Neural Network Models for Paraphrase Identification, Semantic Textual Similarity, Natural Language Inference, and Question Answering

Wuwei Lan and Wei Xu
Department of Computer Science and Engineering
Ohio State University
{lan.105, xu.1265}@osu.edu

Abstract

In this paper, we analyze several neural network designs (and their variations) for sentence pair modeling and compare their performance extensively across eight datasets, including paraphrase identification, semantic textual similarity, natural language inference, and question answering tasks. Although most of these models have claimed state-of-the-art performance, the original papers often reported on only one or two selected datasets. We provide a systematic study and show that (i) encoding contextual information by LSTM and inter-sentence interactions are critical, (ii) Tree-LSTM does not help as much as previously claimed but surprisingly improves performance on Twitter datasets, (iii) the Enhanced Sequential Inference Model (Chen et al., 2017) is the best so far for larger datasets, while the Pairwise Word Interaction Model (He and Lin, 2016) achieves the best performance when less data is available. We release our implementations as an open-source toolkit.

1 Introduction

Sentence pair modeling is a fundamental technique underlying many NLP tasks, including the following:

- Semantic Textual Similarity (STS), which measures the degree of equivalence in the underlying semantics of paired snippets of text (Agirre et al., 2016).
- Paraphrase Identification (PI), which identifies whether two sentences express the same meaning (Dolan and Brockett, 2005; Xu et al., 2015).
- Natural Language Inference (NLI), also known as recognizing textual entailment (RTE), which concerns whether a hypothesis can be inferred from a premise, requiring understanding of the semantic similarity between the hypothesis and the premise (Dagan et al., 2006; Bowman et al., 2015).
- Question Answering (QA), which can be approximated as ranking candidate answer sentences or phrases based on their similarity to the original question (Yang et al., 2015).
- Machine Comprehension (MC), which requires sentence matching between a passage and a question, pointing out the text region that contains the answer. (Rajpurkar et al., 2016).

Traditionally, researchers had to develop different methods specific for each task. Now neural networks can perform all the above tasks with the same architecture by training end to end. Various neural models (He and Lin, 2016; Chen et al., 2017; Parikh et al., 2016; Wieting et al., 2016; Tomar et al., 2017; Wang et al., 2017; Shen et al., 2017a; Yin et al., 2016) have declared state-of-the-art results for sentence pair modeling tasks; however, they were carefully designed and evaluated on selected (often one or two) datasets that can demonstrate the superiority of the model. The research questions are as follows: Do they perform well on other tasks and datasets? How much performance gain is due to certain system design choices and hyperparameter optimizations?

Proceedings of the 27th International Conference on Computational Linguistics, pages 3890–3902
Santa Fe, New Mexico, USA, August 20-26, 2018.

To answer these questions and better understand different network designs, we systematically analyze and compare the state-of-the-art neural models across multiple tasks and multiple domains. Namely, we implement five models and their variations on the same PyTorch platform: InferSent model (Conneau et al., 2017), Shortcut-stacked Sentence Encoder Model (Nie and Bansal, 2017), Pairwise Word Interaction Model (He and Lin, 2016), Decomposable Attention Model (Parikh et al., 2016), and Enhanced Sequential Inference Model (Chen et al., 2017). They are representative of the two most common approaches: **sentence encoding models** that learn vector representations of individual sentences and then calculate the semantic relationship between sentences based on vector distance and **sentence pair interaction models** that use some sorts of word alignment mechanisms (e.g., attention) then aggregate inter-sentence interactions. We focus on identifying important network designs and present a series of findings with quantitative measurements and in-depth analyses, including (i) incorporating inter-sentence interactions is critical; (ii) Tree-LSTM does not help as much as previously claimed but surprisingly improves performance on Twitter data; (iii) Enhanced Sequential Inference Model has the most consistent high performance for larger datasets, while Pairwise Word Interaction Model performs better on smaller datasets and Shortcut-Stacked Sentence Encoder Model is the best performaning model on the Quora corpus. We release our implementations as a toolkit to the research community.[1]

2 General Framework for Sentence Pair Modeling

Various neural networks have been proposed for sentence pair modeling, all of which fall into two types of approaches. The sentence encoding approach encodes each sentence into a fixed-length vector and then computes sentence similarity directly. The model of this type has advantages in the simplicity of the network design and generalization to other NLP tasks. The sentence pair interaction approach takes word alignment and interactions between the sentence pair into account and often show better performance when trained on in-domain data. Here we outline the two types of neural networks under the same general framework:

- **The Input Embedding Layer** takes vector representations of words as input, where pretrained word embeddings are most commonly used, e.g. GloVe (Pennington et al., 2014) or Word2vec (Mikolov et al., 2013). Some work used embeddings specially trained on phrase or sentence pairs that are paraphrases (Wieting and Gimpel, 2017; Tomar et al., 2017); some used subword embeddings, which showed improvement on social media data (Lan and Xu, 2018).
- **The Context Encoding Layer** incorporates word context and sequence order into modeling for better vector representation. This layer often uses CNN (He et al., 2015), LSTM (Chen et al., 2017), recursive neural network (Socher et al., 2011), or highway network (Gong et al., 2017). The sentence encoding type of model will stop at this step, and directly use the encoded vectors to compute the semantic similarity through vector distances and/or the output classification layer.
- **The Interaction and Attention Layer** calculates word pair (or n-gram pair) interactions using the outputs of the encoding layer. This is the key component for the interaction-aggregation type of model. In the PWIM model (He and Lin, 2016), the interactions are calculated by cosine similarity, Euclidean distance, and the dot product of the vectors. Various models put different weights on different interactions, primarily simulating the word alignment between two sentences. The alignment information is useful for sentence pair modeling because the semantic relation between two sentences depends largely on the relations of aligned chunks as shown in the SemEval-2016 task of interpretable semantic textual similarity (Agirre et al., 2016).
- **The Output Classification Layer** adapts CNN or MLP to extract semantic-level features on the attentive alignment and applies softmax function to predict probability for each class.

