Enhancing and Combining Sequential and Tree LSTM for Natural Language Inference

Qian Chen
University of Science and Technology of China
cq1231@mail.ustc.edu.cn

Xiaodan Zhu
National Research Council Canada
xiaodan.zhu@nrc-cnrc.gc.ca

Zhenhua Ling
University of Science and Technology of China
zhling@ustc.edu.cn

Si Wei
iFLYTEK Research
siwei@iflytek.com

Hui Jiang
York University
hj@cse.yorku.ca

Abstract

Reasoning and inference are central to human and artificial intelligence. Modeling inference in human language is notoriously challenging but is fundamental to natural language understanding and many applications. With the availability of large annotated data, neural network models have recently advanced the field significantly. In this paper, we present a new state-of-the-art result, achieving the accuracy of 88.3% on the standard benchmark, the Stanford Natural Language Inference dataset. This result is achieved first through our enhanced sequential encoding model, which outperforms the previous best model that employs more complicated network architectures, suggesting that the potential of sequential LSTM-based models have not been fully explored yet in previous work. We further show that by explicitly considering recursive architectures, we achieve additional improvement. Particularly, incorporating syntactic parse information contributes to our best result; it improves the performance even when the parse information is added to an already very strong system.

1 Introduction

Reasoning and inference are central to both human and artificial intelligence. Modeling inference in human language is notoriously challenging but is fundamental to natural language understanding and many applications. As pointed out in [MacCartney and Manning, 2008], *"a necessary (if not sufficient) condition for true natural language understanding is a mastery of open-domain natural language inference."*

Specifically, natural language inference (NLI) is concerned with determining whether a natural-language hypothesis h can be inferred from a natural-language premise p, as depicted in the following example from [MacCartney, 2009], in which the hypothesis is regarded to be entailed from the premise.

p: *Several airlines polled saw costs grow more than expected, even after adjusting for inflation.*

h: *Some of the companies in the poll reported cost increases.*

The most recent years have seen advance in modeling natural language inference (NLI). An important contribution is the availability of the large annotated dataset, the Stanford Natural Language Inference (SNLI) dataset, made available by [Bowman et al., 2015]. The corpus has 570k human-written English sentence pairs manually labeled by multiple human subjects, which can be use to train computers to learn inference knowledge with rather complicated models. In this benchmark dataset, the computer needs to decide if a premise p entails a hypothesis h, if they are contradicting to each other, or have no inference relation.

Neural network models, which often need relatively large-scale annotated data to estimate parameters, have showed to keep improving the state of the art performance [Rocktäschel et al., 2015, Wang and Jiang, 2015, Munkhdalai and Yu, 2016b, Parikh et al., 2016, Cheng et al., 2016, Liu et al., 2016].

While the previous best performing model [Munkhdalai and Yu, 2016b] wires rather complicated network architectures to achieve its state-of-the-art performance, it is not clear if the potential of the more basic sequential models, e.g., Long Short-Term Memory(LSTM) based architectures, have been fully explored. In this paper, we revisit this problem and further explore the potential of the basic LSTM-based sequential encoder. We show that such models can achieve an accuracy of 87.7% on the SNLI benchmark, outperforming the previous best model [Munkhdalai and Yu, 2016b] that employs more complicated network architectures.

Based on this, we further show that by explicitly considering recursive architectures, we achieve additional improvement, increasing the performance to the accuracy of 88.3%. Particularly, incorporating syntactic parse information contributes to our best result; it improves the performance even when the parse information is added onto an already very strong system.

2 Related Work

Early work on natural language inference has been performed on rather small datasets with more conventional methods (refer to [MacCartney, 2009] for a good literature survey). More recently, [Bowman et al., 2015] made available the SNLI dataset with about 570K human annotated sentence pair. They also experimented with simple classification models as well as simple neural networks that encode the premise and hypothesis independently. [Rocktäschel et al., 2015] proposed neural attention-based model on NLI, which captured the attention information. In general, attention based models have shown to be effective in a wide range of tasks, including machine translation [Bahdanau et al., 2014], speech recognition [Chorowski et al., 2015, Chan et al., 2015], image caption [Xu et al., 2015], and text summarization [Rush et al., 2015, Chen et al., 2016], among others. For NLI, the idea allows neural models to pay attention to specific area of the input sentence.

A variety of more advanced networks have been developed since then [Bowman et al., 2016, Vendrov et al., 2015, Mou et al., 2016, Liu et al., 2016, Munkhdalai and Yu, 2016a], and inter-sentence attention-based models [Rocktäschel et al., 2015, Wang and Jiang, 2015, Cheng et al., 2016, Parikh et al., 2016, Munkhdalai and Yu, 2016b]. Among them, more relevant to our work here are the approaches proposed in [Parikh et al., 2016] and [Munkhdalai and Yu, 2016b], which achieved the best performance.

