Refining Raw Sentence Representations for Textual Entailment Recognition via Attention

Jorge A. Balazs, Edison Marrese-Taylor, Pablo Loyola, Yutaka Matsuo
Graduate School of Engineering
The University of Tokyo
{jorge,emarrese,pablo,matsuo}@weblab.t.u-tokyo.ac.jp

Abstract

In this paper we present the model used by the team Rivercorners for the 2017 RepEval shared task. First, our model separately encodes a pair of sentences into variable-length representations by using a bidirectional LSTM. Later, it creates fixed-length raw representations by means of simple aggregation functions, which are then refined using an attention mechanism. Finally it combines the refined representations of both sentences into a single vector to be used for classification. With this model we obtained test accuracies of 72.057% and 72.055% in the matched and mismatched evaluation tracks respectively, outperforming the LSTM baseline, and obtaining performances similar to a model that relies on shared information between sentences (ESIM). When using an ensemble both accuracies increased to 72.247% and 72.827% respectively.

1 Introduction

The task of Natural Language Inference (NLI) aims at characterizing the semantic concepts of entailment and contradiction, and is essential in tasks ranging from information retrieval to semantic parsing to commonsense reasoning, as both entailment and contradiction are central concepts in natural language meaning (Katz, 1972; van Benthem, 2008).

The aforementioned task has been addressed with a variety of techniques, including those based on symbolic logic, knowledge bases, and neural networks. With the advent of deep learning techniques, NLI has become an important testing ground for approaches that employ distributed word and phrase representations, which are typical of these models.

In this context, the Second Workshop on Evaluating Vector Space Representations for NLP (RepEval 2017) features a shared task meant to evaluate natural language understanding models based on sentence encoders by the means of NLI in the style of a three-class balanced classification problem over sentence pairs. The shared task includes two evaluations, a standard in-domain (matched) evaluation in which the training and test data are drawn from the same sources, and a cross-domain (mismatched) evaluation in which the training and test data differ substantially. This cross-domain evaluation is aimed at testing the ability of submitted systems to learn representations of sentence meaning that capture broadly useful features.

2 Proposed Model

Our work is related to intra-sentence attention models for sentence representation such as the ones described by Liu et al. (2016) and Lin et al. (2017). In particular, our model is based on the notion that, when reading a sentence, we usually need to re-read certain portions of it in order to obtain a comprehensive understanding. To model such phenomenon, we rely on an attention mechanism able to iteratively obtain a richer and more expressive version of a raw sentence representation. The model's architecture is described below:

Word Representation Layer: This layer is in charge of generating a comprehensive vector representation of each token for a given sentence. We construct this representation based on up to two basic components:

- Pre-trained word embeddings: We take pre-trained word embeddings and use them to generate a raw word representation. This can

Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, pages 41–45

be seen as a simple lookup-layer that returns a word vector for each provided word index.

- Character embeddings: We generate a character-based representation of each word, which we concatenate to the word vectors as returned by the previous component. We start by generating a randomly initialized character embedding matrix C. Then, we split each word into its component characters, get their corresponding character embedding vectors from C and feed them into a unidirectional Long Short-Term Memory Network (LSTM) (Hochreiter and Schmidhuber, 1997). We then choose the last hidden state returned by the LSTM as the fixed-size character-based vector representation for each token. Our embedding matrix C is trained with the rest of the model (Wang et al., 2017).

Context Representation Layer: This layer complements the vectors generated by the Word Representation Layer by incorporating contextual information into them. To do this, we utilize a bidirectional LSTM that reads through the embedded sequence and returns the hidden states for each time step. These are context-aware representations focused on each position. Formally, let $\mathcal{S}$ be a sentence such as $\mathcal{S} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\}$, where each $\boldsymbol{x}_i$ is an embedded word vector as returned by the previous layer, then the context-rich word representation $\boldsymbol{h}_i$ is calculated as follows for each time step $i = 1, \ldots, n$:

$$\overrightarrow{\boldsymbol{h}}_i = LSTM(\boldsymbol{x}_i, \overrightarrow{\boldsymbol{h}}_{i-1}) \tag{1}$$

$$\overleftarrow{\boldsymbol{h}}_i = LSTM(\boldsymbol{x}_i, \overleftarrow{\boldsymbol{h}}_{i+1}) \tag{2}$$

$$\boldsymbol{h}_i = [\overrightarrow{\boldsymbol{h}}_i; \overleftarrow{\boldsymbol{h}}_i] \tag{3}$$

Where $\overrightarrow{\boldsymbol{h}}_i$ is the forward contextual vector representation of $\boldsymbol{x}_i$, $\overleftarrow{\boldsymbol{h}}_i$ the backward one, and $[\,\cdot\,;\,\cdot\,]$ represents the concatenation of two vectors. The output of this layer is a variable-length sentence representation for both the premise and hypothesis. We then define a pooling layer in charge of a generating a raw fixed-size representation of each sentence.

