Shortcut-Stacked Sentence Encoders for Multi-Domain Inference

Yixin Nie and **Mohit Bansal**
UNC Chapel Hill
{yixin1, mbansal}@cs.unc.edu

Abstract

We present a simple sequential sentence encoder for multi-domain natural language inference. Our encoder is based on stacked bidirectional LSTM-RNNs with shortcut connections and fine-tuning of word embeddings. The overall supervised model uses the above encoder to encode two input sentences into two vectors, and then uses a classifier over the vector combination to label the relationship between these two sentences as that of entailment, contradiction, or neural. Our Shortcut-Stacked sentence encoders achieve strong improvements over existing encoders on matched and mismatched multi-domain natural language inference (top non-ensemble single-model result in the EMNLP RepEval 2017 Shared Task (Nangia et al., 2017)). Moreover, they achieve the new state-of-the-art encoding result on the original SNLI dataset (Bowman et al., 2015).

1 Introduction and Background

Natural language inference (NLI) or recognizing textual entailment (RTE) is a fundamental semantic task in the field of natural language processing. The problem is to determine whether a given hypothesis sentence can be logically inferred from a given premise sentence. Recently released datasets such as the Stanford Natural Language Inference Corpus (Bowman et al., 2015) (SNLI) and the Multi-Genre Natural Language Inference Corpus (Williams et al., 2017) (MultiNLI) have not only encouraged several end-to-end neural network approaches to NLI, but have also served as an evaluation resource for general representation learning of natural language.

Depending on whether a model will first encode a sentence into a fixed-length vector without any incorporating information from the other sentence, the several proposed models can be categorized into two groups: (1) encoding-based models (or sentence encoders), such as Tree-based CNN encoders (TBCNN) in Mou et al. (2015) or Stack-augmented Parser-Interpreter Neural Network (SPINN) in Bowman et al. (2016), and (2) joint, pairwise models that use cross-features between the two sentences to encode them, such as the Enhanced Sequential Inference Model (ESIM) in Chen et al. (2017) or the bilateral multi-perspective matching (BiMPM) model Wang et al. (2017). Moreover, common sentence encoders can again be classified into tree-based encoders such as SPINN in Bowman et al. (2016) which we mentioned before, or sequential encoders such as the biLSTM model by Bowman et al. (2015).

In this paper, we follow the former approach of encoding-based models, and propose a novel yet simple sequential sentence encoder for the MultiNLI problem. Our encoder does not require any syntactic information of the sentence. It also does not contain any attention or memory structure. It is basically a stacked (multi-layered) bidirectional LSTM-RNN with shortcut connections (feeding all previous layers' outputs and word embeddings to each layer) and word embedding fine-tuning. The overall supervised model uses these shortcut-stacked encoders to encode two input sentences into two vectors, and then we use a classifier over the vector combination to label the relationship between these two sentences as that of entailment, contradiction, or neural (similar to the classifier setup of Bowman et al. (2015) and Conneau et al. (2017)). Our simple shortcut-stacked encoders achieve strong improvements over existing encoders due to its multi-layered and shortcut-connected properties, on both matched and mis-