Specifically, [Parikh et al., 2016] propose a relatively simple but very effective decomposable model. More specifically, the model decomposes the NLI problem into subproblems that can be solved separately. On the other hand, the work of [Munkhdalai and Yu, 2016b] wire much more complicated networks that consider sequential LSTM-based encoding, recursive networks, and complicated combination of attention models, which provide about 0.5% gain over the results reported in [Parikh et al., 2016].

It is however not very clear if the potentials of the basic sequential networks have been fully explored. In this paper, we revisit this problem and further explore the the more basic sequential encoding model based with attention. Since the model of [Parikh et al., 2016] achieve one of the best results, we take it as our baseline. We show that enhancing the model from soft alignment, subcomponent inference, and inference composition improves performance by an absolute 1% accuracy, without using more complicated neural net architectures as in [Munkhdalai and Yu, 2016b] but already outperform them. This suggests the potentials of such very basic models had not been fully explored yet and we hope our work shed some light on the future work along this line.

On the other hand, we also employ rich information from syntactic parse and from recursive networks, i.e., the tree-LSTM [Zhu et al., 2015]. In general, exploring syntax together with semantics for NLI is very attractive to us. As pointed out in [Barker and Jacobson, 2007] "the syntax and the semantics work together in tandem", and natural language inference is very likely to involve both. We show that incorporating syntactic parse information is useful even when the parse information is added to the already very strong system. Used with recursive LSTM, it improves the performance to a new state of the art.

3 Hybrid Neural Inference Models

In this section, we present our hybrid inference networks which are composed of the following components: soft alignment, subcomponent inference collection, inference composition, and extension to recursive structures. Figure 1 depicts a very high level view about the components of the models.

We first revisit NLI by exploiting the basic models based on sequential chain-structured LSTM, which, as we will show in our results, can actually surpass the previous best results [Munkhdalai and Yu, 2016b] that instead leverage more complicated architectures. To this end, we take the basic framework introduced in [Parikh et al., 2016], which has achieved a performance comparable to the best [Munkhdalai and Yu, 2016b], but using a rather simple yet very effective architecture.

In our notation, we have two sentences $\mathbf{a} = (\mathbf{a}_1, \ldots, \mathbf{a}_{\ell_a})$ and $\mathbf{b} = (\mathbf{b}_1, \ldots, \mathbf{b}_{\ell_b})$, where $\mathbf{a}$ is the premise and $\mathbf{b}$ the hypothesis. Each $\mathbf{a}_i$ or $\mathbf{b}_j \in \mathbb{R}^l$ is an embedding of l-dimensional vector, which can be initialized with some pre-trained word embedding. The goal is to predict a label y that indicates the logic relationship between $\mathbf{a}$ and $\mathbf{b}$.

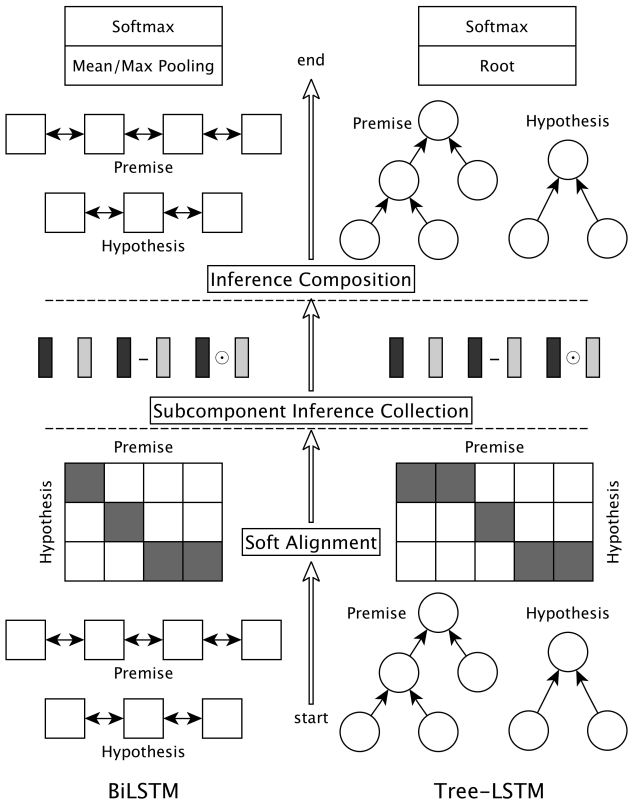

Figure 1: A high level view of our hybrid neural inference networks.