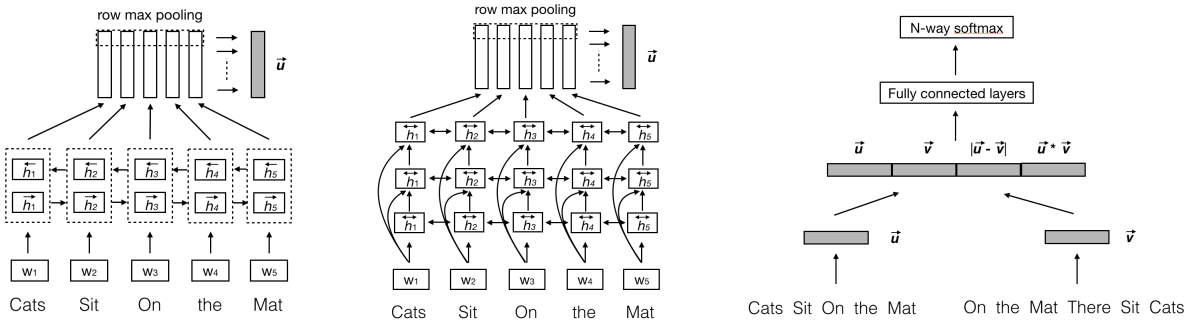
[1]The code is available on the authors' homepages and GitHub: `https://github.com/lanwuwei/SPM_toolkit`

3 Representative Models for Sentence Pair Modeling

Table 1 gives a summary of typical models for sentence pair modeling in recent years. In particular, we investigate five models in depth: two are representative of the sentence encoding type of model, and three are representative of the interaction-aggregation type of model. These models have reported state-or-the-art results with varied architecture design (this section) and implementation details (Section 4.2).

Models	Sentence Encoder	Interaction and Attention	Aggregation and Classification
(Shen et al., 2017b)	Directional self-attention network	-	MLP
(Choi et al., 2017)	Gumbel Tree-LSTM	-	MLP
(Wieting and Gimpel, 2017)	Gated recurrent average network	-	MLP
SSE (Nie and Bansal, 2017)	Shortcut-stacked BiLSTM	-	MLP
(He et al., 2015)	CNN	multi-perspective matching	pooling + MLP
(Rocktäschel et al., 2016)	LSTM	word-by-word neural attention	MLP
(Liu et al., 2016)	LSTM	coupled LSTMs	dynamic pooling + MLP
(Yin et al., 2016)	CNN	attention matrix	logistic regression
DecAtt (Parikh et al., 2016)	-	dot product + soft alignment	summation + MLP
PWIM (He and Lin, 2016)	BiLSTM	cosine, Euclidean, dot product + hard alignment	CNN + MLP
(Wang and Jiang, 2017)	LSTM encodes both context and attention	word-by-word neural attention	MLP
ESIM (Chen et al., 2017)	BiLSTM (Tree-LSTM) before and after attention	dot product + soft alignment	average and max pooling + MLP
(Wang et al., 2017)	BiLSTM	multi-perspective matching	BiLSTM + MLP
(Shen et al., 2017a)	BiLSTM + intra-attention	soft alignment + orthogonal decomposition	MLP
(Ghaeini et al., 2018)	dependent reading BiLSTM	dot product + soft alignment	average and max pooling+MLP

Table 1: Summary of representative neural models for sentence pair modeling. The upper half contains sentence encoding models, and the lower half contains sentence pair interaction models.



(a) InferSent (b) SSE (c) Classification Layer

Figure 1: **Sentence encoding models** focus on learning vector representations of individual sentences and then calculate the semantic relationship between sentences based on vector distance.

3.1 The Bi-LSTM Max-pooling Network (InferSent)

We choose the simple Bi-LSTM max-pooling network from InferSent (Conneau et al., 2017):

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

$$\boldsymbol{v} = max(\overleftrightarrow{\boldsymbol{h}}_1, \overleftrightarrow{\boldsymbol{h}}_2, ..., \overleftrightarrow{\boldsymbol{h}}_n) \tag{2}$$

where $\overleftrightarrow{\boldsymbol{h}}_i$ represents the concatenation of hidden states in both directons. It has shown better transfer learning capabilities than several other sentence embedding models, including SkipThought (Kiros et al., 2015) and FastSent (Hill et al., 2016), when trained on the natural language inference datasets.

3.2 The Shortcut-Stacked Sentence Encoder Model (SSE)

The Shortcut-Stacked Sentence Encoder model (Nie and Bansal, 2017) is a sentence-based embedding model, which enhances multi-layer Bi-LSTM with skip connection to avoid training error accumulation, and calculates each layer as follows:

$$\overleftrightarrow{\boldsymbol{h}}_i^k = BiLSTM(\boldsymbol{x}_i^k, \overleftrightarrow{\boldsymbol{h}}_{i-1}^k) \tag{3}$$

$$\boldsymbol{x}_i^1 = \boldsymbol{w}_i \quad (k=1), \qquad \boldsymbol{x}_i^k = [\boldsymbol{w}_i, \overleftrightarrow{\boldsymbol{h}}_i^{k-1}, \overleftrightarrow{\boldsymbol{h}}_i^{k-2}, ..., \overleftrightarrow{\boldsymbol{h}}_i^1] \quad (k>1) \tag{4}$$

$$\boldsymbol{v} = max(\overleftrightarrow{\boldsymbol{h}}_1^m, \overleftrightarrow{\boldsymbol{h}}_2^m, ..., \overleftrightarrow{\boldsymbol{h}}_n^m) \tag{5}$$

where $\boldsymbol{x}_i^k$ is the input of the kth Bi-LSTM layer at time step i, which is the combination of outputs from all previous layers, $\overleftrightarrow{\boldsymbol{h}}_i^k$ represents the hidden state of the kth Bi-LSTM layer in both directions. The final sentence embedding $\boldsymbol{v}$ is the row-based max pooling over the output of the last Bi-LSTM layer, where n denotes the number of words within a sentence and m is the number of Bi-LSTM layers ($m = 3$ in SSE).

