Enhanced Sentence Alignment Network for Efficient Short Text Matching

Zhe Hu[1] **Zuohui Fu**[2] **Cheng Peng**[1] **Weiwei Wang**[1]
[1]Baidu Inc., Beijing, China
[2]Rutgers University, NJ, USA
[1]{huzhe01, pengcheng06}@baidu.com, elegate@qq.com
[2]zuohui.fu@rutgers.edu

Abstract

Cross-sentence attention has been widely applied in text matching, in which model learns the aligned information between two intermediate sequence representations to capture their semantic relationship. However, commonly the intermediate representations are generated solely based on the preceding layers and the models may suffer from error propagation and unstable matching, especially when multiple attention layers are used. In this paper, we propose an enhanced sentence alignment network with simple gated feature augmentation, where the model is able to flexibly integrate both original word and contextual features to improve the cross-sentence attention. Moreover, our model is less complex with fewer parameters compared to many state-of-the-art structures. Experiments on three benchmark datasets validate our model capacity for text matching.

1 Introduction

Modeling the semantic relationship of a sentence pair is a long standing task in natural language processing, which can be applied in many scenarios such as paraphrase detection and natural language inference (Wang et al., 2017; Bowman et al., 2015; Lan and Xu, 2018). Neural network approaches have achieved impressive results on solving text matching tasks for the good representation learning ability and benefiting from large datasets (Rocktäschel et al., 2015; Wang et al., 2017; Gong et al., 2017).

One of the major paradigms is attention based neural approach which adopts matching and fusion method (Chen et al., 2017; Wang and Jiang, 2016; Duan et al., 2018). Specifically, attention mechanism is used as a key component to compute word or phrase alignments between the two parallel sequences, and then the aligned information is fused to update the sentence representations. Recent work also adopts multiple matching processes to equip model with power on gradually refining the attention results (Yang et al., 2019; Liang et al., 2019; Kim et al., 2019).

Unfortunately, conducting cross-sentence attention between two intermediate sentence representations may lead to unstable matching since different layers aim at capturing different semantic information (Liu et al., 2019a). Also, each intermediate representation is highly correlated to the previous layers, and error propagation would affect the following representations and lead to incorrect alignments since model is unable to amend the information without recalling the original semantic features. Furthermore, in case of multiple alignment blocks are used, models may suffer from difficulty of training such as vanishing gradients, and low-level features are inefficient to be fully trained. Different connection methods are adopted by some recent models to overcome this problem (Tay et al., 2018a; Yang et al., 2019; Nie and Bansal, 2017).

Recently pre-trained language models such as BERT have achieved impressive improvements on text matching tasks (Devlin et al., 2019; Liu et al., 2019b). Despite the promising results, the large parameter size and growing computational requirements make it hard to directly deploy these models to real-time applications (Sanh et al., 2019). Thus designing efficient and effective models to tackle text matching has been of increasing importance.

In this work, we introduce an **E**nhanced **S**entence **A**lignment **N**etwork with Gated Feature Augmentation (**ESAN**), in which our model integrates the word features (embedding outputs) and contextual features (encoding outputs) to the intermediate representations for each cross-sentence attention, as shown in Figure 1. The embedding outputs contain the original word information, and the encoding outputs represent each token with the aggregated contexts, which are helpful to guide the attention layer to properly capture the aligned

information. A gate operation is used to flexibly control how much these two features to be added. Also, incorporating the original semantic features directly to the different levels of representation layers can be viewed as a shortcut connection, which is helpful to reduce the training difficulty on low-level features. We then apply a simple but effective fusion layer to fuse the aligned features and update the sentence representations gradually. Different from previous work (Yang et al., 2019; Kim et al., 2019), we do not apply residual connections between alignment layers or use multiple encoders in alignment layers, and our architecture is *more efficient and less complex* compared with many strong baselines, indicating the feasibility to be deployed in real applications.

To demonstrate the effectiveness of our method, we conduct experiments on three text matching datasets: SNLI, MultiNLI and Quora Question Pairs. The results show our model outperforms strong baselines with fast inference speed. We also conduct model analysis including an ablation study and a case study on attention visualization.