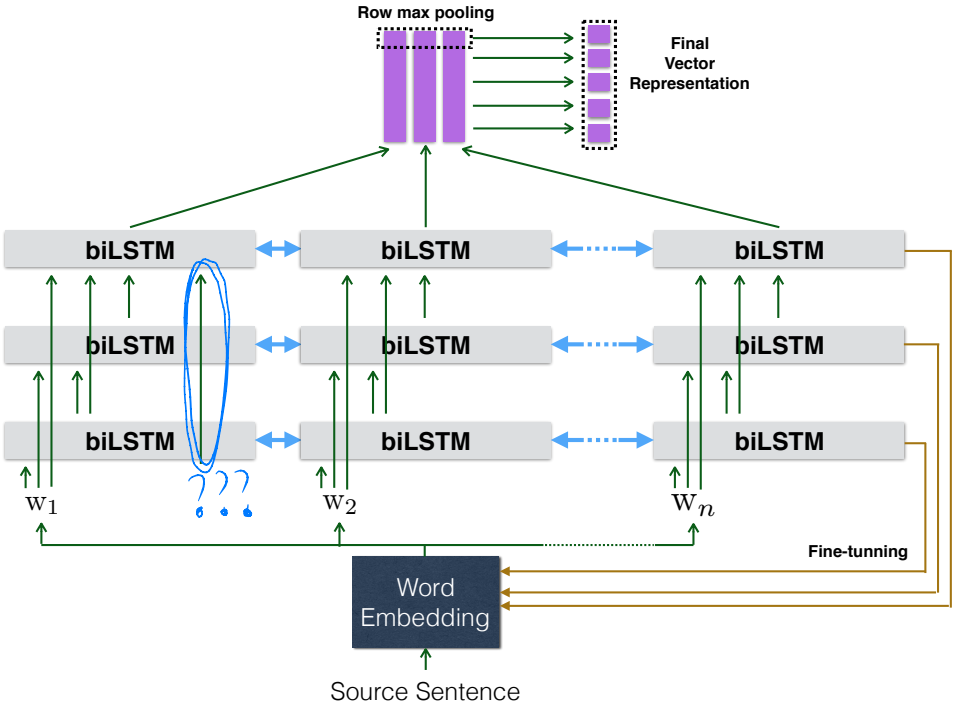

Figure 1: Our encoder's architecture: stacked biLSTM with shortcut connections and fine-tuning.

matched evaluation settings for multi-domain natural language inference, as well as on the original SNLI dataset. It is the top single-model (non-ensemble) result in the EMNLP RepEval 2017 Multi-NLI Shared Task (Nangia et al., 2017), and the new state-of-the-art for encoding-based results on the SNLI dataset (Bowman et al., 2015).

Github Code Link: https://github.com/easonnie/multiNLI_encoder

2 Model

Our model mainly consists of two separate components, a sentence encoder and an entailment classifier. The sentence encoder compresses each source sentence into a vector representation and the classifier makes a three-way classification based on the two vectors of the two source sentences. The model follows the 'encoding-based rule', i.e., the encoder will encode each source sentence into a fixed length vector without any information or function based on the other sentence (e.g., cross-attention or memory comparing the two sentences). In order to fully explore the generalization of the sentence encoder, the same encoder is applied to both the premise and the hypothesis with shared parameters projecting them into the same space. This setting follows the idea of Siamese Networks in Bromley et al. (1994). Figure 1 shows the overview of our encoding model (the standard classifier setup is not shown here; see Bowman et al. (2015) and Conneau et al. (2017) for that).

2.1 Sentence Encoder

Our sentence encoder is simply composed of multiple stacked bidirectional LSTM (biLSTM) layers with shortcut connections followed by a max pooling layer. Let bilstm^i represent the ith biLSTM layer, which is defined as:

$$h_t^i = \text{bilstm}^i(x_t^i, t), \forall t \in [1, 2, ..., n] \quad (1)$$

where h_t^i is the output of the ith biLSTM at time t over input sequence $(x_1^i, x_2^i, ..., x_n^i)$.

In a typical **stacked biLSTM** structure, the input of the next LSTM-RNN layer is simply the output sequence of the previous LSTM-RNN layer. In our settings, the input sequences for the ith biLSTM layer are the concatenated outputs of *all the previous layers*, plus the original word embedding sequence. This gives a **shortcut connection** style setup, related to the widely used idea of residual connections in CNNs for computer vision (He et al., 2016), highway networks for RNNs in speech processing (Zhang et al., 2016), and shortcut connections in hierarchical multitasking learning (Hashimoto et al., 2016); but in our case we feed in all the previous layers' output se-

quences as well as the word embedding sequence to every layer.

Let $W = (w_1, w_2, ..., w_n)$ represent words in the source sentence. We assume $w_i \in \mathbb{R}^d$ is a word embedding vector which are initialized using some pre-trained vector embeddings (and is then fine-tuned end-to-end via the NLI supervision). Then, the input of ith biLSTM layer at time t is defined as:

$$x_t^1 = w_t \quad (2)$$

$$x_t^i = [w_t, h_t^{i-1}, h_t^{i-2}, ...h_t^1] \quad (\text{for } i > 1) \quad (3)$$

where [] represents vector concatenation.