Pooling Layer: This layer is in charge of generating a crude sentence representation vector by reducing the sequence dimension using one of four simple operations, all of which are fed the context-aware token representations obtained previously:

$$\bar{\boldsymbol{h}} = \frac{1}{n}\sum_{i=1}^{n} \boldsymbol{h}_i \tag{4}$$

$$\bar{\boldsymbol{h}} = \sum_{i=1}^{n} \boldsymbol{h}_i \tag{5}$$

$$\bar{\boldsymbol{h}} = [\overrightarrow{\boldsymbol{h}}_n; \overleftarrow{\boldsymbol{h}}_1] \tag{6}$$

$$\bar{\boldsymbol{h}} = \max_{i=1\ldots n} \boldsymbol{h}_i \tag{7}$$

These operations correspond to the *mean* of the word representations (eq. 4), their *sum* (eq. 5), the concatenation of the *last* hidden state for each direction (eq. 6), and the *maximum* one (eq. 7).

Inner Attention Layer: To refine the representations generated by the pooling strategy, we use a global attention mechanism (Luong et al., 2015; Vinyals et al., 2015) that compares each context-aware token representation $\boldsymbol{h}_i$ with the raw sentence representation $\bar{\boldsymbol{h}}$. Formally,

$$u_i = \boldsymbol{v}^\top \tanh(W[\bar{\boldsymbol{h}}; \boldsymbol{h}_i]) \tag{8}$$

$$\alpha_i = \frac{\exp u_i}{\sum_{k=1}^{n} \exp u_k} \tag{9}$$

$$\bar{\boldsymbol{h}}' = \sum_{i=1}^{n} \alpha_i \boldsymbol{h}_i \tag{10}$$

Where both $\boldsymbol{v}$ and W are trainable parameters and $\bar{\boldsymbol{h}}'$ is the refined sentence representation[1].

Aggregation Layer: We apply two matching mechanisms to aggregate the refined sentence representations, which are directly aimed at extracting relationships between the premise and the hypothesis. Concretely, we concatenate the representations of the premise $\bar{\boldsymbol{h}}'_P$ and hypothesis $\bar{\boldsymbol{h}}'_H$ in addition to their element-wise product ($\odot$) and the absolute value ($|\cdot|$) of their difference, obtaining the vector $\boldsymbol{r}$. These last two operations, first proposed by Mou et al. (2015), can be seen as a sentence matching strategy.

$$\boldsymbol{h}_{mul} = \bar{\boldsymbol{h}}'_P \odot \bar{\boldsymbol{h}}'_H \tag{11}$$

$$\boldsymbol{h}_{dif} = |\bar{\boldsymbol{h}}'_P - \bar{\boldsymbol{h}}'_H| \tag{12}$$

$$\boldsymbol{r} = [\bar{\boldsymbol{h}}'_P; \bar{\boldsymbol{h}}'_H; \boldsymbol{h}_{mul}; \boldsymbol{h}_{dif}] \tag{13}$$

Dense Layer: Finally, $\boldsymbol{r}$ is fed to a fully-connected layer whose output is a vector containing the logits for each class, which are then fed to

[1]The refined sentence representation $\bar{\boldsymbol{h}}'$ for both premise and hypothesis is the final representation in which both are treated as separate entities. The representations produced by our best-performing model are available in https://zenodo.org/record/825946.

a softmax function for obtaining their probability distribution. The class with the highest probability is chosen as the predicted relationship between premise and hypothesis.

3 Experiments

To make our results comparable to the baselines reported in the Kaggle platform we randomly sampled 15% of the SNLI corpus (Bowman et al., 2015) and added it to the MultiNLI corpus.

We used the pre-trained 300-dimensional GloVe vectors trained on 840B tokens (Pennington et al., 2014). These embeddings were not fine-tuned during training and unknown word vectors were initialized by randomly sampling from the uniform distribution in $(-0.05, 0.05)$.

Each character embedding was initialized as a 20-dimensional vector and the character-level LSTM output dimension was set to 50. The word-level LSTM output dimension was set to 300, which means that after concatenating word-level and character-level representations the word vectors for each direction are 350-dimensional (i.e., $\boldsymbol{h}_i \in \mathbb{R}^{700}$).

