A Compare-Propagate Architecture with Alignment Factorization for Natural Language Inference

Yi Tay
Nanyang Technological University, Singapore
ytay017@e.ntu.edu.sg

Luu Anh Tuan
Institute for Infocomm Research
at.luu@i2r.a-star.edu.sg

Siu Cheung Hui
Nanyang Technological, University, Singapore
asschui@ntu.edu.sg

Abstract

This paper presents a new deep learning architecture for Natural Language Inference (NLI). Firstly, we introduce a new compare-propagate architecture where alignments pairs are compared and then propagated to upper layers for enhanced representation learning. Secondly, we adopt novel factorization layers for efficient compression of alignment vectors into scalar valued features, which are then be used to augment the base word representations. The design of our approach is aimed to be conceptually simple, compact and yet powerful. We conduct experiments on three popular benchmarks, SNLI, MultiNLI and SciTail, achieving state-of-the-art performance on all. A lightweight parameterization of our model also enjoys a $\approx$ 300% reduction in parameter size compared to the ESIM and DIIN, while maintaining competitive performance. Visual analysis shows that our propagated features are highly interpretable, opening new avenues to explainability in neural NLI models.

1 Introduction

Natural Language Inference (NLI) is a pivotal and fundamental task in language understanding and artificial intelligence. More specifically, given a premise and hypothesis, NLI aims to detect whether the latter *entails* or *contradicts* the former. As such, NLI is also commonly known as Recognizing Textual Entailment (RTE). NLI is known to be a significantly challenging task for machines with success often dependent on a wide repertoire of reasoning techniques.

In recent years we observe a steep improvement in NLI systems, largely contributed by the release of the largest publicly available corpus for NLI - the Stanford Natural Language Inference (SNLI) corpus (Bowman et al., 2015) which comprises $570K$ hand labeled sentence pairs. This improved the feasibility of training complex neural models, given the fact that neural models often require a relatively large amount of training data.

Highly competitive neural models for NLI are mostly based on soft-attention alignments, popularized by (Parikh et al., 2016). The key idea is to learn an alignment of sub-phrases in both sentences and learn to compare the relationship between them. Standard feed-forward neural networks are commonly used to model similarity between aligned (decomposed) sub-phrases and then aggregated into the final prediction layers.

Alignment between sentences have become a staple technique in NLI research and many recent state-of-the-art models such as the Enhanced Sequential Inference Model (ESIM) (Chen et al., 2017a) also incorporate his alignment strategy. The difference here is that ESIM considers a non-parameterized comparison scheme, i.e., *concatenating* the subtraction and element-wise product of aligned sub-phrases, along with two original sub-phrases, into the final comparison vector. A bidirectional LSTM is then used to aggregate the compared alignment vectors.

This paper presents a new neural model for NLI. There are several new novel components in our work. Firstly, we propose a *compare-propagate* architecture where alignment features are propagated to upper layers (such as an RNN-based encoder) for enhancing representation learning. Notably, this is different from the compare-aggregate paradigm that aggregates the compared alignment vectors for prediction (Wang and Jiang, 2016a; Parikh et al., 2016; Chen et al., 2017a). To the best of our knowledge, we are the first to adopt