Then, assuming we have m layers of biLSTM, the final vector representation will be obtained by applying row-max-pool over the output of the last biLSTM layer, similar to Conneau et al. (2017). The final layer is defined as:

$$H^m = (h_1^m, h_2^m, ..., h_n^m) \quad (4)$$

$$v = max(H^m) \quad (5)$$

where $h_i^m, v \in \mathbb{R}^{2d_m}$, $H^m \in \mathbb{R}^{2d_m \times n}$, d_m is the dimension of the hidden state of the last forward and backward LSTM layers, and v is the final vector representation for the source sentence (which is later fed to the NLI classifier).

The closest encoder architecture to ours is that of Conneau et al. (2017), whose model consists of a single-layer biLSTM with a max-pooling layer, which we treat as our starting point. Our experiments (Section 4) demonstrate that our enhancements of the stacked-biRNN with shortcut connections provide significant gains on top of this baseline (for both SNLI and Multi-NLI).

2.2 Entailment Classifier

After we obtain the vector representation for the premise and hypothesis sentence, we apply three matching methods to the two vectors (i) concatenation (ii) element-wise distance and (iii) element-wise product for these two vectors and then concatenate these three match vectors (based on the heuristic matching presented in Mou et al. (2015)). Let v_p and v_h be the vector representations for premise and hypothesis, respectively. The matching vector is then defined as:

$$m = [v_p, v_h, |v_p - v_h|, v_p \otimes v_h] \quad (6)$$

At last, we feed this final concatenated result m into a MLP layer and use a softmax layer to make final classification.

Layers and Dimensions		Accuracy	
#layers	bilstm-dim	Matched	Mismatched
1	512	72.5	72.9
2	512 + 512	73.4	73.6
1	1024	72.9	72.9
2	512 + 1024	73.7	74.2
1	2048	73.0	73.5
2	512 + 2048	73.7	74.2
2	1024 + 2048	73.8	74.4
2	2048 + 2048	74.0	74.6
3	512 + 1024 + 2048	**74.2**	**74.7**

Table 1: Analysis of results for models with different # of biLSTM layers and their hidden state dimensions.

	Matched	Mismatched
without any shortcut connection	72.6	73.4
only word shortcut connection	74.2	74.6
full shortcut connection	**74.2**	**74.7**

Table 2: Ablation results with and without shortcut connections.

Word-Embedding	Matched	Mismatched
fixed	71.8	72.6
fine-tuned	**72.7**	**72.8**

Table 3: Ablation results with and without fine-tuning of word embeddings.

# of MLPs	Activation	Matched	Mismatched
1	tanh	73.7	74.1
2	tanh	73.5	73.6
1	relu	74.1	74.7
2	relu	**74.2**	**74.7**

Table 4: Ablation results for different MLP classifiers.

3 Experimental Setup

3.1 Datasets

As instructed in the RepEval Multi-NLI shared task, we use all of the training data in Multi-NLI combined with 15% randomly selected samples from the SNLI training set resampled at each epoch) as our final training set for all models; and we use both the cross-domain ('mismatched') and in-domain ('matched') Multi-NLI development sets for model selection. For the SNLI test results in Table 5, we train on only the SNLI training set (and we also verify that the tuning decisions hold true on the SNLI dev set).

3.2 Parameter Settings

We use cross-entropy loss as the training objective with Adam-based (Kingma and Ba, 2014) opti-

Model	Accuracy		
	SNLI	Multi-NLI Matched	Multi-NLI Mismatched
CBOW (Williams et al., 2017)	80.6	65.2	64.6
biLSTM Encoder (Williams et al., 2017)	81.5	67.5	67.1
300D Tree-CNN Encoder (Mou et al., 2015)	82.1	–	–
300D SPINN-PI Encoder (Bowman et al., 2016)	83.2	–	–
300D NSE Encoder (Munkhdalai and Yu, 2016)	84.6	–	–
biLSTM-Max Encoder (Conneau et al., 2017)	84.5	–	–
Our biLSTM-Max Encoder	85.2	71.7	71.2
Our Shortcut-Stacked Encoder	**86.1**	**74.6**	**73.6**

