An Exploration of Dropout with RNNs for Natural Language Inference

Amit Gajbhiye[1], Sardar Jaf[1], Noura Al Moubayed[1], A. Stephen McGough[2], and Steven Bradley[1]

[1] Department of Computer Science, Durham University, Durham, UK
{amit.gajbhiye,s.p.bradley,sardar.jaf,noura.al-moubayed}@durham.ac.uk
[2] School of Computing, Newcastle University, Newcastle upon Tyne, UK
{stephen.mcgough}@newcastle.ac.uk

Abstract. Dropout is a crucial regularization technique for the Recurrent Neural Network (RNN) models of Natural Language Inference (NLI). However, dropout have not been evaluated for the effectiveness at different layers and dropout rates in NLI models. In this paper, we propose a RNN model for NLI and empirically evaluate the effect of dropout at different layers in the model. We also investigate the impact of varying dropout rates at these layers. Our empirical evaluation on two datasets of varied size - Stanford Natural Language Inference (SNLI) and SciTail, suggest that dropout at each feed-forward connection severely affect the accuracy of the model at increasing dropout rates. We show that embedding layer regularization is efficient for SNLI whereas recurrent layer regularization is for SciTail. Our model achieved an accuracy 86.14% on the SNLI dataset and 77.05% on SciTail.

Keywords: Neural Networks, Dropout, Natural Language Inference.

1 Introduction

Natural Language Understanding (NLU) is a process to enable computers to understand the semantics of natural language text. The inherent complexities and ambiguities in natural language text makes NLU challenging for computers. The task of Natural Language Inference (NLI) is a fundamental step towards NLU [14]. NLI involves logically inferring a given hypothesis sentence from a given premise sentence.

Recently, the release of a large public dataset the Stanford Natural Language Inference (SNLI) [2] has made it feasible to train complex neural network models for NLI. Recurrent Neural Networks (RNNs), particularly bidirectional LSTMs (BiLSTMs) have shown state-of-the-art results on the SNLI dataset [9]. However, RNNs are susceptible to overfitting − the case when a neural network learns the exact patterns present in the training data but fails to generalize to unseen data [21]. In NLI models, regularization techniques such as early stopping [4], L2 regularization and dropout [20] are used to prevent overfitting.

For RNNs, dropout is an effective regularization technique [21]. The idea of dropout is to randomly omit computing units in a neural network during training but keeping all of them for testing. Dropout consists of element-wise multiplication of the neural network layer activations with a zero-one mask (r_j) during training. Each element of the zero-one mask is drawn independently from $r_j \sim \text{Bernoulli}(p)$, where p is the probability with which the units are retained in the network. During testing, activations of the layer are multiplied by p [19].

Dropout is a crucial regularization technique for NLI [9][20]. However, the location of dropout varies considerably between NLI models and is based on trail-and-error experiments with different locations in the network. To the best of our knowledge no prior work has been performed to evaluate the effectiveness of dropout location and rates.

In this paper, we study the effect of applying dropout at different locations in an RNN for NLI. We also investigate the effect of varying the dropout rate. Our results suggest that applying dropout for every feed forward connection, especially at higher dropout rates degrades the performance of RNN. Our best model achieves an accuracy of 86.14% on the SNLI dataset and an accuracy of 77.05% on SciTail dataset.

To the best of our knowledge this research is the first exploratory analysis of dropout for NLI. The main contributions of this paper are as follows - (1) A comparative analysis of different locations of dropout in a RNN model for NLI and recommendations for the application of dropout. (2) An investigation into the effectiveness of dropout in preventing overfitting for the proposed RNN NLI. (3) An analysis of dropout rates on the performance of the RNN NLI and on dropout locations; and (4) A RNN model based on BiLSTMs for NLI.

The layout of the paper is as follows. In Section 2 we describe the related work. In Section 3 we discuss the details of the proposed RNN based NLI model. Experiments and the results are presented in Section 4. Finally, we present our conclusions in Section 5.

2 Related Work

The RNN NLI models follow a general architecture. It consists of - (1) an embedding layer that take input the word embeddings of premise and hypothesis (2) a sentence encoding layer, generally an RNN that generates representations of the input (3) an aggregation layer that combine the representations and (4) a classifier layer that classify the relationship between premise and hypothesis.