Soft Alignment Most NLI models explore some forms of alignment to associate the relevant subcomponent (e.g., words or phrases) between a premise and hypothesis. This includes early methods motivated from the alignment in conventional automatic machine translation [MacCartney,

2009]. In neural network based models, this is often achieved with soft attention. The work of [Parikh et al., 2016], however, decomposes this process: the word sequence of the premise (or hypothesis) is regarded as a bag-of-word embedding vectors and inter-sentence "alignment" (or attention) is computed individually to softly align each word to the content of hypothesis (or premise, respectively).

While their basic framework is very effective, using pre-trained word embedding by itself does not automatically consider the context round a word in NLI. Parikh et al. did take into account the word order and context information through an optional distance-sensitive intra-sentence attention; in this paper we leverage instead the bidirectional LSTM (BiLSTM) to encoder the context, which shows to play an important role in achieving our best results. The intra-sentence attention used in [Parikh et al., 2016] actually does not further improve over our model.

$$\bar{\mathbf{a}}_i = \text{BiLSTM}_1(\mathbf{a}), \forall i \in [1, \ldots, \ell_a], \tag{1}$$

$$\bar{\mathbf{b}}_j = \text{BiLSTM}_1(\mathbf{b}), \forall j \in [1, \ldots, \ell_b], \tag{2}$$

The BiLSTM hidden vectors above, i.e., $\{\bar{\mathbf{a}}_i\}_{i=1}^{\ell_a}$ and $\{\bar{\mathbf{b}}_j\}_{j=1}^{\ell_b}$, encode a word token and context around it. For completeness, the following equations define a regular chain LSTM.

BiLSTM —> hidden state

Attention framework —> attention weight

$$\mathbf{i}_t = \sigma(\mathbf{W}_i \mathbf{x}_t + \mathbf{U}_i \mathbf{h}_{t-1}), \tag{3}$$

$$\mathbf{f}_t = \sigma(\mathbf{W}_f \mathbf{x}_t + \mathbf{U}_f \mathbf{h}_{t-1}), \tag{4}$$

$$\mathbf{o}_t = \sigma(\mathbf{W}_o \mathbf{x}_t + \mathbf{U}_o \mathbf{h}_{t-1}), \tag{5}$$

$$\mathbf{c}_t = \mathbf{f}_t \odot \mathbf{c}_{t-1} + \mathbf{i}_t \odot \tanh(\mathbf{W}_c \mathbf{x}_t + \mathbf{U}_c \mathbf{h}_{t-1}), \tag{6}$$

$$\mathbf{h}_t = \mathbf{o}_t \odot \tanh(\mathbf{c}_t), \tag{7}$$

where at each word position t, LSTM employs a set of internal vectors: an input gate $\mathbf{i}_t$, a forget gate $\mathbf{f}_t$, an output gate $\mathbf{o}_t$, and a memory cell $\mathbf{c}_t$, to generate a hidden state $\mathbf{h}_t$ (refer to [Hochreiter and Schmidhuber, 1997] for details).

In this work, we found BiLSTM is particularly effective to help build our final NLI models, when applied to both the soft alignment and inference composition. BiLSTM simply runs a forward and a backward LSTM along each input sequence and concatenate two resulting hidden-state at each time (word position), resulting in representation that naturally considers both the past and the future context.

Once BiLSTM hidden state is computed for each word, we simply insert it into the attention framework of [Parikh et al., 2016]. The model in Parikh et al. uses a function $F(\bar{\mathbf{a}}_i)$, i.e., a feed-forward neural network, to map original word representation to calculate attention weight e_{ij} between $\mathbf{a}_i$ in a premise and $\mathbf{b}_j$ in the hypothesis:

$$e_{ij} = \bar{\mathbf{a}}_i^T \bar{\mathbf{b}}_j, \forall i \in [1, \ldots, \ell_a], \forall j \in [1, \ldots, \ell_b]. \tag{8}$$

We instead just simply replace the vector resulted from $F(.)$ with the corresponding BiLSTM vector. We tried the function $F(.)$ but did not find it helps our final best models.