3.3 The Pairwise Word Interaction Model (PWIM)

In the Pairwise Word Interaction model (He and Lin, 2016), each word vector $\boldsymbol{w}_i$ is encoded with context through forward and backward LSTMs: $\overrightarrow{\boldsymbol{h}}_i = LSTM^f(\boldsymbol{w}_i, \overrightarrow{\boldsymbol{h}}_{i-1})$ and $\overleftarrow{\boldsymbol{h}}_i = LSTM^b(\boldsymbol{w}_i, \overleftarrow{\boldsymbol{h}}_{i+1})$. For every word pair $(\boldsymbol{w}_i^a, \boldsymbol{w}_j^b)$ across sentences, the model directly calculates word pair interactions using cosine similarity, Euclidean distance, and dot product over the outputs of the encoding layer:

$$D(\overrightarrow{\boldsymbol{h}}_i, \overrightarrow{\boldsymbol{h}}_j) = [cos(\overrightarrow{\boldsymbol{h}}_i, \overrightarrow{\boldsymbol{h}}_j), \|\overrightarrow{\boldsymbol{h}}_i - \overrightarrow{\boldsymbol{h}}_j\|, \overrightarrow{\boldsymbol{h}}_i \cdot \overrightarrow{\boldsymbol{h}}_j] \tag{6}$$

The above equation not only applies to forward hidden state $\overrightarrow{\boldsymbol{h}}_i$ and backward hidden state $\overleftarrow{\boldsymbol{h}}_i$, but also to the concatenation $\overleftrightarrow{\boldsymbol{h}}_i = [\overrightarrow{\boldsymbol{h}}_i, \overleftarrow{\boldsymbol{h}}_i]$ and summation $\boldsymbol{h}_i^+ = \overrightarrow{\boldsymbol{h}}_i + \overleftarrow{\boldsymbol{h}}_i$, resulting in a tensor $\mathbf{D}^{13\times|sent1|\times|sent2|}$ after padding one extra bias term. A "hard" attention is applied to the interaction tensor to build word alignment: selecting the most related word pairs and increasing the corresponding weights by 10 times. Then a 19-layer deep CNN is applied to aggregate the word interaction features for final classification.

3.4 The Decomposable Attention Model (DecAtt)

The Decomposable Attention model (Parikh et al., 2016) is one of the earliest models to introduce attention-based alignment for sentence pair modeling, and it achieved state-of-the-art results on the SNLI dataset with about an order of magnitude fewer parameters than other models (see more in Table 5) without relying on word order information. It computes the word pair interaction between $\boldsymbol{w}_i^a$ and $\boldsymbol{w}_j^b$ (from input sentences s_a and s_b, each with m and n words, respectively) as $e_{ij} = F(\boldsymbol{w}_i^a)^T F(\boldsymbol{w}_j^b)$, where F is a feedforward network; then alignment is determined as follows:

$$\boldsymbol{\beta}_i = \sum_{j=1}^{n} \frac{exp(e_{ij})}{\sum_{k=1}^{n} exp(e_{ik})} \boldsymbol{w}_j^b \qquad\qquad \boldsymbol{\alpha}_j = \sum_{i=1}^{m} \frac{exp(e_{ij})}{\sum_{k=1}^{m} exp(e_{kj})} \boldsymbol{w}_i^a \tag{7}$$

where $\boldsymbol{\beta}_i$ is the soft alignment between $\boldsymbol{w}_i^a$ and subphrases $\boldsymbol{w}_j^b$ in sentence s_b, and vice versa for $\boldsymbol{\alpha}_j$. The aligned phrases are fed into another feedforward network G: $\boldsymbol{v}_i^a = G([\boldsymbol{w}_i^a; \boldsymbol{\beta}_i])$ and $\boldsymbol{v}_j^b = G([\boldsymbol{w}_j^b; \boldsymbol{\alpha}_j])$ to generate sets $\{\boldsymbol{v}_i^a\}$ and $\{\boldsymbol{v}_j^b\}$, which are aggregated by summation and then concatenated together for classification.