Different NLI models apply dropout at different layers in the general NLI architecture. NLI models proposed by Ghaeini et al. [9] and Tay et al. [20] use dropout for each feed-forward layer in the network whereas others have applied dropout only to the final classifier layer [13]. Bowman et al. [2] apply dropout only to the input and output of sentence encoder.The models proposed by Bowman et al. [3] and Choi et al. [7] applied dropout to the output of embedding layer and to the input and output of classifier layer. Chen et al. [4] and Cheng et al. [6] use dropout but do not sufficiently elaborate on the location.

Dropout rates are also crucial for the NLI models [15]. Even the models which apply dropout at the same locations vary dropout rates from [0.1 - 0.5].

Dropout research by Pachitariu and Sahani [16], Pham et.al [18] and Zaremba et al. [21] on RNNs, have established that recurrent connection dropout should not be applied as it affects the long term dependencies in sequential data.

Bluche et al. [1] studied dropout at different positions with respect to the LSTM units in the network proposed in [18]. The results show that significant performance difference is observed when dropout is applied to distinct places. They concluded that applying dropout only after recurrent layers as applied by Pham et al. [18] or between every feed-forward layer, done by Zaremba et al. [21], does not always yield good results. Cheng et al. [5], investigated the effect of applying dropout in LSTMs. They randomly switch off the outputs of various gates of LSTM, achieving an optimal word error rate when dropout is applied to output, forget and input gates of the LSTM.

Our work differs in the size of datasets used to evaluate the RNN model. Evaluations in previous research were conducted on datasets with fewer samples. We evaluate the RNN model on a relatively large, SNLI dataset (570,000 data sample) as well as on a smaller SciTail dataset (27,000 data samples). Furthermore, previous studies concentrate only on the location of dropout in the network with fixed dropout rate.We further investigate the effect of varying dropout rates. We focus on the application of widely used conventional dropout [19] to non-recurrent connection in RNNs.

3 Recurrent Neural Network Model for NLI Task

The proposed RNN NLI model follow the general architecture of NLI models and is depicted in Fig.1. The model enhances the intra-attention model [13] with soft-attention mechanism [11]. The embedding layer takes as input word embeddings in the sequence of length L. The recurrent layer with BiLSTM units encodes

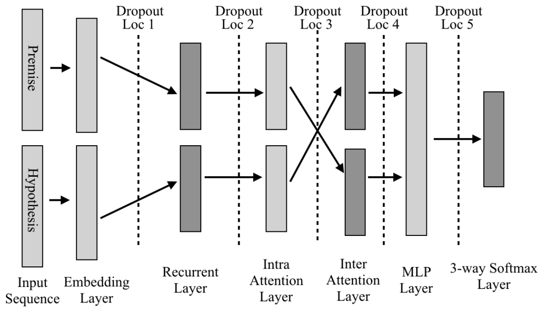

Fig. 1. The Recurrent Neural Network Model with possible Dropout Locations

the sentence. Next, the intra-attention layer generates the attention weighted sentence representation following the Equations (1) – (3)

$$M = \tanh\left(W^y Y + W^h R_{avg} \otimes e_L\right) \tag{1}$$

$$\alpha = softmax\left(w^T M\right) \tag{2}$$

$$R = Y\alpha^T \tag{3}$$

where, W^y, W^h are trained projection matrices, w^T is the transpose of trained parameter vector w, Y is the matrix of hidden output vectors of the BiLSTM layer, R_{avg} is obtained from the average pooling of Y, $e_L \in \mathbb{R}^L$ is a vector of 1s, α is a vector of attention weights and R is the attention weighted sequence representation. The attention weighted sequence representation is generated for premise and hypothesis and is denoted as R_p and R_h. The attention weighted representation gives more importance to the words which are important to the semantics of the sequence and also captures its global context.