2 Method

2.1 Encoding Layer

Given inputs S_a and S_b, the model first passes each sequence to an embedding layer to get word representations. We use pre-trained word vectors as word embeddings and keep it fixed during training. Character-based word representations is also leveraged, in which we use 1D convolutional network on the character embeddings, and then apply max pooling over time dimension of each token. The word vectors and character-based vectors are concatenated. Following Chen et al. (2018), we further concatenate syntactical features including part-of-speech (POS) tagging feature and binary exact match feature for the NLI task. The embedding outputs are regarded as the final word features: $E_a = \{e_{a_i}\} \in \mathbb{R}^{m \times d}$ and $E_b = \{e_{b_j}\} \in \mathbb{R}^{n \times d}$, where m, n are the sequence lengths. We then pass E_a and E_b to a Bidirectional LSTM encoder to obtain the contextual features $H_a = \{h_{a_i}\}$ and $H_b = \{h_{b_j}\}$, with the same dimension size of d.

Intuitively, the word features contain the original information of each token, and the contextual features represent each word with aggregated context information. They will be used as additional features to enhance the following alignment process.

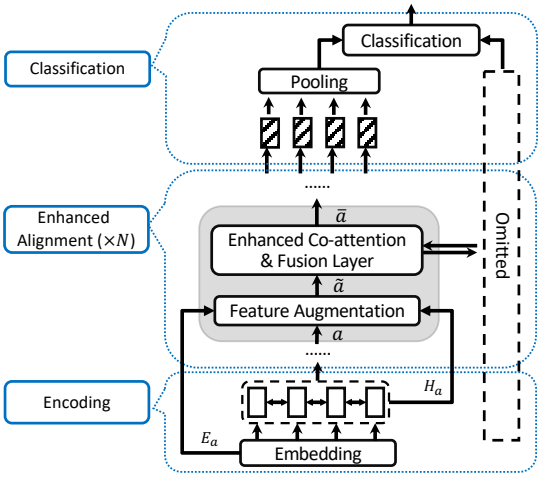

Figure 1: The overview of our model. Word features (E_a) and contextual features (H_a) are used for the enhanced sentence alignment layer, and multiple alignments are stacked with independent parameters. Symmetric structure is applied, and we omit the right part for space limitation.

2.2 Enhanced Sentence Alignment Layer

The proposed enhanced sentence alignment layer takes the intermediate representations a and b as inputs. As shown in Figure 1, the enhanced alignment layer consists of: (1) gated feature augmentation, (2) co-attention and (3) fusion layer. Multiple enhanced sentence alignment layers are stacked to enable the model to gradually refine the alignments.

2.2.1 Gated Feature Augmentation.

Given two intermediate sequence representations a and b, which are the inputs of the current alignment layer, we first augment the word and contextual features to each representation as different levels of the original semantic features. Specifically, for sequence representation $a = \{a_i | a_i \in \mathbb{R}^d, i = 1, 2, ..., m\}$, we augment the word feature E_a and contextual feature H_a with a gate operation to enable the model to selectively keep the features from different parts, which is formally defined as:

$$g_{e_i} = \sigma(W_g a_i + W_e e_{a_i} + z_e) \quad (1)$$

$$g_{h_i} = \sigma(W_g a_i + W_h h_{a_i} + z_h) \quad (2)$$

$$\widetilde{a}_i = a_i + g_{e_i} \circ e_{a_i} + g_{h_i} \circ h_{a_i} \quad (3)$$

where $W_* \in \mathbb{R}^{d \times d}$ and $z_* \in \mathbb{R}^d$ are trainable parameters. The same operation is performed for sequence b. For the first alignment inputs (the encoding outputs), we only augment the word features. Inspired by residual connections (He et al., 2016), we also try a simplified version of augmentation without gate operation:

$$\widetilde{a}_i = a_i + e_{a_i} + h_{a_i} \quad (4)$$

2.2.2 Co-attention.

We then apply the co-attention between two enhanced sequence representations $\widetilde{a}$ and $\widetilde{b}$ to capture their relationship. We first calculate similarity score e_{ij} of token $\widetilde{a}_i$ and token $\widetilde{b}_j$:

$$e_{ij} = \mathrm{ReLU}(W_c\widetilde{a}_i)^{\mathrm{T}} \cdot \mathrm{ReLU}(W_c\widetilde{b}_j) \quad (5)$$

where W_c is a trainable parameter, and the bias term is omitted. Then the attentive representations of each sequence are computed by the weighted sum of the other sequence to highlight the relevant elements:

$$a'_i = \sum_{j=1}^{n} \frac{\exp(e_{ij})}{\sum_{k=1}^{n}\exp(e_{ik})}\widetilde{b}_j \quad (6)$$

$$b'_j = \sum_{i=1}^{m} \frac{\exp(e_{ij})}{\sum_{k=1}^{m}\exp(e_{kj})}\widetilde{a}_i \quad (7)$$

where m, n are the lengths of sequence $\widetilde{a}$ and $\widetilde{b}$.