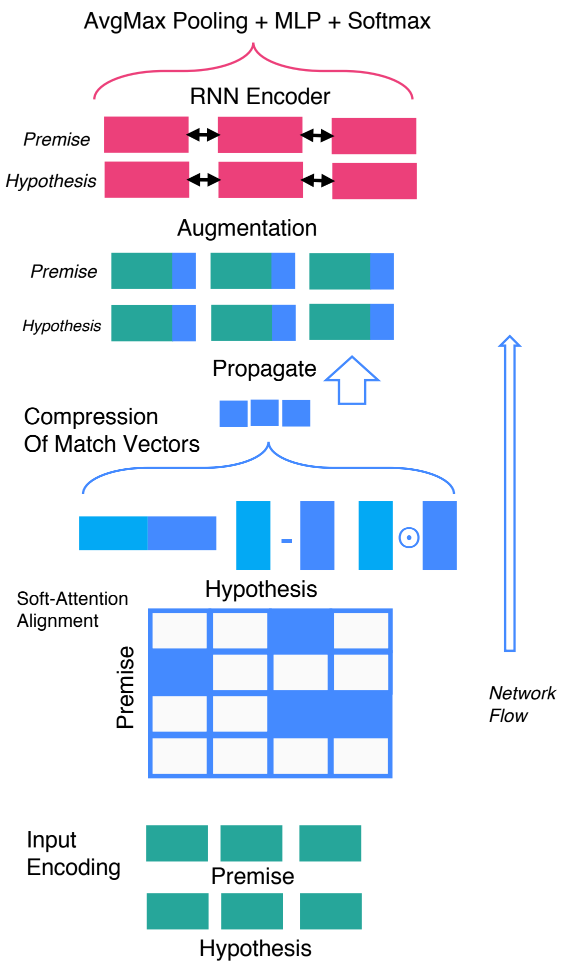

Figure 1: High level overview of our proposed architecture (*best viewed in color*). Alignment vectors are compressed and then propagated to upper representation learning layers (RNN encoders). Intra-attention is omitted in this diagram due to the lack of space.

such a paradigm. Secondly, in order to achieve an efficient propagation of alignment features, we propose alignment factorization layers to reduce each alignment vectors to a single scalar valued feature. Each scalar valued feature is used to augment the base word representation, allowing the subsequent RNN encoder layers to benefit from not only global but cross sentence information. Figure 1 depicts a high-level overview of our proposed model architecture.

There are several major advantages to our proposed architecture. Firstly, our model is relatively compact, i.e., to avoid large alignment (or match) vectors being propagated across the network, we compress alignment feature vectors and augment them to word representations instead. As a result, our model is more parameter efficient compared to the ESIM since the width of the middle layers of the network is now much smaller.

Secondly, the compare-propagate paradigm enables *highly interpretable* features since each alignment pair is compressed to a scalar. Previous models such as the ESIM use subtractive operations on alignment vectors, edging on the intuition that these vectors represent contradiction. Our model is capable of visually demonstrating this phenomena. As such, our design choice enables a *new* way of deriving insight from neural NLI models.

Thirdly, our alignment factorization layers are expressive and powerful, combining ideas from standard machine learning literature (Rendle, 2010) with modern neural NLI models. The factorization layer tries to decompose the alignment vector (constructed from the variations $a-b$, $a \odot b$ and $[a;b]$), learning higher-order feature interactions between each compared alignment. In other words, it models the second-order (pairwise) interactions between *each* feature in *every* alignment vector using factorized parameters, allowing more expressive comparison to be made over traditional feed-forward neural networks (FFN). The effectiveness of the factorization alignment over alternative baselines such as feed-forward neural networks is confirmed by early experiments.

1.1 Our Contributions

The major contributions of this work are summarized as follows:

- We introduce a *Compare-Propagate* architecture for NLI. The key idea is to use the myriad of generated comparison vectors for augmentation of the base word representation instead of simply aggregating them for prediction. Subsequently, a standard compositional encoder can then be used to learn representations from the augmented word representations. We show that we are able to derive meaningful insight from visualizing these augmented features.

- For the first time, we adopt expressive factorization layers to model the relationships between soft-aligned sub-phrases of sentence pairs. Empirical experiments confirm the effectiveness of this new layer over standard fully connected layers.

- Overall, we propose a new neural model - CAFE (**C**ompare-propagate **A**lignment-

Factorized **E**ncoders) for NLI. Our model achieves state-of-the-art performance on SNLI, MultiNLI and the new SciTail dataset, outperforming existing state-of-the-art models such as the ESIM. Ablation studies confirm the effectiveness of each proposed component in our model.

2 Related Work

Natural language inference (or textual entailment recognition) is a long standing problem in NLP research, typically carried out on smaller datasets using traditional methods (Maccartney, 2009; Dagan et al., 2006; MacCartney and Manning, 2008; Iftene and Balahur-Dobrescu, 2007).

The relatively recent creation of $570K$ human annotated sentence pairs (Bowman et al., 2015) have spurred on many recent works that use neural networks for NLI. Many advanced neural architectures have been proposed for the NLI task, with most exploiting some variant of neural attention which learns to pay attention to important segments in a sentence (Parikh et al., 2016; Chen et al., 2017a; Wang and Jiang, 2016b; Rocktäschel et al., 2015; Yu and Munkhdalai, 2017).