The interaction between R_p and R_h is performed by inter-attention layer, following the Equations (4) – (6).

$$I_v = R_p^T R_h \tag{4}$$

$$\tilde{R_p} = softmax(I_v) R_h \tag{5}$$

$$\tilde{R_h} = softmax(I_v) R_p \tag{6}$$

where, I_v is the interaction vector. $\tilde{R_p}$ contains the words which are relevant based on the content of sequence R_h. Similarly, $\tilde{R_h}$ contains words which are important with respect to the content of sequence R_p. The final sequence encoding is obtained from the element-wise multiplication of intra-attention weighted representation and inter-attention weighted representation as follows:

$$F_p = \tilde{R_p} \odot R_p \tag{7}$$

$$F_h = \tilde{R_h} \odot R_h \tag{8}$$

To classify the relationship between premise and hypothesis a relation vector is formed from the encoding of premise and hypothesis generated in Equation (7) and (8), as follows:

$$\mathrm{v}_{p,avg} = \mathrm{averagepooling}(F_p), \mathrm{v}_{p,max} = \mathrm{maxpooling}(F_p) \tag{9}$$

$$\mathrm{v}_{h,avg} = \mathrm{averagepooling}(F_h), \mathrm{v}_{h,max} = \mathrm{maxpooling}(F_h) \tag{10}$$

$$F_{relation} = [\mathrm{v}_{p,avg}; \mathrm{v}_{p,max}; \mathrm{v}_{h,avg}; \mathrm{v}_{h,max}] \tag{11}$$

where v is a vector of length L. The relation vector ($F_{relation}$) is fed to the MLP layer. The three-way softmax layer outputs the probability for each class of NLI.

4 Experiments and Results

4.1 Experimental Setup

We report the empirical evaluations on two varied size datasets − SNLI [2] and SciTail [10]. The standard train, validation and test splits are used. The validation set is used for hyper-parameter tuning. The non-regularized model is our baseline model. The parameters for the baseline model are selected separately for SNLI and SciTail dataset by a grid search from the combination of L2 regularization $[1e-4, 1e-5, 1e-6]$, batch size $[64, 256, 512]$ and learning rate $[0.001, 0.0003, 0.0004]$. The Adam [12] method is used as an optimizer.The first momentum is set to 0.9 and the second to 0.999. The word embeddings are initialized with pre-trained 300-D Glove 840B vectors [17]. Extensive experiments with different combination of dropout locations and hidden units were conducted however we show only the best results for reasons of space.

4.2 Dropout at Different Layers for NLI Model

Table 1 presents the models with different combinations of layers to the output of which dropout are applied in our model depicted in Fig. 1. Table 2. shows the results for the models in Table 1. Each model is evaluated with dropout rates ranging from 0.1 to 0.5 with a granularity of 0.1.

Dropout at Individual Layers We first apply dropout at each layer including the embedding layer. Although the embedding layer is the largest layer it is often not regularized for many language applications [8]. However, we observe the benefit of regularizing it. For SNLI, the highest accuracy is achieved when the embedding layer is regularized (Model 2, DR 0.4).

For SciTail, the highest accuracy is attained when the recurrent layer is regularized (Model 3, DR 0.1). The dropout injected noise at lower layers prevents

Table 1. Models with corresponding layers to the outputs of which dropout is applied.

Model	Layer
Model 1	No Dropout (Baseline)
Model 2	Embedding
Model 3	Recurrent
Model 4	Embedding and Recurrent
Model 5	Recurrent and Intra-Attention
Model 6	Inter-Attention and MLP
Model 7	Recurrent, Inter-Attention and MLP
Model 8	Embedding, Inter-Attention and MLP
Model 9	Embedding, Recurrent, Inter-Attention and MLP
Model 10	Recurrent, Intra-Attention, Inter-Attention and MLP
Model 11	Embedding, Intra-Attention, Inter-Attention and MLP
Model 12	Embedding, Recurrent, Intra-Attention, Inter-Attention and MLP
Model 13	Embedding, Recurrent, Inter-Attention and MLP

Table 2. Model accuracy with varying dropout rates for SNLI and SciTail datasets. Bold numbers shows the highest accuracy for the model within the dropout range.