Table 5: Final Test Results on SNLI and Multi-NLI datasets.

mization with 32 batch size. The starting learning rate is 0.0002 with half decay every two epochs. The number of hidden units for MLP in classifier is 1600. Dropout layer is also applied on the output of each layer of MLP, with dropout rate set to 0.1. We used pre-trained 300D Glove 840B vectors (Pennington et al., 2014) to initialize the word embeddings. Tuning decisions for word embedding training strategy, the hyperparameters of dimension and number of layers for biLSTM, and the activation type and number of layers for MLP, are all explained in Section 4.

4 Results and Analysis

4.1 Ablation Analysis Results

We now investigate the effectiveness of each of the enhancement components in our overall model. These ablation results are shown in Tables 1, 2, 3 and 4, all based on the Multi-NLI development sets. Finally, Table 5 shows results for different encoders on SNLI and Multi-NLI test sets.

First, Table 1 shows the performance changes for different number of biLSTM layers and their varying dimension size. The dimension size of a biLSTM layer is referring to the dimension of the hidden state for both the forward and backward LSTM-RNNs. As shown, each added layer model improves the accuracy and we achieve a substantial improvement in accuracy (around 2%) on both matched and mismatched settings, compared to the single-layer biLSTM in Conneau et al. (2017). We only experimented with up to 3 layers with 512, 1024, 2048 dimensions each, so the model still has potential to improve the result further with a larger dimension and more layers.

Next, in Table 2, we show that the shortcut connections among the biLSTM layers is also an important contributor to accuracy improvement (around 1.5% on top of the full 3-layered stacked-RNN model). This demonstrates that simply stacking the biLSTM layers is not sufficient to handle a complex task like Multi-NLI and it is significantly better to have the higher layer connected to both the output and the original input of all the previous layers (note that Table 1 results are based on multi-layered models with shortcut connections).

Next, in Table 3, we show that fine-tuning the word embeddings also improves results, again for both the in-domain task and cross-domain tasks (the ablation results are based on a smaller model with a 128+256 2-layer biLSTM). Hence, all our models were trained with word embeddings being fine-tuned. The last ablation in Table 4 shows that a classifier with two layers of relu is preferable than other options. Thus, we use that setting for our strongest encoder.

4.2 Multi-NLI and SNLI Test Results

Finally, in Table 5, we report the test results for MNLI and SNLI. First for Multi-NLI, we improve substantially over the CBOW and biLSTM Encoder baselines reported in the dataset paper (Williams et al., 2017). We also show that our final shortcut-based stacked encoder achieves around 3% improvement as compared to the 1-layer biLSTM-Max Encoder in the second last row (using the exact same classifier and optimizer settings). Our shortcut-encoder was also the top singe-model (non-ensemble) result on the EMNLP RepEval Shared Task leaderboard.

Next, for SNLI, we compare our shortcut-stacked encoder with the current state-of-the-art encoders from the SNLI leaderboard (`https://nlp.stanford.edu/projects/snli/`). We also compare to the recent biLSTM-Max Encoder of Conneau et al. (2017), which served as our model's 1-layer starting point.[1] The results indicate that 'Our Shortcut-Stacked Encoder' sur-

[1]Note that the 'Our biLSTM-Max Encoder' results in the second-last row are obtained using our reimplementation of the Conneau et al. (2017) model; our version is 0.7% better, likely due to our classifier and optimizer settings.

passes all the previous state-of-the-art encoders, and achieves the new best encoding-based result on SNLI, suggesting the general effectiveness of simple shortcut-connected stacked layers in sentence encoders.

5 Conclusion

We explored various simple combinations and connections of biLSTM-RNN layered architectures and developed a Shortcut-Stacked Sentence Encoder for natural language inference. Our model is the top single result in the EMNLP RepEval 2017 Multi-NLI Shared Task, and it also surpasses the state-of-the-art encoders for the SNLI dataset. In future work, we are also evaluating the effectiveness of shortcut-stacked sentence encoders on several other semantic tasks.