3.5 The Enhanced Sequential Inference Model (ESIM)

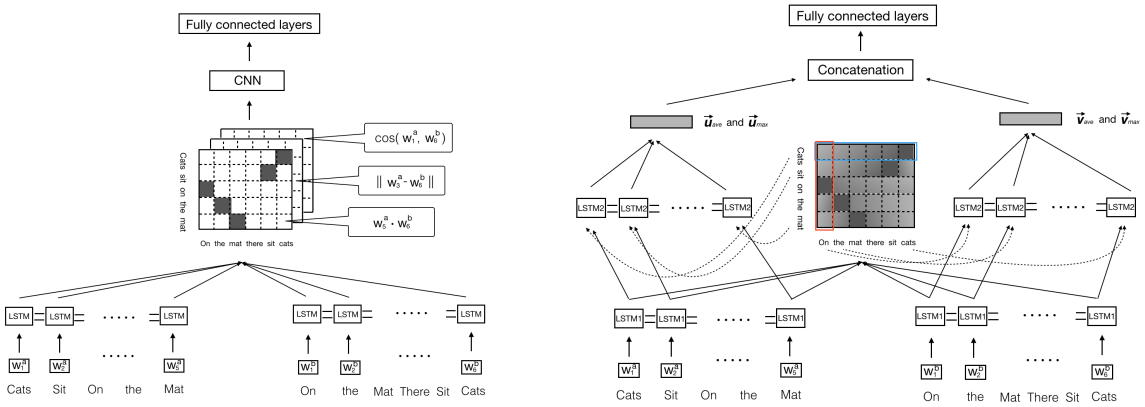
The Enhanced Sequential Inference Model (Chen et al., 2017) is closely related to the DecAtt model, but it differs in a few aspects. First, Chen et al. (2017) demonstrated that using Bi-LSTM to encode sequential contexts is important for performance improvement. They used the concatenation $\overline{\boldsymbol{w}}_i = \overleftrightarrow{\boldsymbol{h}}_i = [\overrightarrow{\boldsymbol{h}}_i, \overleftarrow{\boldsymbol{h}}_i]$ of both directions as in the PWIM model. The word alignment $\boldsymbol{\beta}_i$ and $\boldsymbol{\alpha}_j$ between $\overline{\boldsymbol{w}}^a$ and $\overline{\boldsymbol{w}}^b$ are calculated the same way as in DecAtt. Second, they showed the competitive performance of recursive architecture with constituency parsing, which complements with sequential LSTM. The feedforward function G in DecAtt is replaced with Tree-LSTM:

$$\boldsymbol{v}_i^a = TreeLSTM([\overline{\boldsymbol{w}}_i^a; \boldsymbol{\beta}_i; \overline{\boldsymbol{w}}_i^a - \boldsymbol{\beta}_i; \overline{\boldsymbol{w}}_i^a \odot \boldsymbol{\beta}_i]) \tag{8}$$

$$\boldsymbol{v}_j^b = TreeLSTM([\overline{\boldsymbol{w}}_j^b; \boldsymbol{\alpha}_j; \overline{\boldsymbol{w}}_j^b - \boldsymbol{\alpha}_j; \overline{\boldsymbol{w}}_j^b \odot \boldsymbol{\alpha}_j]) \tag{9}$$

Third, instead of using summation in aggregation, ESIM adapts the average and max pooling and concatenation $\boldsymbol{v} = [\boldsymbol{v}_{ave}^a; \boldsymbol{v}_{max}^a; \boldsymbol{v}_{ave}^b; \boldsymbol{v}_{max}^b]$ before passing through multi-layer perceptron (MLP) for classification:

$$\boldsymbol{v}_{ave}^a = \sum_{i=1}^{m} \frac{\boldsymbol{v}_i^a}{m}, \qquad \boldsymbol{v}_{max}^a = \max_{i=1}^{m} \boldsymbol{v}_i^a, \qquad \boldsymbol{v}_{ave}^b = \sum_{j=1}^{n} \frac{\boldsymbol{v}_j^b}{n}, \qquad \boldsymbol{v}_{max}^b = \max_{j=1}^{n} \boldsymbol{v}_j^b \tag{10}$$



(a) PWIM

(b) ESIM$_{seq}$ (DecAtt is similar and simpler.)

Figure 2: **Sentence pair interaction models** use different word alignment mechanisms before aggregation.

4 Experiments and Analysis

4.1 Datasets

We conducted sentence pair modeling experiments on eight popular datasets: two NLI datasets, three PI datasets, one STS dataset and two QA datasets. Table 2 gives a comparison of these datasets:

- **SNLI** (Bowman et al., 2015) contains 570k hypotheses written by crowdsourcing workers given the premises. It focuses on three semantic relations: the premise entails the hypothesis (entailment), they contradict each other (contradiction), or they are unrelated (neutral).
- **Multi-NLI** (Williams et al., 2017) extends the SNLI corpus to multiple genres of written and spoken texts with 433k sentence pairs.

Dataset	Size		Example and Label	
SNLI	train	550,152	s_a: *Two men on bicycles competing in a race.*	**entailment**
	dev	10,000	s_b: *Men are riding bicycles on the street.*	neutral
	test	10,000		contradict
Multi-NLI	train	392,703	s_a: *The Old One always comforted Ca'daan, except today.*	entailment
	dev	20,000	s_b: *Ca'daan knew the Old One very well.*	**neutral**
	test	20,000		contradict
Quora	train	384,348	s_a: *What should I do to avoid sleeping in class?*	**paraphrase**
	dev	10,000	s_b: *How do I not sleep in a boring class?*	non-paraphrase
	test	10,000		
Twitter-URL	train	42,200	s_a: *Letter warned Wells Fargo of "widespread" fraud in 2007.*	**paraphrase**
	dev	-	s_b: *Letters suggest Wells Fargo scandal started earlier.*	non-paraphrase
	test	9,324		
PIT-2015	train	11,530	s_a: *Ezekiel Ansah w the 3D shades Popped out lens*	**paraphrase**
	dev	4,142	s_b: *Ezekiel Ansah was wearing lens less 3D glasses*	non-paraphrase
	test	838		
STS-2014	train	7,592	s_a: *Then perhaps we could have avoided a catastrophe.*	score $[0, 5]$
	dev	-	s_b: *Then we might have been able to avoid a disaster.*	**4.6**
	test	3,750		
WikiQA	train	8,672	s_a: *How much is 1 tablespoon of water?*	**true**
	dev	1,130	s_b: *In Australia one tablespoon (measurement unit) is 20 mL.*	false
	test	2,351		
TrecQA	train	53,417	s_a: *Who was Lincoln's Secretary of State?*	true
	dev	1,148	s_b: *William Seward*	**false**
	test	1,517		

Table 2: Basic statistics and examples of different datasets for sentence pair modeling tasks.