The attention weights e_{ij} are then normalized and used to obtain new vectors as follows:

$$\tilde{\mathbf{a}}_i = \sum_{j=1}^{\ell_b} \frac{\exp(e_{ij})}{\sum_{k=1}^{\ell_b} \exp(e_{ik})} \bar{\mathbf{b}}_j, \forall i \in [1, \ldots, \ell_a], \tag{9}$$

$$\tilde{\mathbf{b}}_j = \sum_{i=1}^{\ell_a} \frac{\exp(e_{ij})}{\sum_{k=1}^{\ell_a} \exp(e_{kj})} \bar{\mathbf{a}}_i, \forall j \in [1, \ldots, \ell_b]. \tag{10}$$

Here $\tilde{\mathbf{a}}_i$, called as *mimic* vector, which is a weighted summation of $\{\mathbf{b}_j\}_{j=1}^{\ell_b}$, is softly aligned to context dependent representation vector $\bar{\mathbf{a}}_i$, and *vice versa* for $\tilde{\mathbf{b}}_j$. Intuitively, when mimic vector $\tilde{\mathbf{a}}_i$ is semantically related to the vector $\mathbf{a}_i$ in the premise *p*, the two vectors will be collected to determine the final inference relation between *p* and *h*.

Subcomponent Inference Collection After alignment is performed, one can collect the inference information from these softly aligned subcomponents of sentences. These subcomponents could correspond to words and their context in a sequential model, phrases and their context in a recursive tree model, or even skip n-grams due to the nature of soft attention. More specifically, the subcomponent pairs between a premise and hypothesis that we use here is the hidden vector resulted from BiLSTM that encodes the current word and its context (e.g., in premise), as well as the inter-sentence soft-aligned content from the other sentence (e.g., the hypothesis). Later, when we incorporated tree-LSTM, the subcomponent could correspond to a node in a tree.

The inference relationship between the subcomponents of the sentence pairs is critical to help determine the overall inference between these two sentences. We therefore carefully model them here. First, we leverage heuristic matching [Mou et al., 2016] between original vectors and mimic vectors. The work of [Mou et al., 2016] used it to compare the premise and hypothesis sentence embeddings, while we extend it to subcomponents of sentences. Complicated models could be used here to model high-order interaction between subcomponent pairs, but that could dramatically increase number of parameters. Here we explore three simple matching heuristics and showed they improve the performance significantly. Particularly we use the concatenation of the two vectors, their difference, and element-wise product. Our experiment shows that these simple heuristics is every effective for NLI, compared with concatenation used in [Parikh et al., 2016]. The minus operation, for example, could help capture contradiction information.

$$\mathbf{m}_a = [\bar{\mathbf{a}}, \tilde{\mathbf{a}}, \bar{\mathbf{a}} - \tilde{\mathbf{a}}, \bar{\mathbf{a}} \odot \tilde{\mathbf{a}}] \tag{11}$$

$$\mathbf{m}_b = [\bar{\mathbf{b}}, \tilde{\mathbf{b}}, \bar{\mathbf{b}} - \tilde{\mathbf{b}}, \bar{\mathbf{b}} \odot \tilde{\mathbf{b}}], \tag{12}$$

Inference Composition To determine the inference relationship between the entire premise sentence and its hypothesis, we explore a composition layer to mix subcomponent inference information.

We first discuss several simple strategies. Then, we introduce tree-LSTM composition. The work presented in [Parikh et al., 2016] separately compare pairs of the representation vector and *mimic* vector using a feed-forward neural network. Here we use another layer of BiLSTM to model the interaction between subcomponent inference collected above.

$$\mathbf{v}_{1,i} = \text{BiLSTM}_2(\mathbf{m}_a), \forall i \in [1, \ldots, \ell_a], \tag{13}$$

$$\mathbf{v}_{2,j} = \text{BiLSTM}_2(\mathbf{m}_b), \forall j \in [1, \ldots, \ell_b]. \tag{14}$$

Then we convert the resulting vectors obtained above to a fixed-length vector and feed it to the final classifier to determine overall inference relationship. We use average pooling and max pooling technology and concatenate these vectors to get fixed length vector $\mathbf{c}$, instead of using summation [Parikh et al., 2016]. We consider that summation is sensitive to the sequence length and is less robust. Our experiments verify average pooling and max pooling have better results than summation. The final fixed length vector $\mathbf{v}$ is calculated as follows:

$$\mathbf{v}_{1,\text{ave}} = \sum_{i=1}^{\ell_a} \mathbf{v}_{1,i}/\ell_a, \tag{15}$$

$$\mathbf{v}_{2,\text{ave}} = \sum_{j=1}^{\ell_b} \mathbf{v}_{1,j}/\ell_b, \tag{16}$$

$$\mathbf{v}_{1,\text{max}} = \max_{i=1}^{\ell_a} \mathbf{v}_{1,i}, \tag{17}$$

$$\mathbf{v}_{2,\text{max}} = \max_{j=1}^{\ell_b} \mathbf{v}_{1,j}, \tag{18}$$

$$\mathbf{v} = [\mathbf{v}_{1,\text{ave}}, \mathbf{v}_{2,\text{ave}}, \mathbf{v}_{1,\text{max}}, \mathbf{v}_{2,\text{max}}]. \tag{19}$$

We put $\mathbf{v}$ into a final multilayer perceptron (MLP) classifier. Specially, the MLP has a hidden layer with *tanh* activation and *softmax* output layer in our experiments. For training, we use multi-class

cross-entropy loss. Again, the composition from subcomponent inference can be performed with recursive composition to incorporate syntactic information, as discussed below.