6 Addendum: Shortcut vs. Residual

In later experiments, we found that a residual connection can achieve similar accuracies with fewer number of parameters, compared to a shortcut connection. Therefore, in order to reduce the model size and to also follow the SNLI leaderboard settings (e.g., 300D and 600D embeddings), we performed some additional SNLI experiments with the shortcut connections replaced with residual connections, where the input to each next biLSTM layer is the concatenation of the word embedding and the summation of outputs of all previous layers (related to ResNet in computer vision (He et al., 2016)). Table 6 shows these residual-connection SNLI test results and the parameter comparison to shortcut-connection models (using 3 stacked-biLSTM layers, and one 800-unit MLP layer, based on SNLI dev set tuning).

Model	#param	Dev	Test
300D Residual-Stacked-Encoder	9.7M	86.4	85.7
600D Residual-Stacked-Encoder	28.9M	**87.0**	**86.0**
600D Shortcut-Stacked-Encoder	34.7M	86.8	85.9

Table 6: Results on SNLI for the fewer-parameter Residual-Stacked Encoder models. Each model has 3 biLSTM-stacked layers and 1 MLP layer. The #param column denotes the number of parameters in millions.

Acknowledgments

We thank the shared task organizers and the anonymous reviewers. This work was partially supported by a Google Faculty Research Award, an IBM Faculty Award, a Bloomberg Data Science Research Grant, and NVidia GPU awards.

References

Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. 2015. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics.

Samuel R Bowman, Jon Gauthier, Abhinav Rastogi, Raghav Gupta, Christopher D Manning, and Christopher Potts. 2016. A fast unified model for parsing and sentence understanding. *arXiv preprint arXiv:1603.06021* .

Jane Bromley, Isabelle Guyon, Yann LeCun, Eduard Säckinger, and Roopak Shah. 1994. Signature verification using a "Siamese" time delay neural network. In *Advances in Neural Information Processing Systems*. pages 737–744.

Qian Chen, Xiaodan Zhu, Zhen-Hua Ling, Si Wei, Hui Jiang, and Diana Inkpen. 2017. Enhanced lstm for natural language inference. In *Proc. ACL*.

Alexis Conneau, Douwe Kiela, Holger Schwenk, Loic Barrault, and Antoine Bordes. 2017. Supervised learning of universal sentence representations from natural language inference data. *arXiv preprint arXiv:1705.02364* .

Kazuma Hashimoto, Caiming Xiong, Yoshimasa Tsuruoka, and Richard Socher. 2016. A joint many-task model: Growing a neural network for multiple nlp tasks. *arXiv preprint arXiv:1611.01587* .

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. pages 770–778.

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

Lili Mou, Rui Men, Ge Li, Yan Xu, Lu Zhang, Rui Yan, and Zhi Jin. 2015. Natural language inference by tree-based convolution and heuristic matching. *arXiv preprint arXiv:1512.08422* .

Tsendsuren Munkhdalai and Hong Yu. 2016. Neural semantic encoders. *arXiv preprint arXiv:1607.04315* .

Nikita Nangia, Adina Williams, Angeliki Lazaridou, and Samuel R. Bowman. 2017. The repeval 2017 shared task: Multi-genre natural language inference with sentence representations. In *Proceedings of*

RepEval 2017: The Second Workshop on Evaluating Vector Space Representations for NLP. Association for Computational Linguistics.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*. pages 1532–1543. http://www.aclweb.org/anthology/D14-1162.

Zhiguo Wang, Wael Hamza, and Radu Florian. 2017. Bilateral multi-perspective matching for natural language sentences. *arXiv preprint arXiv:1702.03814* .

Adina Williams, Nikita Nangia, and Samuel R Bowman. 2017. A broad-coverage challenge corpus for sentence understanding through inference. *arXiv preprint arXiv:1704.05426* .

Yu Zhang, Guoguo Chen, Dong Yu, Kaisheng Yaco, Sanjeev Khudanpur, and James Glass. 2016. Highway long short-term memory rnns for distant speech recognition. In *Acoustics, Speech and Signal Processing (ICASSP), 2016 IEEE International Conference on*. IEEE, pages 5755–5759.