- **Quora** (Iyer et al., 2017) contains 400k question pairs collected from the Quora website. This dataset has balanced positive and negative labels indicating whether the questions are duplicated or not.
- **Twitter-URL** (Lan et al., 2017) includes 50k sentence pairs collected from tweets that share the same URL of news articles. This dataset contains both formal and informal language.
- **PIT-2015** (Xu et al., 2015) comes from SemEval-2015 and was collected from tweets under the same trending topic. It contains naturally occurred (i.e. written by independent Twitter users spontaneously) paraphrases and non-paraphrases with varied topics and language styles.
- **STS-2014** (Agirre et al., 2014) is from SemEval-2014, constructed from image descriptions, news headlines, tweet news, discussion forums, and OntoNotes (Hovy et al., 2006).
- **WikiQA** (Yang et al., 2015) is an open-domain question-answering dataset. Following He and Lin (2016), questions without correct candidate answer sentences are excluded, and answer sentences are truncated to 40 tokens, resulting in 12k question-answer pairs for our experiments.
- **TrecQA** (Wang et al., 2007) is an answer selection task of 56k question-answer pairs and created in Text Retrieval Conferences (TREC). For both WikiQA and TrecQA datasets, the best answer is selected according to the semantic relatedness with the question.

4.2 Implementation Details

We implement all the models with the same PyTorch framework.[2][3] Below, we summarize the implementation details that are key for reproducing results for each model:

- **SSE:** This model can converge very fast, for example, 2 or 3 epochs for the SNLI dataset. We control the convergence speed by updating the learning rate for each epoch: specifically, $lr = \frac{1}{2^{\frac{epoch\_i}{2}}} * init\_lr$, where $init\_lr$ is the initial learning rate and $epoch\_i$ is the index of current epoch.

[2]InferSent and SSE have open-source PyTorch implementations by the original authors, for which we reused part of the code.
[3]Our code is available at: `https://github.com/lanwuwei/SPM_toolkit`

- **DecAtt:** It is important to use gradient clipping for this model: for each gradient update, we check the L2 norm of all the gradient values, if it is greater than a threshold b, we scale the gradient by a factor $\alpha = b/L2\_norm$. Another useful procedure is to assemble batches of sentences with similar length.
- **ESIM:** Similar but different from DecAtt, ESIM batches sentences with varied length and uses masks to filter out padding information. In order to batch the parse trees within Tree-LSTM recursion, we follow Bowman et al.'s (2016) procedure that converts tree structures into the linear sequential structure of a shift reduce parser. Two additional masks are used for producing left and right children of a tree node.
- **PWIM:** The cosine and Euclidean distances used in the word interaction layer have smaller values for similar vectors while dot products have larger values. The performance increases if we add a negative sign to make all the vector similarity measurements behave consistently.

4.3 Analysis

4.3.1 Re-implementation Results vs. Previously Reported Results

Table 3 and 4 show the results reported in the original papers and the replicated results with our implementation. We use accuracy, F1 score, Pearson's r, Mean Average Precision (MAP), and Mean Reciprocal Rank (MRR) for evaluation on different datasets following the literature. Our reproduced results are slightly lower than the original results by $0.5 \sim 1.5$ points on accuracy. We suspect the following potential reasons: (i) less extensive hyperparameter tuning for each individual dataset; (ii) only one run with random seeding to report results; and (iii) use of different neural network toolkits: for example, the original ESIM model was implemented with Theano, and PWIM model was in Torch.

4.3.2 Effects of Model Components

Herein, we examine the main components that account for performance in sentence pair modeling.

How important is LSTM encoded context information for sentence pair modeling?
Regarding DecAtt, Parikh et al. (2016) mentioned that "intra-sentence attention is optional"; they can achieve competitive results without considering context information. However, not surprisingly, our experiments consistently show that encoding sequential context information with LSTM is critical. Compared to DecAtt, ESIM shows better performance on every dataset (see Table 4 and Figure 3). The main difference between ESIM and DecAtt that contributes to performance improvement, we found, is the use of Bi-LSTM and Tree-LSTM for sentence encoding, rather than the different choices of aggregation functions.

Why does Tree-LSTM help with Twitter data?
Chen et al. (2017) offered a simple combination (ESIM$_{seq+tree}$) by averaging the prediction probabilities of two ESIM variants that use sequential Bi-LSTM and Tree-LSTM respectively, and suggested "parsing information complements very well with ESIM and further improves the performance". However, we found that adding Tree-LSTM only helps slightly or not at all for most datasets, but it helps noticably with the two Twitter paraphrase datasets. We hypothesize the reason is that these two datasets come from real-world tweets which often contain extraneous text fragments, in contrast to SNLI and other datasets that have sentences written by crowdsourcing workers. For example, the segment "*ever wondered ,*" in the sentence pair *ever wondered , why your recorded #voice sounds weird to you?* and *why do our recorded voices sound so weird to us?* introduces a disruptive context into the Bi-LSTM encoder, while Tree-LSTM can put it in a less important position after constituency parsing.

How important is attentive interaction for sentence pair modeling? Why does SSE excel on Quora?
Both ESIM and DecAtt (Eq. 7) calculate an attention-based soft alignment between a sentence pair, which was also proposed in (Rocktäschel et al., 2016) and (Wang and Jiang, 2017) for sentence pair modeling, whereas PWIM utilizes a hard attention mechanism. Both attention strategies are critical for model performance. In PWIM model (He and Lin, 2016), we observed a $1\sim2$ point performance drop after

Model	SNLI	Multi-NLI	Quora	Twitter-URL	PIT-2015	STS-2014	WikiQA	TrecQA
	Acc	Acc_m/Acc_um	Acc	F1	F1	r	MAP/MRR	MAP/MRR
InferSent	0.845	-/-	-	-	-	0.700[5]	-	-
SSE	0.860	0.746/0.736	-	-	-	-	-	-
DecAtt	0.863	-	0.865[3]	-	-	-	-	-
ESIM$_{tree}$	0.878	-	-	-	-	-	-	-
ESIM$_{seq}$	0.880	0.723/0.721[4]	-	-	-	-	-	-
ESIM$_{seq+tree}$	0.886	-	-	-	-	-	-	-
PWIM	-	-	-	0.749	0.667	0.767	0.709/0.723	0.759/0.822

Table 3: Reported results from original papers, which are mostly limited to a few datasets. For the Multi-NLI dataset, Acc_m represents testing accuracy for the matched genre and Acc_um for the unmatched genre.