For the Inner Attention Layer we defined the parameter W as a square matrix matching the dimension of the concatenated vector $[\bar{\boldsymbol{h}}; \boldsymbol{h}_i]$ (i.e., $W \in \mathbb{R}^{1400\times1400}$), and $\boldsymbol{v}$ as a vector matching the same dimension (i.e., $\boldsymbol{v} \in \mathbb{R}^{1400}$). Both W and $\boldsymbol{v}$ were initialized by randomly sampling from the uniform distribution on the interval $(-0.005, 0.005)$.

The final layer was created as a 3-layer MLP with 2000 hidden units each, and with ReLU activations.

Additionally, we used the Rmsprop optimizer with a learning rate of 0.001. We applied dropout of 0.25 only between the MLP layers of the Dense Layer.

Further, we found out that normalizing the capitalization of words by making all characters lowercase, and transforming numbers into a specific numeric token improved the model's performance while reducing the size of the embedding matrix. We also ignored the sentence pairs with a premise longer than 200 words during training (for improved memory stability), and those without a valid label ("-") both during training and validation.

Since one of the most conceptually important parts of our model was the raw sentence representation created in the Pooling Layer, we used four different methods for generating it (eqs. 4 – 7). Results are reported in Table 1.

We also tried using other architectures that rely on some sort of "inner" attention such as the *self-attentive* model proposed by Lin et al. (2017) and the *co-attentive* model by Xiong et al. (2016), but our preliminary results were not promising so we did not invest in fine-tuning them.

All the experiments were repeated without using character-level embeddings (i.e., $\boldsymbol{h}_i \in \mathbb{R}^{600}$).

4 Results

Table 1 presents the results of using different pooling strategies for generating a raw sentence representation vector from the word vectors. We can observe that that both the *mean* method, and picking the last hidden state for both directions performed slightly better than the two other strategies, however at 95% confidence we cannot assert that any of these methods is statistically different from one another.

This could be interpreted as if any of the four methods was good enough for capturing the overall meaning of the sentence, and the heavy lifting was done by the attention mechanism. It would be interesting to test these four strategies without the presence of attention to see whether it really plays an important role in this task or whether the predictive power lies within the sentence matching mechanism.

Method	**w/o. chars**	**w. chars**
mean	71.3 ± 1.2	71.3 ± 0.7
sum	70.7 ± 1.0	70.9 ± 0.8
last	70.9 ± 0.6	71.0 ± 1.2
max	70.6 ± 1.1	71.0 ± 1.1

Table 1: Mean matched validation accuracies (%) broken down by type of pooling method and presence or absence of character embeddings. Confidence intervals are calculated at 95% confidence over 10 runs for each method.

Another interesting result, as shown by Table 1 and Table 2, is that the model seemed to be insensitive to the usage of character embeddings, which was surprising because in our experiments with more complex models relying on shared information between premise and hypothesis, such as the one presented by Wang et al. (2017), the usage of character embeddings had a considerable impact

Method	w/o. chars	w. chars
mean	**72.3**	71.8
sum	71.6	71.6
last	71.4	**72.1**
max	71.1	71.6

Table 2: Best matched validation accuracies (%) obtained by each pooling method in presence and absence of character embeddings.

in model performance[2].

In Table 3 we report the accuracies obtained by our best model in both matched (first 5 genres) and mismatched (last 5 genres) development sets. We can observe that our implementation performed like ESIM overall, however ESIM relies on an attention mechanism that has access to both premise and hypothesis (Chen et al., 2017), while our model's treats each one separately. This supports the notion that inner attention is a powerful concept.

Genre	CBOW	ESIM	InnerAtt
Fiction	67.5	73.0	73.2
Government	67.5	74.8	75.2
Slate	60.6	67.9	67.2
Telephone	63.7	72.2	73.0
Travel	64.6	73.7	72.8
9/11	63.2	71.9	70.5
Face-to-face	66.3	71.2	74.5
Letters	68.3	74.7	75.4
Oup	62.8	71.7	71.5
Verbatim	62.7	71.9	69.5
MultiNLI Overall	**64.7**	**72.2**	**72.3**

Table 3: Validation accuracies (%) for our best model broken down by genre. Both CBOW and ESIM results are reported as in (Williams et al., 2017).

We picked the best model based on the best validation accuracy score obtained on the matched development set (72.257%). This model is as described in the previous section but without using character embeddings[3].