Amongst the myriad of neural architectures proposed for NLI, the ESIM (Chen et al., 2017a) model is one of the best performing models. The ESIM, primarily motivated by soft subphrase alignment in (Parikh et al., 2016), learns alignments between BiLSTM encoded representations and aggregates them with another BiLSTM layer. The authors also propose the usage of subtractive composition, claiming that this helps model contradictions amongst alignments.

Compare-Aggregate models are also highly popular in NLI tasks. While this term was coined by (Wang and Jiang, 2016a), many prior NLI models follow this design (Wang et al., 2017; Parikh et al., 2016; Gong et al., 2017; Chen et al., 2017a). The key idea is to aggregate matching features and pass them through a dense layer for prediction. (Wang et al., 2017) proposed BiMPM, which adopts multi-perspective cosine matching across sequence pairs. (Wang and Jiang, 2016a) proposed a one-way attention and convolutional aggregation layer. (Gong et al., 2017) learns representations with highway layers and adopts ResNet for learning features over an interaction matrix.

There a several other notable models for NLI. For instance, models that leverage directional self-attention (Shen et al., 2017) or Gumbel-Softmax (Choi et al., 2017). DGEM is a graph based attention model which was proposed together with a new entailment challenge dataset, SciTail (Khot et al., 2018).

Our work compares and compresses alignment pairs using factorization layers which leverages the rich history of standard machine learning literature. Our factorization layers incorporates highly expressive factorization machines (FMs) (Rendle, 2010) into neural NLI models. In standard machine learning tasks, FMs remain a very competitive choice for learning feature interactions (Xiao et al., 2017) for both standard classification and regression problems. Intuitively, FMs are adept at handling data sparsity (typically interactions) by using factorized parameters to approximate a feature matching matrix. This makes it suitable in our model architecture since interaction between subphrase alignment pairs is typically very sparse[1] as well.

3 Our Proposed Model

In this section, we provide a layer-by-layer description of our model architecture. Our model accepts two sentences as an input, i.e., P (premise) and H (hypothesis).

3.1 Input Encoding Layer

This layer aims to learn a k-dimensional representation for each word. Following (Gong et al., 2017), we learn feature-rich word representations by concatenating word embeddings, character features and syntactic (part-of-speech tag) features. Character representations are learned using a convolutional encoder with max pooling function and is commonly used in many relevant literature (Wang et al., 2017; Chen et al., 2017b).

3.1.1 Highway Encoding Layers

Subsequently, we pass each concatenated word vector into a two layer highway network (Srivastava et al., 2015) in order to learn a k-dimensional representation. Highway networks are gated projection layers which learn adaptively control how much information is being carried to the next layer. Our strategy is similar to (Parikh et al., 2016) which trains the projection layer in place of tuning the embedding matrix. The usage of high-

[1]Word-word interactions are already extremely sparse, let alone phrasal interactions.

way layers over standard projection layers is empirically motivated. However, an intuition would be that the gates in this layer adapt to learn the relative importance of each word to the NLI task.

Let $H(.)$ and $T(.)$ be single layered affine transforms with ReLU and sigmoid activation functions respectively. A single highway network layer is defined as:

$$y = H(x, W_H) \cdot T(x, W_T) + C \cdot x \quad (1)$$

where $C = (1 - T(x, W_T))$ and $W_H, W_T \in \mathbb{R}^{r \times d}$ Notably, the dimensions of the affine transform might be different from the size of the input vector. In this case, an additional nonlinear transform is used to project x to the same dimensionality. The output of this layer is $\bar{P} \in \mathbb{R}^{k \times \ell_P}$ (premise) and $\bar{H} \in \mathbb{R}^{k \times \ell_H}$ (hypothesis), with each word converted to a r-dimensional vector. ,