Model	SNLI	Multi-NLI	Quora	Twitter-URL	PIT-2015	STS-2014	WikiQA	TrecQA
	Acc	Acc_m/Acc_um	Acc	F1	F1	r	MAP/MRR	MAP/MRR
InferSent	0.846	0.705/0.703	0.866	0.746	0.451	0.715	0.287/0.287	0.521/0.559
SSE	0.855	0.740/0.734	**0.878**	0.650	0.422	0.378	0.624/0.638	0.628/0.670
DecAtt	0.856	0.719/0.713	0.845	0.652	0.430	0.317	0.603/0.619	0.660/0.712
ESIM$_{tree}$	0.864	0.736/0.727	0.755	0.740	0.447	0.493	0.618/0.633	0.698/0.734
ESIM$_{seq}$	0.870	0.752/0.738	0.850	0.748	0.520	0.602	0.652/0.664	**0.771/0.795**
ESIM$_{seq+tree}$	**0.871**	**0.753/0.748**	0.854	0.759	0.538	0.589	0.647/0.658	0.749/0.768
PWIM	0.822	0.722/0.716	0.834	**0.761**	**0.656**	**0.743**	**0.706/0.723**	0.739/0.795

Table 4: Replicated results with our reimplementation in PyTorch across multiple tasks and datasets. The best result in each dataset is denoted by a **bold** typeface, and the second best is denoted by an underline.

removing the hard attention, 0$\sim$3 point performance drop and $\sim$25% training time reduction after removing the 19-layer CNN aggregation. Likely without even the authors of SSE knowing, the SSE model performs extraordinarily well on the Quora corpus, perhaps because Quora contains many sentence pairs with less complicated inter-sentence interactions (e.g., many identical words in the two sentences) and incorrect ground truth labels (e.g., *What is your biggest regret in life?* and *What's the biggest regret you've had in life?* are labeled as non-duplicate questions by mistake).

4.3.3 Learning Curves and Training Time

Figure 3 shows the learning curves. The DecAtt model converges quickly and performs well on large NLI datasets due to its design simplicity. PWIM is the slowest model (see time comparison in Table 5) but shows very strong performance on semantic similarity and paraphrase identification datasets. ESIM and SSE keep a good balance between training time and performance.

[3]This number was reported in (Tomar et al., 2017) by co-authors of DecAtt (Parikh et al., 2016).
[4]This number was reproduced by Williams et al. (2017).
[5]This number was generated by InferSent traind on SNLI and Multi-NLI datasets.

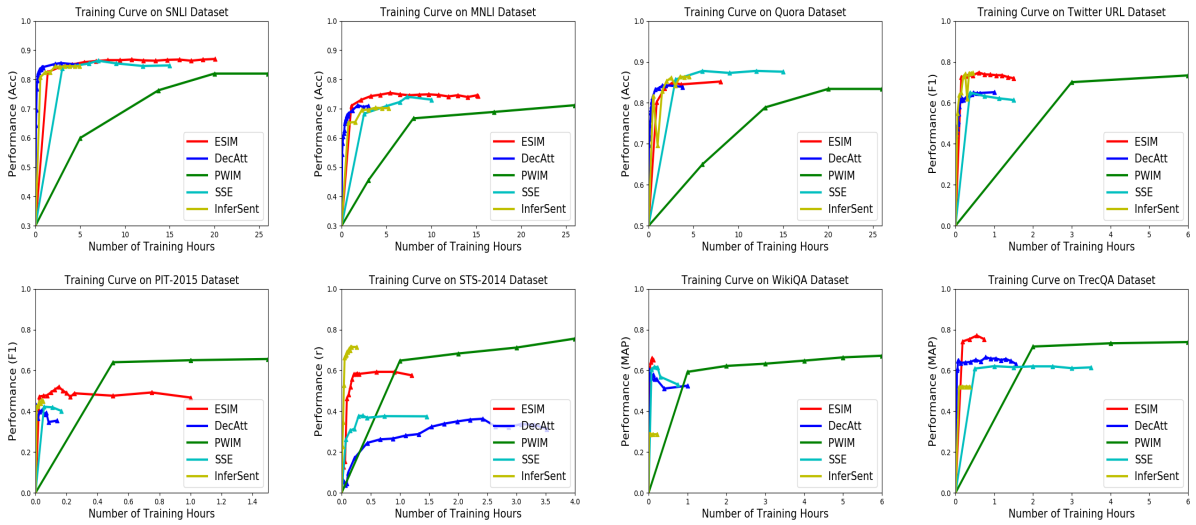


Figure 3: Training curves of ESIM, DecAtt, PWIM, SSE and InferSent models on eight datasets.

	InferSent	**SSE**	**DecAtt**	**ESIM$_{seq}$**	**ESIM$_{tree}$**	**PWIM**
Number of parameters	47M	140M	380K	4.3M	7.7M	2.2M
Avg epoch time (seconds) / sentence pair	0.005	0.032	0.0006	0.013	0.016	0.60
Ratio compared to DecAtt model	$\times$8	$\times$53	1	$\times$22	$\times$26	$\times$1000

Table 5: Average training time per sentence pair in the Twitter-URL dataset (similar time for other datasets).