2.2.3 Fusion Layer

We apply a simple yet effective fusion layer to fuse the aligned features to the original representations. The output of fusion layer $\bar{a}$ is computed as follows:

$$\bar{a}_i = \mathrm{ReLU}(W_f[\widetilde{a}_i; a'_i; \widetilde{a}_i - a'_i; \widetilde{a}_i \circ a'_i] + z_f) \quad (8)$$

where W_f and z_f are trainable parameters, $[;]$ represents concatenation, and $\circ$ is element-wise product. The output has the same dimension size as $\widetilde{a}$ and a'.

2.3 Pooling and Classification Layer

We use both mean and max pooling on each sequence to get the corresponding vector representations, as inputs of the classification layer. Mean pooling aggregates global semantics and max pooling represents the import semantic features. Then we apply an MLP with softmax to get the final distributions. Formally, assume the outputs of the last fusion layer is V_a and V_b, we first compute the feature vector:

$$V'_a = [\frac{1}{m}\sum_{i=1}^{m} v_{a_i}; \max_{i=1}^{m} v_{a_i}] \quad (9)$$

$$V'_b = [\frac{1}{m}\sum_{i=1}^{m} v_{b_i}; \max_{i=1}^{m} v_{b_i}] \quad (10)$$

$$V = [V'_a; V'_b; V'_a - V'_b; V'_a \circ V'_b] \quad (11)$$

Then a multi-layer perceptron (MLP) is used to calculate the final target:

$$\hat{y} = \mathrm{softmax}(W_2\mathrm{ReLU}(W_1V + z_1) + z_2) \quad (12)$$

where $W_*$and z_* are trainable parameters.

Model	Test Accuracy (%)
ESIM (Chen et al., 2017)	88.0
BiMPM (Wang et al., 2017)	87.5
DIIN (Gong et al., 2017)	88.0
CAFE (Tay et al., 2018b)	88.5
CSRAN (Tay et al., 2018a)	88.7
ADIN (Liang et al., 2019)	88.8
RE2 (Yang et al., 2019)	88.9
OSOA-DFN (Liu et al., 2019a)	88.8
ESAN	**89.0**

Table 1: Experiment results on SNLI dataset. Our model yields better results (in bold).

3 Experimental Setups

Datasets and Preprocessing. We evaluate our model on three large-scale benchmark datasets: SNLI dataset (Bowman et al., 2015), MultiNLI dataset (Williams et al., 2018) and Quora Question Pairs (Quora) dataset. We follow the same data splits as provided in the original papers for SNLI and MultiNLI. For Quora, we use the same split as Wang et al. (2017)[1]. Accuracy is used to evaluate the model performance for all three datasets.

Training Details and Parameters. We tune the number of enhanced alignment layers from 2 to 3 in all experiments, which can be easily extended to more layers. We apply 300D-840B Glove (Pennington et al., 2014) as pre-trained word vectors. The 1D convolutional network is used for char embedding with kernel size 5 and 100 filters. We tune the number of recurrent layers from 1 to 2, and the dimension of feed-forward layers from 150 to 300 with ReLU (Glorot et al., 2011) as activation function. Adam optimizer (Kingma and Ba, 2014) is used with β_1 to be 0.9 and β_2 to be 0.999 during training. We use cropping or padding to limit each token to have 16 characters in char embedding. Dropout with dropout rate of 0.2 is applied to prevent overfitting. We set initial learning rate as 0.001 with exponential decay. The batch size is tuned from 64 to 256. More details are in Supplementary.

4 Results

4.1 Quantitative Results

Our model outperforms strong baselines with competitive results on all three datasets. For a fair comparison, we do not include the methods with pre-trained language models such as BERT (Devlin et al., 2019) or ensemble systems.

[1]Data statistics are in Supplementary.

Model	Test Accuracy (%)
BiMPM (Wang et al., 2017)	88.2
DIIN (Gong et al., 2017)	89.1
CAFE (Tay et al., 2018b)	88.7
CSRAN (Tay et al., 2018a)	89.2
RE2 (Yang et al., 2019)	89.2
OSOA-DFN (Liu et al., 2019a)	89.0
Enhanced-RCNN (Peng et al., 2020)	**89.3**
ESAN	**89.3**

Table 2: Experiment results on Quora datasets. Our model yields better results than all comparisons (in bold).