3.2 Inter-Attention Alignment Layer

This layer learns an alignment of sub-phrases between $\bar{P}$ and $\bar{H}$. Let $F(.)$ be a standard projection layer with ReLU activation function. The alignment matrix of two sequences is defined as follows:

$$e_{ij} = F(\bar{p}_i)^\top \cdot F(\bar{h}_j) \quad (2)$$

where $E \in \mathbb{R}^{\ell_p \times \ell_h}$ and $\bar{p}_i$, $\bar{h}_j$ are the i-th and j-th word in the premise and hypothesis respectively.

$$\beta_i := \sum_{j=1}^{\ell_p} \frac{exp(e_{ij})}{\sum_{k=1}^{\ell_p} exp(e_{ik})} \bar{p}_j \quad (3)$$

$$\alpha_j := \sum_{i=1}^{\ell_h} \frac{exp(e_{ij})}{\sum_{k=1}^{\ell_h} exp(e_{kj})} \bar{h}_i \quad (4)$$

where β_i is the sub-phrase in $\bar{P}$ that is softly aligned to h_i. Intuitively, β_i is a weighted sum across $\{p_j\}_{j=1}^{\ell_p}$, selecting the most relevant parts of $\bar{P}$ to represent h_i.

3.3 Intra-Attention Alignment Layer

This layer learns a *self-alignment* of sentences and is applied to both $\bar{P}$ and $\bar{H}$ independently. For the sake of brevity, let $\bar{S}$ represent either $\bar{P}$ or $\bar{H}$, the intra-attention alignment is computed as:

$$s'_i := \sum_{j=1}^{\ell_p} \frac{exp(f_{ij})}{\sum_{k=1}^{\ell_p} exp(f_{ik})} \bar{s}_j \quad (5)$$

where $f_{ij} = G(\bar{s}_i)^\top \cdot G(\bar{s}_j)$ and $G(.)$ is a nonlinear projection layer with ReLU activation function. The intra-attention layer models similarity of each word with respect to the entire sentence, capturing long distance dependencies and 'global' context of the entire sentence.

3.4 Alignment Factorization Layers

This layer aims to learn a scalar valued feature for each comparison between aligned sub-phrases. Firstly, we introduce our factorization operation, which lives at the core of our neural model.

3.4.1 Factorization Operation

Given an input vector x, the factorization operation (Rendle, 2010) is defined as:

$$L(x) = w_0 + \sum_{i=1}^{n} w_i \, x_i \quad (6)$$

$$P(x) = \sum_{i=1}^{n} \sum_{j=i+1}^{n} \langle v_i, v_j \rangle \, x_i \, x_j \quad (7)$$

$$F_{fm}(x) = L(x) + P(x) \quad (8)$$

where $F_{fm}(x)$ is a scalar valued output. $\langle .; . \rangle$ is the dot product between two vectors and w_0 is the global bias. The parameters of this layer are $w_0 \in \mathbb{R}, w \in \mathbb{R}^r$ and $v \in \mathbb{R}^{r \times k}$. Intuitively, $L(x)$ represents a linear regression layer while $P(x)$ learns pairwise feature interactions by trying to factorize the feature interaction matrix.

3.4.2 Alignment Factorization

This layer compares the alignment between inter-attention aligned representations, i.e., (β_i, h_i) and (α_j, p_j). Let (a, b) represent an alignment pair, we apply the following operations:

$$y_c = F_{fm}([a; b]) \quad (9)$$

$$y_s = F_{fm}(a - b) \quad (10)$$

$$y_m = F_{fm}(a \odot b) \quad (11)$$

where $y_c, y_s, y_m \in \mathbb{R}$ and $Z(.)$ is the factorization operation, $[.; .]$ is the concatenation operator and $\odot$ is the element-wise multiplication. The intuition of modeling subtraction is targeted at capturing contradiction. However, instead of simply concatenating the extra comparison vectors, we compress them using factorization layers. Finally, for each alignment pair, we obtain three scalar-valued features which map precisely to a word in the sequence.

Next, for each sequence, we also apply alignment factorization on the intra-aligned sentences. Let (s, s') represent an *intra-aligned* pair from either the premise or hypothesis, we compute the following operations:

$$v_c = F_{fm}([s; s']) \tag{12}$$
$$v_s = F_{fm}(s - s') \tag{13}$$
$$v_m = F_{fm}(s \odot s') \tag{14}$$

where $v_c, v_s, v_m \in \mathbb{R}$ and $Z(.)$ is the factorization operation. Applying alignment factorization to intra-aligned representations produces another three scalar-valued features which are mapped to each word in the sequence. Note that each of the *six* factorization operations has its own parameters but shares them amongst all words in the sentences.