Models	**Dataset**	**Dropout Rate (DR)**				
		0.1	0.2	0.3	0.4	0.5
Model 1	SNLI	84.45				
	SciTail	74.18				
Model 2	SNLI	84.56	84.59	84.42	**86.14**	84.85
	SciTail	**75.45**	75.12	74.22	73.10	74.08
Model 3	SNLI	84.12	**84.21**	83.76	81.04	79.63
	SciTail	**76.15**	75.78	73.50	73.19	75.26
Model 4	SNLI	83.83	**85.22**	84.34	80.82	79.92
	SciTail	74.65	**76.08**	74.22	74.46	73.19
Model 5	SNLI	**84.72**	83.43	72.89	70.49	62.13
	SciTail	**75.87**	75.13	75.26	73.71	72.25
Model 6	SNLI	84.17	**84.32**	83.71	82.79	81.68
	SciTail	73.85	**75.68**	75.26	73.95	73.28
Model 7	SNLI	**84.33**	82.97	82.00	81.15	79.25
	SciTail	73.75	**75.02**	74.37	73.37	73.42
Model 8	SNLI	84.67	**85.82**	84.60	84.14	83.94
	SciTail	73.80	73.52	69.29	**75.82**	73.89
Model 9	SNLI	**84.44**	83.05	82.09	81.64	79.62
	SciTail	75.68	**76.11**	75.96	70.84	74.55
Model 10	SNLI	**84.45**	80.95	75.31	70.81	69.34
	SciTail	73.30	**75.21**	74.98	74.65	71.59
Model 11	SNLI	**84.31**	82.43	78.94	74.93	70.54
	SciTail	**75.63**	73.47	74.93	74.93	70.32
Model 12	SNLI	**84.32**	82.60	73.36	71.53	66.67
	SciTail	73.47	**75.63**	74.74	73.42	74.40

higher fully connected layers from overfitting. We further experimented regularizing higher fully connected layers (Intra-Attention, Inter-Attention, MLP) individually, however no significant performance gains observed[3].

Dropout at Multiple Layers We next explore the effect of applying dropout at multiple layers. For SNLI and SciTail, the models achieve higher performance when dropout is applied to embedding, as done for individual layers, and recurrent layer (Model 4, DR 0.2).

It is interesting to note that regularizing the recurrent layer helps SciTail (Model 7, DR 0.2) whereas regularizing the embedding layer helps SNLI (Model 8, DR 0.2). A possible explanation to this is that for the smaller SciTail dataset the model can not afford to lose information in the input, whereas for the larger SNLI dataset the model has a chance to learn even with the loss of information in input. Also, the results from the models 7 and 8 suggests that applying dropout

[3] results are not shown in Table 2

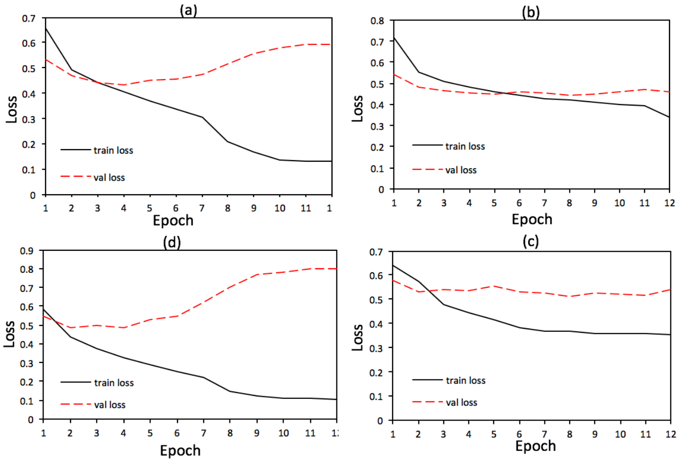

Fig. 2. Convergence Curves: (a) Baseline Model for SNLI, (b) Best Model for SNLI, (c) 100 Unit Model for SciTail, (d) 300 Unit Model for SciTail

at a single lower layer (Embedding or Recurrent; depending on the amount of training data) and to the inputs and outputs of MLP layer improves performance.

We can infer from models 9, 10, 11, 12 that applying dropout to each feed forward connection helps preventing the model overfit for SciTail (DR 0.1 and 0.2). However, for both the datasets with different dropout locations the performance of the model decreases as the dropout rate increases[4].