Model	Test Accuracy (%)	
	Matched	Mismatched
DIIN (Gong et al., 2017)	78.8	77.8
CAFE (Tay et al., 2018b)	78.7	77.9
AF-DMN (Duan et al., 2018)	76.9	76.3
MwAN (Tan et al., 2018)	78.5	77.7
ADIN (Liang et al., 2019)	78.8	77.9
ESAN	**79.3**	**78.4**

Table 3: Experiment results on MultiNLI dataset. Our model yields better results than all comparisons (in bold).

The results for SNLI and Quora are shown in Table 1 and Table 2. For SNLI, our model achieves 89.0% test accuracy, which is higher than all comparisons including some strong state-of-the-art models. For Quora, our model also achieves the best performance, with 89.3% test accuracy. Table 3 presents the results on MultiNLI, and our model produces higher accuracy on both in-domain (matched) and out-domain (mismatched) test sets, which further proves the model ability for natural language inference task. Above all, the results on the challenging datasets verify our model effectiveness for solving text matching tasks.

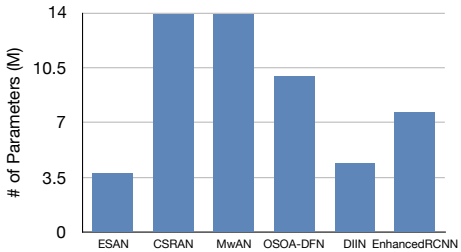

Figure 2: Total number of parameters for different models on Quora dataset.

4.2 Model Analysis

Ablation Study. To verify the effectiveness of our model components, we conduct an ablation study on Quora as shown in Table 4. The first line represents the model variant without feature augmentation (using original co-attention between intermediate representations) and the result drops dramatically. It shows that feature augmentation plays a key role to enhance the alignment process. In the next two settings, after removing word features and contextual features respectively, both the results drop, and removing contextual features brings more decrease to the final performance. These two features are complementary to each other to improve the following cross-sentence attention. For the last ablation study, we apply simple augmentation without gate as Equation 4, and the performance decreases by 0.3 percentage point, which indicates the usefulness of the gate operation.

Models	Acc. (%)
ESAN	**89.4**
(w/o Feat. Augment.)	88.1
(w/o Word Feat.)	89.0
(w/o Contextual Feat.)	88.9
(w/ Simple Augment.)	89.1

Table 4: Analysis of model components on Quora dev set.

Models	parameter size	time (s/batch)
ESAN	3.9M	0.04 ± 0.01
BERT	109.5M	0.88 ± 0.06

Table 5: Paramter size and inference time for ESAN and BERT on Quora Question Pairs.

Model Efficiency. Figure 2 presents the comparison of total number of parameters for our model and baselines. Some strong comparisons such as CSRAN and MwAN contain more than 10M parameters, while our model has less parameters (3.9M) and achieves better results. We also compare the inference time with BERT to show the efficiency of our model in Table 5. Specifically, we set the sentence lengths as 20. Both models are required to make predictions for a batch of 8 sentence pairs on a MacBook Pro with Intel Core i7 CPUs. For BERT, we add a linear layer on top of the [CLS] token for classification as the original paper did (Devlin et al., 2019). We report the average and the standard deviation of processing 1000 batches. From the results we can see ESAN has a higher inference speed than BERT with less model complexity, which further indicates ESAN is more efficient and can be applied in many real scenarios.

4.3 Attention Visualization.

We present a case study through the attention visualization to investigate what our model learns in cross-sentence attention. We take an instance from SNLI, where sentence 1 is "police officer with riot shield stands in front of crowd" and sentence 2 is "a police officer stands in front of a crowd". The

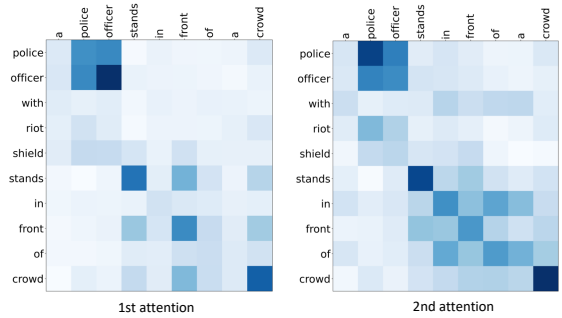

Figure 3: Visualization of attention results for a natural language inference case.

attention results are shown in Figure 3.

