency loss problem in lower levels of BiSRNNs, making it capable to obtain more backward dependency information in lower levels and to get more accurate long-term backward dependencies in higher levels. This thus results in higher accuracy improvements in BPIE-BiSRNNs compared to BiSRNNs.

3.3. Effect of Increasing Text Length

We also investigate the effect of increasing the text length on the accuracy improvement of BPIE-BiSRNNs (resp., BPIE-SRNNs) relative to BiSRNNs (resp., SRNNs). The experiments are on five subsets of text sequences selected from *Yelp 2013*, where the sequence lengths are 100-200, 300-400, 500-600, 700-800, and 900-1000 words, respectively.

As shown in Table 3, the accuracy improvement of BPIE-BiSRNNs relative to BiSRNNs greatly rises up with the increase of the text length: when the text length is 100-200 words, the accuracy improvement is only 1.40%; it then rises up to 3.66% and 3.75%, when the text length increases to 300-400 and 500-600 words, respectively; the accuracy improvement finally reaches up to 8.40% and 8.06%, when the text lengths are 700-800 and 900-1000 words, respectively. In addition, the comparative results of BPIE-SRNNs and SRNNs are similar and exhibit the same finding; so they are omitted due to space limit. This observation further demonstrates the capability of BPIE on discovering long-term semantic dependencies, and proves that BPIE-BiSRNNs and BPIE-SRNNs are better choices than BiSRNNs and SRNNs in handling long text analysis tasks.

Table 3. Accuracy on different text lengths

Acc (%) Length / model	100−200	300−400	500−600	700−800	900−1000
BiSRNNs	66.87	66.01	68.28	57.25	53.23
BPIE-BiSRNNs	68.27	69.67	72.03	65.65	61.29
improvement	1.40	3.66	3.75	8.40	8.06

4. CONCLUSION

We have proposed a breaking point information enrichment mechanism to strengthen dependencies between sliced subsequences without hindering parallelization. The resulting BPIE-SRNN model has also been extended to a bidirectional model, called BPIE-BiSRNN. In experiments, BPIE-SRNNs and BPIE-BiSRNNs always achieve a much better accuracy than SRNNs and BiSRNNs, while having a superior training efficiency. In future works, we will apply BPIE-SRNNs and BPIE-BiSRNNs to improve more NLP applications. Moreover, we will also investigate more techniques, e.g., distance masking [19], to further enhance the models' performances.

5. REFERENCES

[1] Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, et al., "Google's neural machine translation system: Bridging the gap between human and machine translation," *arXiv preprint arXiv:1609.08144*, 2016.

[2] Shuohang Wang and Jing Jiang, "Machine comprehension using match-lstm and answer pointer," in *Proceedings of the International Conference on Learning Representations (ICLR)*, 2017, pp. 1–15.

[3] Kenton Lee, Shimi Salant, Tom Kwiatkowski, Ankur Parikh, Dipanjan Das, and Jonathan Berant, "Learning recurrent span representations for extractive question answering," *arXiv preprint arXiv:1611.01436*, 2016.

[4] Bill MacCartney and Christopher D Manning, "Modeling semantic containment and exclusion in natural language inference," in *Proceedings of the International Conference on Computational Linguistics (COLING)*, 2008, pp. 521–528.

[5] Yoshua Bengio, Patrice Simard, and Paolo Frasconi, "Learning long-term dependencies with gradient descent is difficult," *IEEE transactions on neural networks*, vol. 5, no. 2, pp. 157–166, 1994.

[6] Sepp Hochreiter and Jürgen Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[7] Klaus Greff, Rupesh K Srivastava, Jan Koutník, Bas R Steunebrink, and Jürgen Schmidhuber, "Lstm: A search space odyssey," *IEEE transactions on neural networks and learning systems*, vol. 28, no. 10, pp. 2222–2232, 2017.

[8] Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio, "Learning phrase representations using rnn encoder–decoder for statistical machine translation," in *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2014, pp. 1724–1734.

[9] Adams Wei Yu, Hongrae Lee, and Quoc Le, "Learning to skim text," in *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, 2017, vol. 1, pp. 1880–1890.

[10] Shaonan Wang, Jiajun Zhang, and Chengqing Zong, "Learning sentence representation with guidance of human attention," in *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, 2017, pp. 4137–4143.

[11] Siwei Lai, Liheng Xu, Kang Liu, and Jun Zhao, "Recurrent convolutional neural networks for text classification.," in *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, 2015, pp. 2267–2273.

[12] Pedro HO Pinheiro and Ronan Collobert, "Recurrent convolutional neural networks for scene labeling," in *Proceedings of the International Conference on Machine Learning (ICML)*, 2014, number EPFL-CONF-199822.

[13] Francisco Javier Ordóñez and Daniel Roggen, "Deep convolutional and lstm recurrent neural networks for multimodal wearable activity recognition," *Sensors*, vol. 16, no. 1, pp. 115, 2016.

[14] Zeping Yu and Gongshen Liu, "Sliced recurrent neural networks," in *Proceedings of the International Conference on Computational Linguistics (COLING)*, 2018, pp. 2953–2964.

[15] Xiang Zhang, Junbo Zhao, and Yann LeCun, "Character-level convolutional networks for text classification," in *Advances in neural information processing systems*, 2015, pp. 649–657.

[16] Chollet François and others., "Keras: The python deep learning library," https://github.com/keras-team/keras.

[17] Jeffrey Pennington, Richard Socher, and Christopher Manning, "Glove: Global vectors for word representation," in *Proceedings of the conference on empirical methods in natural language processing (EMNLP)*, 2014, pp. 1532–1543.

[18] Diederik P Kingma and Jimmy Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[19] Jinbae Im and Sungzoon Cho, "Distance-based self-attention network for natural language inference," *arXiv preprint arXiv:1712.02047*, 2017.