In addition, we created an ensemble by training 4 models as described earlier but initialized with different random seeds. The prediction is made by averaging the probability distributions returned by each model and then picking the class with the highest probability for each example. This improved our best test results, as reported by Kaggle, from 72.057% to 72.247% in the matched evaluation track, and from 72.055% to 72.827% in the mismatched evaluation track.

5 Conclusions and Future work

We presented the model used by the team Rivercorners in the 2017 RepEval shared task. Despite being conceptually simple and not relying on shared information between premise and hypothesis for encoding each sentence, nor on tree structures, our implementation achieved results as good as the ESIM model.

As future work we plan to incorporate part-of-speech embeddings to our implementation and concatenate them at the same level as we did with the character embeddings. We also plan to use pre-trained character embeddings to see whether they have any positive impact on performance.

Additionally, we think we could obtain better results by fine-tuning some hyperparameters such as the character embedding dimensions, the character-level LSTM encoder output dimension, and the Dense Layer architecture.

Further, we would like to see how different types of attention affect the overall performance. For this implementation we used the *concat* scoring scheme (eq. 8), as described by Luong et al. (2015), but there are several others that could provide better results.

Finally, we would like to exploit the structured nature of dependency parse trees by means of recursive neural networks (Tai et al., 2015) to enrich our initial sentence representations.

6 Resources

The code for replicating the results presented in this paper is available in the following link: `https://github.com/jabalazs/repeval_rivercorners`.

7 Acknowledgements

We thank the anonymous reviewers for helping us improve this paper through their feedback.

[2]This type of models were not allowed in this competition which is why we do not report further on them.

[3]Without the use of character embeddings, the sentence representations are 600-dimensional.

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*. Association for Computational Linguistics, Lisbon, Portugal, pages 632–642. http://aclweb.org/anthology/D15-1075.

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

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Computation* 9(8):1735–1780. http://www.bioinf.jku.at/publications/older/2604.pdf.

Jerrold J. Katz. 1972. *Semantic Theory*. Harper & Row, New York.

Zhouhan Lin, Minwei Feng, Cicero Nogueira dos Santos, Mo Yu, Bing Xiang, Bowen Zhou, and Yoshua Bengio. 2017. A structured self-attentive sentence embedding. In *Proceedings of the 5th International Conference on Learning Representations (ICLR)*. https://arxiv.org/pdf/1703.03130.pdf.

Yang Liu, Chengjie Sun, Lei Lin, and Xiaolong Wang. 2016. Learning natural language inference using bidirectional LSTM model and inner-attention. *CoRR* abs/1605.09090. http://arxiv.org/pdf/1605.09090.pdf.

Thang Luong, Hieu Pham, and Christopher D. Manning. 2015. Effective Approaches to Attention-based Neural Machine Translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Lisbon, Portugal, pages 1412–1421. http://aclweb.org/anthology/D15-1166.

Lili Mou, Hao Peng, Ge Li, Yan Xu, Lu Zhang, and Zhi Jin. 2015. Discriminative neural sentence modeling by tree-based convolution. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Lisbon, Portugal, pages 2315–2325. http://aclweb.org/anthology/D15-1279.

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.

Kai Sheng Tai, Richard Socher, and Christopher D. Manning. 2015. Improved semantic representations from tree-structured long short-term memory networks. *arXiv preprint arXiv:1503.00075* https://arxiv.org/pdf/1503.00075.pdf.

Johan van Benthem. 2008. A brief history of natural logic. In M. Chakraborty, B. Löwe, M. Nath Mitra, and S. Sarukki, editors, *Logic, Navya-Nyaya and Applications: Homage to Bimal Matilal*, College Publications.

Oriol Vinyals, Lukasz Kaiser, Terry Koo, Slav Petrov, Ilya Sutskever, and Geoffrey Hinton. 2015. Grammar as a foreign language. In *Advances in Neural Information Processing Systems*. pages 2773–2781. http://papers.nips.cc/paper/5635-grammar-as-a-foreign-language.

Zhiguo Wang, Wael Hamza, and Radu Florian. 2017. Bilateral multi-perspective matching for natural language sentences. *CoRR* abs/1702.03814. http://arxiv.org/abs/1702.03814.

Adina Williams, Nikita Nangia, and Samuel R. Bowman. 2017. A broad-coverage challenge corpus for sentence understanding through inference http://arxiv.org/pdf/1704.05426.pdf.

Caiming Xiong, Victor Zhong, and Richard Socher. 2016. Dynamic coattention networks for question answering. *arXiv preprint arXiv:1611.01604* .