In the first attention, the model tends to align elements mostly in word-level. For example, "stands" and "crowd" in two sequences are successfully connected. Also, the model correctly aligns phrase "police officer" together which is one of the key components. In the second attention, the model tries to refine the attention distribution and gives "stands" and "crowd" larger weights. Also, the model tends to align longer phrases together instead of individual words. For example, phrases "in front of" in two sequences are connected. Notably, "riot shield" is also aligned to "police officer". We hypothesis that the model learns this phrase is used to describe entity "police officer", thus correctly aligning these two would help to make the final decision. With the proper alignments, the model correctly classifies their relationship as "entailment".

5 Related Work

Text matching is a key technique for many NLP tasks such as natural language inference (Bowman et al., 2015), paraphrase identification (Wang et al., 2017) and machine reading comprehension (Rajpurkar et al., 2016; Wang et al., 2018). As a long standing problem, this area has been in the center of attention and investigated widely.

Benefiting from large-scale datasets, neural networks have achieved much success for solving this problem. One of the paradigms uses sentence encoding structure, in which two sentences are encoded into vector representations, and then the vectors are combined to make the final prediction (Conneau et al., 2017; Yin and Schütze, 2015; Mueller and Thyagarajan, 2016). However, the interaction of the two input sequences is not directly considered during the encoding process, which makes the model difficult to capture complex relationship.

Later work adopts matching and aggregation method to model the alignments of the two sentences. Wang and Jiang (2016) uses a match-LSTM to conduct word-level matching of the two sequences. Parikh et al. (2016) propose a simple attention operation and use a feed-forward network to integrate the aligned representations. BiMPM (Wang et al., 2017) uses multi-perspective matching operation to compare two sequences, and applies a Bi-LSTM network for aggregation. Gong et al. (2017) uses DensNet as feature extractor to extract the semantic feature from the interaction tensor.

To better capture the sentence alignments in different levels, multiple attention operations can be stacked together. Yang et al. (2019) propose a simple but effective framework with richer alignment features. Tay et al. (2018a) leverages multi-level attention refinement component to conduct more extensive matching and improve the results. ADIN (Liang et al., 2019) stacks asynchronous inference layers for a multi-step reasoning process.

Recently the pre-trained language models have achieved state-of-the-art results on text matching tasks with pre-training and finetuning procedure (Devlin et al., 2019). Nevertheless, large parameter size and slow inference speed make it hard to directly deploy these structures to the real applications. Different from above methods, we propose a simple but effective gated augmentation layer to enrich the intermediate representations with the original word features and contextual features, and thus guide model to produce better alignments.

6 Conclusions

In this work, we present ESAN, an enhanced sentence alignment network for text matching. We flexibly integrate both word and contextual features to the intermediate representations with a gate operation to conduct better co-attention between two sequences. Our model outperforms strong baselines on three datasets and contains fewer parameters, which indicates the model capacity on producing proper alignments for text matching. In the future, we also plan to apply our methods to some other scenarios such as question answering.

Acknowledgements

This work is done when Weiwei Wang was at Baidu. We thank the anonymous reviewers for their constructive suggestions. We also thank Yu Yin for the helpful discussion and suggestions, and Zizhe Xie for proofreading the paper.

References

Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. 2016. Tensorflow: A system for large-scale machine learning. In *12th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 16)*, pages 265–283.

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*, pages 632–642, Lisbon, Portugal. Association for Computational Linguistics.

Qian Chen, Xiaodan Zhu, Zhen-Hua Ling, Diana Inkpen, and Si Wei. 2018. Neural natural language inference models enhanced with external knowledge. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2406–2417, Melbourne, Australia. Association for Computational Linguistics.

Qian Chen, Xiaodan Zhu, Zhen-Hua Ling, Si Wei, Hui Jiang, and Diana Inkpen. 2017. Enhanced LSTM for natural language inference. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1657–1668, Vancouver, Canada. Association for Computational Linguistics.

Alexis Conneau, Douwe Kiela, Holger Schwenk, Loïc Barrault, and Antoine Bordes. 2017. Supervised learning of universal sentence representations from natural language inference data. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 670–680, Copenhagen, Denmark. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Chaoqun Duan, Lei Cui, Xinchi Chen, Furu Wei, Conghui Zhu, and Tiejun Zhao. 2018. Attention-fused deep matching network for natural language inference. In *IJCAI*, pages 4033–4040.