For simplicity, we call the model we obtain so far as Enhanced BiLSTM Inference Model (**EBIM**) in the remainder of this paper.

Extension to Recursive Structures As discussed earlier in this paper, we are very interested in exploring syntax together with semantics for NLI. As pointed out in [Barker and Jacobson, 2007] "the syntax and the semantics work together in tandem", and natural language inference is very likely to involve both of them. We show in this paper that incorporating syntactic parse information is useful even when the parse information is added to the already very strong system. Incorporated in recursive tree-LSTM, syntactic parse information contributes to achieving our best results.

To explore this, we replace the two BiLSTM (one used in performing soft alignment and one in inference composition) with tree-LSTM. In general, tree-LSTM has recently been proposed to explicitly model tree structures [Zhu et al., 2015]. Specifically, the forward propagation of tree-LSTM is computed as follows in Equation 20– 26, and a node (a memory block) of the network is wired as in Figure 2.

Figure 2 shows the memory block at each node of a recursive tree structure. In general, at each node, an input vector $\mathbf{x}_t$ and the hidden vectors of two children (the left child $\mathbf{h}_{t-1}^L$ and the right $\mathbf{h}_{t-1}^R$) are taken in as the input to calculate their parent node hidden vector h_t.[1] These sources of information are used to configure the four gates as well, i.e., the input gate $\mathbf{i}_t$, output gate $\mathbf{o}_t$, as well as the two forget gates $\mathbf{f}_t^L$ and $\mathbf{f}_t^R$. The memory cell $\mathbf{c}_t$ considers each child's cell vector, $\mathbf{c}_{t-1}^L$ and $\mathbf{c}_{t-1}^R$, which are gated by the left forget gate and right forget gate, respectively.

$$\mathbf{i}_t = \sigma(\mathbf{W}_i\mathbf{x}_t + \mathbf{U}_i^L\mathbf{h}_{t-1}^L + \mathbf{U}_i^R\mathbf{h}_{t-1}^R), \tag{20}$$

$$\mathbf{f}_t^L = \sigma(\mathbf{W}_f\mathbf{x}_t + \mathbf{U}_f^{LL}\mathbf{h}_{t-1}^L + \mathbf{U}_f^{LR}\mathbf{h}_{t-1}^R), \tag{21}$$

$$\mathbf{f}_t^R = \sigma(\mathbf{W}_f\mathbf{x}_t + \mathbf{U}_f^{RL}\mathbf{h}_{t-1}^L + \mathbf{U}_f^{RR}\mathbf{h}_{t-1}^R), \tag{22}$$

$$\mathbf{o}_t = \sigma(\mathbf{W}_o\mathbf{x}_t + \mathbf{U}_o^L\mathbf{h}_{t-1}^L + \mathbf{U}_o^R\mathbf{h}_{t-1}^R), \tag{23}$$

$$\mathbf{u}_t = \tanh(\mathbf{W}_c\mathbf{x}_t + \mathbf{U}_c^L\mathbf{h}_{t-1}^L + \mathbf{U}_c^R\mathbf{h}_{t-1}^R), \tag{24}$$

$$\mathbf{c}_t = \mathbf{f}_t^L \odot \mathbf{c}_{t-1}^L + \mathbf{f}_t^R \odot \mathbf{c}_{t-1}^R + \mathbf{i}_t \odot \mathbf{u}_t, \tag{25}$$

$$\mathbf{h}_t = \mathbf{o}_t \odot \tanh(\mathbf{c}_t), \tag{26}$$

where σ is the sigmoid function, $\odot$ is the element-wise multiplication of two vectors, and all $\mathbf{W} \in \mathbb{R}^{d\times l}$, $\mathbf{U} \in \mathbb{R}^{d\times d}$ are weight matrices to be learned. Note that for brevity, all bias vectors are also omitted.

We use binary tree-LSTM in this paper. The tree structure for each sentence (the premise or hypothesis) is producer by a constituency parser. And as noted in [Zhu et al., 2015], one can always choose to binarize a non-binary tree, and the syntactic information will largely be kept. Binarization can help avoid the need of designing different types of memory blocks for tree nodes with different topology. In addition to using syntactic parse trees, we also borrow the idea from [Munkhdalai and Yu, 2016b] to use full binary trees without encoding syntactic information.