4.3.4 Effects of Training Data Size

As shown in Figure 4, we experimented with different training sizes of the largest SNLI dataset. All the models show improved performance as we increase the training size. ESIM and SSE have very similar trends and clearly outperform PWIM on the SNLI dataset. DecAtt shows a performance jump when the training size exceeds a threshold.

4.3.5 Categorical Performance Comparison

We conducted an in-depth analysis of model performance on the Multi-domain NLI dataset based on different categories: text genre, sentence pair overlap, and sentence length. As shown in Table 7, all models have comparable performance between matched genre and unmatched genre. Sentence length and overlap turn out to be two important factors – the longer the sentences and the fewer tokens in common, the more challenging it is to determine their semantic relationship. These phenomena shared by the state-of-the-art systems reflect their similar design framework which is symmetric at processing both sentences in the pair, while question answering and natural language inference tasks are directional (Ghaeini et al., 2018). How to incorporate asymmetry into model design will be worth more exploration in future research.

4.3.6 Transfer Learning Experiments

In addition to the cross-domain study (Table 7), we conducted transfer learning experiments on three paraphrase identification datasets (Table 6). The most noteworthy phenomenon is that the SSE model performs better on Twitter-URL and PIT-2015 when trained on the large out-of-domain Quora data than the small in-domain training data. Two likely reasons are: 1) the SSE model with over 29 million parameters is data hungry and 2) SSE model is a sentence encoding model, which generalizes better across domains/tasks than sentence pair interaction models. Sentence pair interaction models may encounter difficulties on Quora, which contains sentence pairs with the highest word overlap (51.5%) among all datasets and often causes

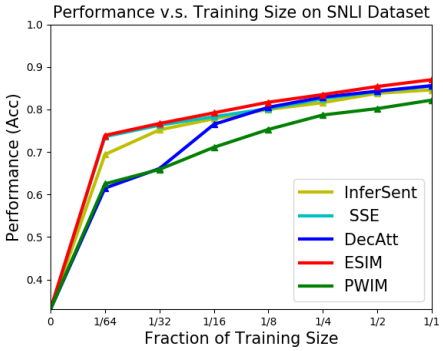


Figure 4: Performance vs. training size (log scale in x-axis) on SNLI dataset.

Models	Quora	URL	PIT	train/test
	trained on Quora			on PIT
InferSent	0.866	0.528	0.394	0.451
SSE	0.878	0.681	0.594	0.422
DecAtt	0.845	0.649	0.497	0.430
ESIM$_{seq}$	0.850	0.643	0.501	0.520
PWIM	0.835	0.601	0.518	0.656
	trained on URL			
InferSent	0.703	0.746	0.535	0.451
SSE	0.630	0.650	0.477	0.422
DecAtt	0.632	0.652	0.450	0.430
ESIM$_{seq}$	0.641	0.748	0.511	0.520
PWIM	0.678	0.761	0.634	0.656

Table 6: Transfer learning experiments for paraphrase identification task.

	Category	#Examples	InferSent	SSE	DecAtt	ESIM$_{seq}$	PWIM
Matched Genre	Fiction	1973	0.703	0.727	0.706	0.742	0.707
	Government	1945	0.753	0.746	0.743	0.790	0.751
	Slate	1955	0.653	0.670	0.671	0.697	0.670
	Telephone	1966	0.718	0.728	0.717	0.753	0.709
	Travel	1976	0.705	0.701	0.733	0.752	0.714
Mismatched Genre	9/11	1974	0.685	0.710	0.699	0.737	0.711
	Face-to-face	1974	0.713	0.729	0.720	0.761	0.710
	Letters	1977	0.734	0.757	0.754	0.775	0.757
	OUP	1961	0.698	0.715	0.719	0.759	0.710
	Verbatim	1946	0.691	0.701	0.709	0.725	0.713
Overlap	>60%	488	0.756	0.795	0.805	0.842	0.811
	30% ∼ 60%	3225	0.740	0.751	0.745	0.769	0.743
	<30%	6102	0.685	0.689	0.691	0.727	0.682
Length	>20 tokens	3730	0.692	0.676	0.685	0.731	0.694
	10∼20 tokens	3673	0.712	0.725	0.721	0.753	0.720
	<10 tokens	2412	0.721	0.758	0.748	0.762	0.724

Table 7: Categorical performance (accuracy) on Multi-NLI dataset. Overlap is the percentage of shared tokens between two sentences. Length is calculated based on the number of tokens of the longer sentence.

the interaction patterns to focus on a few key words that differ. In contrast, the Twitter-URL dataset has the lowest overlap (23.0%) with a semantic relationship that is mainly based on the intention of the tweets.

5 Conclusion

We analyzed five different neural models (and their variations) for sentence pair modeling and conducted a series of experiments with eight representative datasets for different NLP tasks. We quantified the importance of the LSTM encoder and attentive alignment for inter-sentence interaction, as well as the transfer learning ability of sentence encoding based models. We showed that the SNLI corpus of over 550k sentence pairs cannot saturate the learning curve. We systematically compared the strengths and weaknesses of different network designs and provided insights for future work.

Acknowledgements

We thank Ohio Supercomputer Center (Center, 2012) for computing resources. This work was supported in part by NSF CRII award (RI-1755898) and DARPA through the ARO (W911NF-17-C-0095). The content of the information in this document does not necessarily reflect the position or the policy of the U.S. Government, and no official endorsement should be inferred.

References

Eneko Agirre, Carmen Banea, Claire Cardie, Daniel Cer, Mona Diab, Aitor Gonzalez-Agirre, Weiwei Guo, Rada Mihalcea, German Rigau, and Janyce Wiebe. 2014. Semeval-2014 task 10: Multilingual semantic textual similarity. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*.