4.3 The Effectiveness of Dropout for Overfitting

We study the efficacy of dropout on overfitting. The main results are shown in Fig. 2. For SNLI, Fig. 2 (a) - (b), shows the convergence curves for the baseline model and the model achieving the highest accuracy (Model 2, DR 0.4). The convergence curve show that dropout is very effective in preventing overfitting. However, for the smaller SciTail dataset when regularizing multiple layers we observe that the highest accuracy achieving model (Model 9, DP 0.2), overfits significantly (Fig. 2(d)). This overfitting due to the large model size. With limited training data of SciTail, our model with higher number of hidden units learns the relationship between the premise and the hypothesis most accurately (Fig. 2(d)). However, these relationships are not representative of the validation set data and thus the model does not generalize well. When we reduced the model size (50, 100 and 200 hidden units) we achieved the best accuracy for SciTail at 100 hidden units (Table 3). The convergence curve (Fig. 2(c)) shows that dropout effectively prevents overfitting in the model with 100 hidden units in comparison to 300 units. Furthermore, for SciTail dataset, the model with 100

[4] please refer Section 4.4 for details

Table 3. Accuracy for 100 unit model for SciTail dataset

Models	Dataset	Dropout Rate (DR)				
		0.1	0.2	0.3	0.4	0.5
Model 13	SciTail	76.72	76.25	72.58	**77.05**	74.22

units achieved higher accuracies for almost all the experiments when compared to models with 50, 200 and 300 hidden units.

The results of this experiment suggest that given the high learning capacity of RNNs an appropriate model size selection according to the amount of training data is essential. Dropout may independently be insufficient to prevent overfitting in such scenarios.

4.4 Dropout Rate Effect on Accuracy and Dropout Location

We next investigate the effect of varying dropout rates on the accuracy of the models and on various dropout locations. Fig 3. illustrates varying dropout rates and the corresponding test accuracy for SNLI. We observe some distinct trends from the plot. First, the dropout rate and location does not affect the accuracy of the models 2 and 8 over the baseline. Second, in the dropout range [0.2 - 0.5], the dropout locations affect the accuracy of the models significantly. Increasing the dropout rate from 0.2 to 0.5 the accuracy of the models 5 and 12 decreases significantly by 21.3% and 15.9% respectively. For most of the models (3, 4, 6, 7, 9 and 10) the dropout rate of 0.5 decreases accuracy.

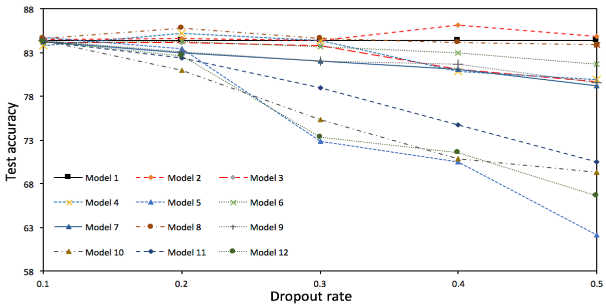

Fig. 3. Plot showing the variation of accuracy across the dropout range for SNLI.

From the experiments on SciTail dataset (Fig. 4), we can observe that the dropout rate and its location do not have significant affect on most of the models, with the exception of model 8 (which shows erratic performance). Finally, for almost all the experiments a large dropout rate (0.5) decreases the accuracy of the model. The dropout rate of 0.5 works for a wide rang of neural networks and tasks [19]. However, our results show that this is not desirable for RNN

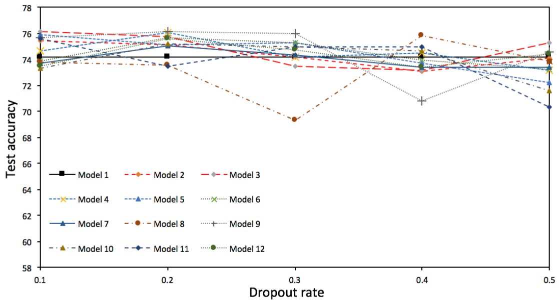

Fig. 4. Plot showing the variation of accuracy across the dropout range for SciTail.

models of NLI. Based on our evaluations a dropout rate ranging from $[0.2 - 0.4]$ is advised.

5 Conclusions

In this paper, we reported the outcome of experiments conducted to investigate the effect of applying dropout at different layers in a RNN model for the NLI. We have shown that different location of dropout significantly affects the performance of the NLI model. Through extensive exploration, for the correct dropout location in our model, we achieved the accuracies of 86.14% on SNLI and 77.05% on SciTail datasets.