Note that, after replacing BiLSTM with tree-LSTM in our experiment, we feed the root hidden states of tree-LSTM to the classifier, rather than using average pooling or max pooling as in BiLSTM. We found root hidden states have better performance in our experiments. Meanwhile, we ensemble the EBIM and tree-LSTM based model, and it tends to yield a further improvement when there is a significant diversity between the two models. We use a simple strategy that averaging predicted probabilities of two models as a final predicted probabilities.

[1] The non-leaf nodes have no corresponding word, and the initial embedding vector is set to be $\mathbf{0}$.

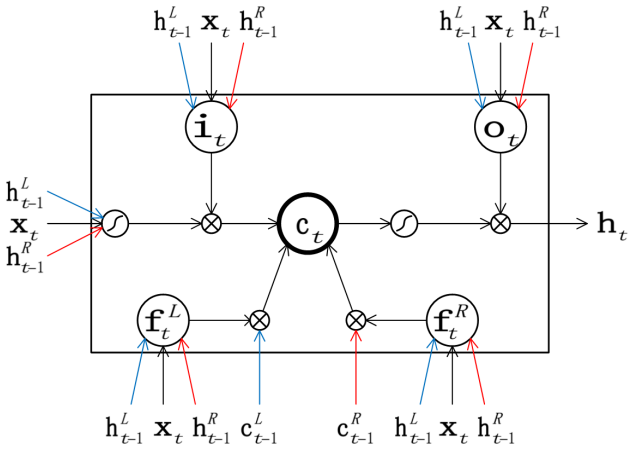

Figure 2: A tree-LSTM memory block.

4 Experiment Set-up

Data The Stanford Natural Language Inference (SNLI) corpus [Bowman et al., 2015] focuses on three basic relationship between a premise and a potential hypothesis: the premise entails the hypothesis (*entailment*), they contradict each other (*contradiction*), or they are not related (*neutral*). The original SNLI corpus contains also "*the other*" category, which includes the sentence pairs lacking consensus among multiple human annotators. Same as in previous work, we remove this category. We used the same split as in [Bowman et al., 2015] and as in other previous work.

The parse trees used in this paper are produced by the Stanford PCFG Parser 3.5.3 [Klein and Manning, 2003] and they are delivered as a part of the SNLI corpus. We use classification accuracy as the evaluation metric, same as in previous work.

Parameters

Training Details To help duplicate our results, we publish our code at http://www.place-holder.com. Below, we list our training details. We use the Adam method [Kingma and Ba, 2014] for optimization. The first momentum is set to be 0.9 and the second 0.999. The initial learning rate is 0.0004 and the batch size is 32. All hidden states of LSTMs, tree-LSTMs, and word embeddings are 300 dimensions.

We use dropout with a dropout rate of 0.5, which is applied to the neural network classifier and word embedding layer. We use pre-trained *300-D Glove 840B* vectors [Pennington et al., 2014] to initialize our word embeddings. Out-of-vocabulary (OOV) words are initialized randomly with Gaussian samples. All vectors including word embedding are updated during training.

5 Results

Overall Performance Table 1 shows the results of different models. The first row is a baseline classifier presented in [Bowman et al., 2015] that considers handcrafted features such as BLEU score of the hypothesis with respect to the premise, the overlapped words, and the length difference between them, etc.

The next group of models (2)-(7) are based on sentence encoding. The model of [Bowman et al., 2016] encodes the premise and hypothesis with two different LSTMs. The model in [Vendrov et al., 2015] uses unsupervised 'skip-thoughts' pre-training in GRU encoders. The approach proposed in [Mou et al., 2016] consider also tree-based CNN to capture sentence-level semantics, while the model of [Bowman et al., 2016] introduces a Stack-augmented Parser-Interpreter Neural Network (SPINN), which combines parsing and interpretation within a single tree-sequence hybrid model. The work of [Liu et al., 2016] use BiLSTM to generate sentence representation, and then replace average pooling with intra-attention. The approach proposed in [Munkhdalai and Yu, 2016a] presents a memory augmented neural network, Neural Semantic Encoders (NSE), to encode sentences.

Table 1: Performance (accuracy) of models on the benchmark data SNLI. Our final model achieves the accuracy of 88.3%, the best result seen so far on SNLI, while our enhanced sequential encoding model attains an accuracy of 87.7%, which also outperform the previous models. The numbers of parameters reported here, as in previous work, do not include word embeddings vectors.