Eneko Agirre, Aitor Gonzalez-Agirre, Inigo Lopez-Gazpio, Montse Maritxalar, German Rigau, and Larraitz Uria. 2016. Semeval-2016 task 2: Interpretable semantic textual similarity. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval)*.

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)*.

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. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (ACL)*.

Ohio Supercomputer Center. 2012. Oakley supercomputer. `http://osc.edu/ark:/19495/hpc0cvqn`.

Qian Chen, Xiaodan Zhu, Zhenhua 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 (ACL)*.

Jihun Choi, Kang Min Yoo, and Sang-goo Lee. 2017. Unsupervised learning of task-specific tree structures with tree-LSTMs. *arXiv preprint arXiv:1707.02786*.

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 (EMNLP)*.

Ido Dagan, Oren Glickman, and Bernardo Magnini. 2006. The PASCAL recognising textual entailment challenge. In *Proceedings of the First International Conference on Machine Learning Challenges: Evaluating Predictive Uncertainty Visual Object Classification, and Recognizing Textual Entailment*.

William B Dolan and Chris Brockett. 2005. Automatically constructing a corpus of sentential paraphrases. In *Proceedings of the Third International Workshop on Paraphrasing (IWP)*.

Reza Ghaeini, Sadid A Hasan, Vivek Datla, Joey Liu, Kathy Lee, Ashequl Qadir, Yuan Ling, Aaditya Prakash, Xiaoli Z Fern, and Oladimeji Farri. 2018. DR-BiLSTM: Dependent reading bidirectional LSTM for natural language inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*.

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

Hua He and Jimmy Lin. 2016. Pairwise word interaction modeling with deep neural networks for semantic similarity measurement. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*.

Hua He, Kevin Gimpel, and Jimmy Lin. 2015. Multi-perspective sentence similarity modeling with convolutional neural networks. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Felix Hill, Kyunghyun Cho, and Anna Korhonen. 2016. Learning distributed representations of sentences from unlabelled data. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*.

Eduard Hovy, Mitchell Marcus, Martha Palmer, Lance Ramshaw, and Ralph Weischedel. 2006. Ontonotes: The 90% solution. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the ACL (NAACL)*.

Shankar Iyer, Nikhil Dandekar, and Kornl Csernai. 2017. First Quora Dataset Release: Question Pairs. In *https://data.quora.com/First-Quora-Dataset-Release-Question-Pairs*.

Ryan Kiros, Yukun Zhu, Ruslan R Salakhutdinov, Richard Zemel, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 2015. Skip-thought vectors. In *Advances in Neural Information Processing Systems (NIPS)*.

Wuwei Lan and Wei Xu. 2018. The importance of subword embeddings in sentence pair modeling. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*.

Wuwei Lan, Siyu Qiu, Hua He, and Wei Xu. 2017. A continuously growing dataset of sentential paraphrases. In *Proceedings of The 2017 Conference on Empirical Methods on Natural Language Processing (EMNLP)*.

Pengfei Liu, Xipeng Qiu, and Xuanjing Huang. 2016. Modelling interaction of sentence pair with coupled-LSTMs. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems (NIPS)*.

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*.

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 (EMNLP)*.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

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 (EMNLP)*.

Tim Rocktäschel, Edward Grefenstette, Karl Moritz Hermann, Tomáš Kočiskỳ, and Phil Blunsom. 2016. Reasoning about entailment with neural attention. In *Proceedings of the International Conference on Learning Representations (ICLR)*.

Gehui Shen, Yunlun Yang, and Zhi-Hong Deng. 2017a. Inter-weighted alignment network for sentence pair modeling. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Tao Shen, Tianyi Zhou, Guodong Long, Jing Jiang, Shirui Pan, and Chengqi Zhang. 2017b. Disan: Directional self-attention network for RNN/CNN-free language understanding. In *Proceedings of the Association for the Advancement of Artificial Intelligence (AAAI)*.

Richard Socher, Cliff C Lin, Chris Manning, and Andrew Y Ng. 2011. Parsing natural scenes and natural language with recursive neural networks. In *Proceedings of the 28th International Conference on Machine Learning (ICML)*.

Gaurav Singh Tomar, Thyago Duque, Oscar Täckström, Jakob Uszkoreit, and Dipanjan Das. 2017. Neural paraphrase identification of questions with noisy pretraining. In *Proceedings of the First Workshop on Subword and Character Level Models in NLP*.

Shuohang Wang and Jing Jiang. 2017. A compare-aggregate model for matching text sequences. In *Proceedings of the International Conference on Learning Representations (ICLR)*.

Mengqiu Wang, Noah A Smith, and Teruko Mitamura. 2007. What is the Jeopardy model? A quasi-synchronous grammar for qa. In *Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*.

Zhiguo Wang, Wael Hamza, and Radu Florian. 2017. Bilateral multi-perspective matching for natural language sentences. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence (IJCAI)*.

John Wieting and Kevin Gimpel. 2017. Revisiting recurrent networks for paraphrastic sentence embeddings. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (ACL)*.

John Wieting, Mohit Bansal, Kevin Gimpel, and Karen Livescu. 2016. Towards universal paraphrastic sentence embeddings. In *Proceedings of the 4th International Conference on Learning Representations (ICLR)*.

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

Wei Xu, Chris Callison-Burch, and William B. Dolan. 2015. SemEval-2015 Task 1: Paraphrase and semantic similarity in Twitter (PIT). In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval)*.

Yi Yang, Wen-tau Yih, and Christopher Meek. 2015. WikiQA: A challenge dataset for open-domain question answering. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Wenpeng Yin, Hinrich Schtze, Bing Xiang, and Bowen Zhou. 2016. ABCNN: Attention-based convolutional neural network for modeling sentence pairs. *Transactions of the Association for Computational Linguistics (TACL)*.