Based on our empirical evaluations, the following is recommended for regularizing a RNN model for NLI task: (1) Embedding layer should be regularized for large datasets like SNLI. For smaller datasets such as SciTail regularizing recurrent layer is an efficient option. The dropout injected noise at these layers prevent the higher fully connected layers from overfitting. (2) When regularizing multiple layers, regularizing a lower layer (embedding or recurrent; depending on the amount of data) with the inputs and outputs of MLP layer should be considered. The performance of our model decreased when dropout is applied at each intermediate feed-forward connection. (3) When dropout is applied at multiple feed forward connections, it is almost always better to apply it at lower rate $-$ $[0.2 - 0.4]$. (4) Given the high learning capacity of RNNs, an appropriate model size selection according to the amount of training data is essential. Dropout may independently be insufficient to prevent overfitting in the scenarios otherwise. In future research, we aim to investigate the effect different dropout rates at distinct layers in the model.

References

1. Bluche, T., Kermorvant, C., Louradour, J.: Where to apply dropout in recurrent neural networks for handwriting recognition? In: Document Analysis and Recognition (ICDAR), 2015 13th International Conference on. pp. 681–685. IEEE (2015)

2. Bowman, S.R., Angeli, G., Potts, C., Manning, C.D.: A large annotated corpus for learning natural language inference. In: Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing. pp. 632–642. Association for Computational Linguistics (2015)
3. Bowman, S.R., Gauthier, J., Rastogi, A., Gupta, R., Manning, C.D., Potts, C.: A fast unified model for parsing and sentence understanding. In: Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). vol. 1, pp. 1466–1477 (2016)
4. Chen, Q., Zhu, X., Ling, Z.H., Inkpen, D.: Natural language inference with external knowledge. arXiv preprint arXiv:1711.04289 (2017)
5. Cheng, G., Peddinti, V., Povey, D., Manohar, V., Khudanpur, S., Yan, Y.: An exploration of dropout with lstms. In: Proceedings of Interspeech (2017)
6. Cheng, J., Dong, L., Lapata, M.: Long short-term memory-networks for machine reading. In: Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing. pp. 551–561 (2016)
7. Choi, J., Yoo, K.M., goo Lee, S.: Learning to compose task-specific tree structures. AAAI (2017)
8. Gal, Y., Ghahramani, Z.: A theoretically grounded application of dropout in recurrent neural networks. In: Advances in neural information processing systems. pp. 1019–1027 (2016)
9. Ghaeini, R., Hasan, S.A., Datla, V., Liu, J., Lee, K., Qadir, A., Ling, Y., Prakash, A., Fern, X.Z., Farri, O.: Dr-bilstm: Dependent reading bidirectional lstm for natural language inference. arXiv preprint arXiv:1802.05577 (2018)
10. Khot, T., Sabharwal, A., Clark, P.: Scitail: A textual entailment dataset from science question answering. In: Proceedings of AAAI (2018)
11. Kim, Y., Denton, C., Hoang, L., Rush, A.M.: Neural machine translation by jointly learning to align and translate. Proceedings of ICLR (2017)
12. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)
13. Liu, Y., Sun, C., Lin, L., Wang, X.: Learning natural language inference using bidirectional lstm model and inner-attention. CoRR abs/1605.09090 (2016)
14. MacCartney, B.: Natural language inference. Stanford University (2009)
15. Munkhdalai, T., Yu, H.: Neural tree indexers for text understanding. In: Proceedings of the conference. Association for Computational Linguistics. Meeting. vol. 1, p. 11. NIH Public Access (2017)
16. Pachitariu, M., Sahani, M.: Regularization and nonlinearities for neural language models: when are they needed? arXiv preprint arXiv:1301.5650 (2013)
17. 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)
18. Pham, V., Bluche, T., Kermorvant, C., Louradour, J.: Dropout improves recurrent neural networks for handwriting recognition. In: Frontiers in Handwriting Recognition (ICFHR), 2014 14th International Conference on. pp. 285–290. IEEE (2014)
19. Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.: Dropout: A simple way to prevent neural networks from overfitting. The Journal of Machine Learning Research 15(1), 1929–1958 (2014)
20. Tay, Y., Tuan, L.A., Hui, S.C.: A compare-propagate architecture with alignment factorization for natural language inference. arXiv preprint arXiv:1801.00102 (2017)
21. Zaremba, W., Sutskever, I., Vinyals, O.: Recurrent neural network regularization. arXiv preprint arXiv:1409.2329 (2014)