Model	#Para.	Train	Test
(1) Handcrafted features [Bowman et al., 2015]	-	99.7	78.2
(2) 300D LSTM encoders [Bowman et al., 2016]	3.0M	83.9	80.6
(3) 1024D pretrained GRU encoders [Vendrov et al., 2015]	15M	98.8	81.4
(4) 300D tree-based CNN encoders [Mou et al., 2016]	3.5M	83.3	82.1
(5) 300D SPINN-PI encoders [Bowman et al., 2016]	3.7M	89.2	83.2
(6) 600D BiLSTM intra-attention encoders [Liu et al., 2016]	2.8M	84.5	84.2
(7) 300D NSE encoders [Munkhdalai and Yu, 2016a]	3.0M	86.2	84.6
(8) 100D LSTM with attention [Rocktäschel et al., 2015]	250k	85.3	83.5
(9) 300D mLSTM [Wang and Jiang, 2015]	1.9M	92.0	86.1
(10) 450D LSTMN with deep attention fusion [Cheng et al., 2016]	3.4M	88.5	86.3
(11) 200D decomposable attention model [Parikh et al., 2016]	380K	89.5	86.3
(12) Intra-sentence attention + (11) [Parikh et al., 2016]	580K	90.5	86.8
(13) 300D NTI-SLSTM-LSTM [Munkhdalai and Yu, 2016b]	3.2M	88.5	87.3
(14) 600D EBIM	8.6M	92.9	**87.7**
(15) 600D EBIM + 300D Syntactic tree-LSTM	12M	93.0	**88.3**

The next group of methods in the table, model (8)-(13), are inter-sentence attention-based model. The model marked with [Rocktäschel et al., 2015] is LSTMs enforcing so called word-by-word attention. The model in [Wang and Jiang, 2015] extends this idea to explicitly enforce word-by-word matching between the hypothesis and the premise. Long short-term memory-networks (LSTMN) with deep attention fusion [Cheng et al., 2016] link the current word to previous words stored in memory. The work of [Parikh et al., 2016] proposed a decomposable attention model without relying on any word-order information. In general, adding intra-sentence attention yields further improvement, which is not very surprising as it could help align the relevant text spans between premise and hypothesis. The model of [Munkhdalai and Yu, 2016b] extends the framework in [Wang and Jiang, 2015] to a full n-ary tree model and achieved further improvement.

We first show that our EBIM model achieves an accuracy of 87.7%, which has already outperformed previous best model reported in [Munkhdalai and Yu, 2016b] that use more complicated network architectures, including recursive models. It also significantly better than the model of [Parikh et al., 2016] (attaining a 86.8% accuracy). We showed that the potentials of very basic models such as chain LSTM-based models had not been fully explored in the previous work. It could deserve a further exploration on the power of such models.

We ensemble our EBIM model with syntactic tree-LSTM [Zhu et al., 2015] based on binary parse trees, and achieve significant improvement over our best sequential encoding model EBIM, obtaining an accuracy of 88.3%. The syntactic tree-LSTM complement very well with EBIM in achieving the result. Also, according to the results shown in the table, the information brought in from syntactic tree-LSTM generalize well from training to testing: making the performance on train and test data closer.

Ablation Results To investigate the effectiveness the major components of our models, Table 2 provides additional ablation analysis. From the best model, we first replace the syntactic tree-LSTM with the full tree-LSTM without encoding parsing information, similarly to in [Munkhdalai and Yu, 2016b]. More specifically, two adjacent words in a sentence are merged to form a parent node, and

Table 2: Ablation performance of our best models.

Model	#Para.	Train	Test
(15) 600D EBIM + 300D Syn. tree-LSTM	12M	93.0	88.3
(16) 600D EBIM + 300D tree-LSTM	12M	93.5	88.1
(14) 600D EBIM	8.6M	92.9	87.7
(17) w/o average & max pooling	8.3M	93.6	87.1
(18) w/o difference & product	5.4M	93.3	86.7
(19) w/o comparison BiLSTM	2.0M	90.2	86.3
(20) w/o representation BiLSTM	360K	85.5	83.6

this process continues and results in a *full* binary tree, where padding words are inserted to the nodes when there are no enough leaves to form a full tree. Each tree node is implemented with a tree-LSTM block [Zhu et al., 2015]. Table 2 depicts that with this replacement, the performance drops to 88.1%, showing that incorporating syntactic parse information is useful even when the parse information is added to the already very strong system. Again, adding syntactic parsing information seems to make the model generalize well from training to testing: in model (15) the performance for testing is closer to that in training, compared with those in model (16).