Xavier Glorot, Antoine Bordes, and Yoshua Bengio. 2011. Deep sparse rectifier neural networks. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, volume 15 of *Proceedings of Machine Learning Research*, pages 315–323, Fort Lauderdale, FL, USA. PMLR.

Yichen Gong, Heng Luo, and Jian Zhang. 2017. Natural language inference over interaction space. *arXiv preprint arXiv:1709.04348*.

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.

Seonhoon Kim, Inho Kang, and Nojun Kwak. 2019. Semantic sentence matching with densely-connected recurrent and co-attentive information. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pages 6586–6593.

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

Wuwei Lan and Wei Xu. 2018. Neural network models for paraphrase identification, semantic textual similarity, natural language inference, and question answering. In *Proceedings of the 27th International Conference on Computational Linguistics (COLING)*.

Di Liang, Fubao Zhang, Qi Zhang, and Xuanjing Huang. 2019. Asynchronous deep interaction network for natural language inference. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2692–2700, Hong Kong, China. Association for Computational Linguistics.

Mingtong Liu, Yujie Zhang, Jinan Xu, and Yufeng Chen. 2019a. Original semantics-oriented attention and deep fusion network for sentence matching. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2652–2661, Hong Kong, China. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019b. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Jonas Mueller and Aditya Thyagarajan. 2016. Siamese recurrent architectures for learning sentence similarity. In *thirtieth AAAI conference on artificial intelligence*.

Yixin Nie and Mohit Bansal. 2017. Shortcut-stacked sentence encoders for multi-domain inference. In *Proceedings of the 2nd Workshop on Evaluating Vector Space Representations for NLP*, pages 41–45, Copenhagen, Denmark. Association for Computational Linguistics.

Ankur Parikh, Oscar Täckström, Dipanjan Das, and Jakob Uszkoreit. 2016. A decomposable attention model for natural language inference. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2249–2255, Austin, Texas. Association for Computational Linguistics.

Shuang Peng, Hengbin Cui, Niantao Xie, Sujian Li, Jiaxing Zhang, and Xiaolong Li. 2020. Enhanced-rcnn: An efficient method for learning sentence similarity. In *Proceedings of The Web Conference 2020*, pages 2500–2506.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Austin, Texas. Association for Computational Linguistics.

Tim Rocktäschel, Edward Grefenstette, Karl Moritz Hermann, Tomáš Kočiskỳ, and Phil Blunsom. 2015. Reasoning about entailment with neural attention. *arXiv preprint arXiv:1509.06664*.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*.

Chuanqi Tan, Furu Wei, Wenhui Wang, Weifeng Lv, and Ming Zhou. 2018. Multiway attention networks for modeling sentence pairs. In *IJCAI*, pages 4411–4417.

Yi Tay, Anh Tuan Luu, and Siu Cheung Hui. 2018a. Co-stack residual affinity networks with multi-level attention refinement for matching text sequences. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4492–4502, Brussels, Belgium. Association for Computational Linguistics.

Yi Tay, Anh Tuan Luu, and Siu Cheung Hui. 2018b. Compare, compress and propagate: Enhancing neural architectures with alignment factorization for natural language inference. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1565–1575, Brussels, Belgium. Association for Computational Linguistics.

Shuohang Wang and Jing Jiang. 2016. Learning natural language inference with LSTM. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1442–1451, San Diego, California. Association for Computational Linguistics.

Wei Wang, Ming Yan, and Chen Wu. 2018. Multi-granularity hierarchical attention fusion networks for reading comprehension and question answering. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1705–1714, Melbourne, Australia. Association for Computational Linguistics.

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 Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1112–1122, New Orleans, Louisiana. Association for Computational Linguistics.

Runqi Yang, Jianhai Zhang, Xing Gao, Feng Ji, and Haiqing Chen. 2019. Simple and effective text matching with richer alignment features. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4699–4709, Florence, Italy. Association for Computational Linguistics.

Wenpeng Yin and Hinrich Schütze. 2015. Convolutional neural network for paraphrase identification. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 901–911, Denver, Colorado. Association for Computational Linguistics.