From the EBIM model (14), we first remove the final average and max pooling and replace it with summation as used in Parikh et al.. The performance drops to 87.1%. If we further remove the difference and element-wise product from *subcomponent inference collection*, the performance drops to 86.7%. Further removing BiLSTM from *inference composition* and simply using a feed-forward neural network reduce the performance to 86.3%. Finally, we remove BiLSTM from *soft alignment*; the performance significantly dropped to 83.6%.

6 Conclusions

We present several neural network models towards better solving the natural language inference (NLI) problem, which achieve the best results seen so far on the SNLI benchmark. The result is first achieved through our enhanced sequential inference model, which has already outperformed the previous best model that employs more complicated network architectures, suggesting that the potentials of sequential LSTM-based models have not been fully explored yet in previous work. We further show that by explicitly considering recursive architectures, we achieve additional improvement. Particularly, incorporating syntactic parse information contributes to our best result; it improves the performance even when the parse information is added to an already very strong system.

References

Dzmitry Bahdanau, KyungHyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *CoRR*, abs/1409.0473, 2014.

Chris Barker and Pauline Jacobson. *Direct Compositionality*. Oxford University Press, 2007.

Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. A large annotated corpus for learning natural language inference. In *EMNLP*, 2015.

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

William Chan, Navdeep Jaitly, Quoc Viet Le, and Oriol Vinyals. Listen, attend and spell. *CoRR*, abs/1508.01211, 2015.

Qian Chen, Xiao-Dan Zhu, Zhen-Hua Ling, Si Wei, and Hui Jiang. Distraction-based neural networks for modeling document. In *IJCAI*, 2016.

Jianpeng Cheng, Li Dong, and Mirella Lapata. Long short-term memory-networks for machine reading. *arXiv preprint arXiv:1601.06733*, 2016.

Jan Chorowski, Dzmitry Bahdanau, Dmitriy Serdyuk, KyungHyun Cho, and Yoshua Bengio. Attention-based models for speech recognition. In *NIPS*, 2015.

S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural Computation*, (8):1735–1780, 1997.

Diederik P. Kingma and Lei Jimmy Ba. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980, 2014.

Dan Klein and Christopher D. Manning. Accurate unlexicalized parsing. In *ACL*, 2003.

Yang Liu, Chengjie Sun, Lei Lin, and Xiaolong Wang. Learning natural language inference using bidirectional lstm model and inner-attention. *arXiv preprint arXiv:1605.09090*, 2016.

Bill MacCartney. *Natural Language Inference*. PhD thesis, Stanford University, 2009.

Bill MacCartney and Christopher D. Manning. Modeling semantic containment and exclusion in natural language inference. In *Proceedings of the 22Nd International Conference on Computational Linguistics - Volume 1*, COLING '08, pages 521–528, Stroudsburg, PA, USA, 2008. Association for Computational Linguistics. ISBN 978-1-905593-44-6. URL http://dl.acm.org/citation.cfm?id=1599081.1599147.

Lili Mou, Rui Men, Ge Li, Yan Xu, Lu Zhang, Rui Yan, and Zhi Jin. Natural language inference by tree-based convolution and heuristic matching. In *The 54th Annual Meeting of the Association for Computational Linguistics*, page 130, 2016.

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

Tsendsuren Munkhdalai and Hong Yu. Neural tree indexers for text understanding. *arXiv preprint arXiv:1607.04492*, 2016b.

Ankur P Parikh, Oscar Täckström, Dipanjan Das, and Jakob Uszkoreit. A decomposable attention model for natural language inference. *arXiv preprint arXiv:1606.01933*, 2016.

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

Tim Rocktäschel, Edward Grefenstette, Karl Moritz Hermann, Tomás Kociský, and Phil Blunsom. Reasoning about entailment with neural attention. *CoRR*, abs/1509.06664, 2015.

Alexander M. Rush, Sumit Chopra, and Jason Weston. A neural attention model for abstractive sentence summarization. In *EMNLP*, 2015.

Ivan Vendrov, Ryan Kiros, Sanja Fidler, and Raquel Urtasun. Order-embeddings of images and language. *arXiv preprint arXiv:1511.06361*, 2015.

Shuohang Wang and Jing Jiang. Learning natural language inference with lstm. *CoRR*, abs/1512.08849, 2015.

Kelvin Xu, Lei Jimmy Ba, Ryan Kiros, KyungHyun Cho, Aaron C. Courville, Ruslan Salakhutdinov, Richard S. Zemel, and Yoshua Bengio. Show, attend and tell: Neural image caption generation with visual attention. In *ICML*, 2015.

Xiaodan Zhu, Parinaz Sobhani, and Hongyu Guo. Long Short-Term Memory over Recursive Structures. In *Proceedings of International Conference on Machine Learning*, 2015.