A Supplemental Material

A.1 Additional Experiment Details

Data Statistics. The statistics of datasets are shown in Table 6. For SNLI and MultiNLI, we follow the same data split as original papers (Bowman et al., 2015; Williams et al., 2018), and for Quora we use the same split as Wang et al. (2017). Notably, the test set labels of MultiNLI are not provided, and we obtain the test accuracy from submission on Kaggle[2].

Preprocessing. We use hard cutoff for sentence length on all three datasets with cropping or padding. For Quora and SNLI, we set length as 30, and for MultiNLI we set length as 48. We mask the padding tokens during experiments. We only tokenize the sentence during preprocessing.

[2]In-domain: https://www.kaggle.com/c/multinli-matched-open-evaluation/leaderboard;
out-domain: https://www.kaggle.com/c/multinli-mismatched-open-evaluation/overview

Dataset	Train	Dev	Test	# Classes
SNLI	549K	9.8K	9.8K	3
Quora	384K	10K	10K	2
MultiNLI-1	392k	9.8K	9.8K	3
MultiNLI-2	392k	9.8K	9.8K	3

Table 6: Statistics on the datasets for experiments. MultiNLI-1 represents in-domain setting, and MultiNLI-2 indicates out-domain setting.

Training Details. We implement our model using TensorFlow (Abadi et al., 2016) and train the experiments on NVIDIA Tesla V100 GPU. CUDNN implementation for BiLSTM network is used to improve speed. For all feed-forward layers, we apply ReLU (Glorot et al., 2011) as activation function, and Adam optimizer (Kingma and Ba, 2014) is used with β_1 to be 0.9 and β_2 to be 0.999 during training. We use cropping or padding to limit each token to have 16 characters in char embedding. The threshold for gradient clipping is set to 5, and l_2 regularizer strength is set to 6e-5. Each epoch takes around 4.4 minutes with a batch size of 128 on Quora. Cross-entropy is applied as loss function during training.

A.2 Does Feature Augmentation Improve Alignments?

To better understand how our model uses augmented features to enhance the cross-sentence alignments, we also calculate attention results using the original intermediate representations and show the visualizations in Figure 4. The two figures in the upper row are attention results computed with original intermediate representations, and the lower row shows the attention results computed with enhanced representations[3]. The sentence 1 is "police officer with riot shield stands in front of crowd" and the sentence 2 is "a police officer stands in front of a crowd?"

As we can see, in the first alignment, computing the cross-sentence attention with original intermediate representations would bring some noisy alignments (shown in upper left). However, the attention results with enhanced representations contain less noises and the key components such as "police officer" and "crowd" are correctly aligned between two sequences (shown in lower left). In the second alignment, similar as previous, the attention with original representations are noisier and the dark cluster covers more irrelevant parts (shown in upper right). With the augmentation of original semantic features, we can observe in the lower right figure the attention is properly conducted with better connections between two sequences.

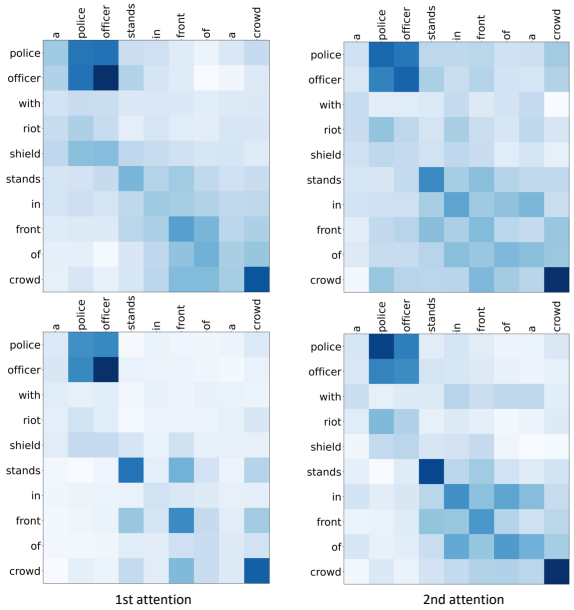

Figure 4: Visualization of attention results. The upper row are the attention computed using original intermediate representations, and the lower row are computed using enhanced sentence representations.

Above all, the attention results with original intermediate representations contain more noises, which would lead to incorrect alignments and unstable matching. With the augmentation of the original semantic features, the model is able to produce a proper alignments and thus better capture their semantic relationship.

[3]Notably here we only calculate the additional attention results with the original intermediate representations, and do not use them as inputs for the following layers.