3.5 Propagation and Augmentation

Finally, the *six* factorized features are then aggregated[2] via concatenation to form a final feature vector that is propagated to upper representation learning layers via augmentation of the word representation $\bar{P}$ or $\bar{H}$.

$$u_i = [s_i; f^i_{intra}; f^i_{inter}] \tag{15}$$

where s_i is i-th word in $\bar{P}$ or $\bar{H}$., f^i_{intra} and f^i_{inter} are the intra-aligned $[v_c; v_s; v_m]$ and inter-aligned $[y_c; y_s; y_m]$ features for the i-th word in the sequence respectively. Intuitively, f^i_{intra} augments each word with global knowledge of the sentence and f^i_{inter} augments each word with cross-sentence knowledge via inter-attention.

3.6 Sequential Encoder Layer

For each sentence, the augmented word representations $u_1, u_2, \dots u_\ell$ is then passed into a sequential encoder layer. We adopt a standard vanilla LSTM encoder.

$$h_i = LSTM(u, i), \forall i \in [1, \dots \ell] \tag{16}$$

where ℓ represents the maximum length of the sequence. Notably, the parameters of the LSTM are *siamese* in nature, sharing weights between both premise and hypothesis. We do not use a bidirectional LSTM encoder, as we found that it did not lead to any improvements on the held-out set.

[2]Following (Parikh et al., 2016), we may also concatenate the intra-aligned vector to u_i which we found to have speed up convergence.

A logical explanation would be because our word representations are already augmented with global (intra-attention) information. As such, modeling in the reverse direction is unnecessary, resulting in some computational savings.

3.6.1 Pooling Layer

Next, to learn an overall representation of each sentence, we apply a pooling function across all hidden outputs of the sequential encoder. The pooling function is a concatenation of temporal max and average (avg) pooling.

$$x = [\max([h_1, \cdots h_\ell]); avg([h_1, \cdots h_\ell])] \tag{17}$$

where x is a final $2k$-dimensional representation of the sentence (premise or hypothesis). We also experimented with *sum* and *avg* standalone poolings and found *sum* pooling to be relatively competitive.

3.7 Prediction Layers

Finally, given a fixed dimensional representation of the premise x_p and hypothesis x_h, we pass their concatenation into a two-layer h-dimensional highway network. Since the highway network has been already defined earlier, we omit the technical details. The final prediction layers of our model is computed as follows:

$$y_{out} = H_2(H_1([x_p; x_h; x_p \odot x_h; x_p - x_h])) \tag{18}$$

where $H_1(.), H_2(.)$ are highway network layers with ReLU activation. The output is then passed into a final linear softmax layer.

$$y_{pred} = softmax(W_F \cdot y_{out} + b_F) \tag{19}$$

where $W_F \in \mathbb{R}^{h \times 3}$ and $b_F \in \mathbb{R}^3$. The network is then trained using standard multi-class cross entropy loss with L2 regularization.

4 Experiments

4.1 Experimental Setup

To ascertain the effectiveness of our models, we use the SNLI (Bowman et al., 2015) and MultiNLI (Williams et al., 2017) benchmarks which are standard and highly competitive benchmarks for the NLI task. We also include the newly released SciTail dataset (Khot et al., 2018) which is a binary entailment classification task constructed from science questions. Notably, SciTail is known to be a difficult dataset for NLI, made evident by the low accuracy scores even though it is binary in nature.

- **SNLI** - We compare against competitors across three settings. The first setting disallows cross sentence attention. In the second setting, cross sentence is allowed. The first two setting only comprises single models. The last setting is a comparison between model ensembles. Though we compare with many other models (Wang and Jiang, 2016b; Rocktäschel et al., 2015; Parikh et al., 2016; Yu and Munkhdalai, 2017; Sha et al., 2016), the key SoTA competitors on this dataset are the BiMPM (Wang et al., 2017), ESIM (Chen et al., 2017a) and DIIN (Gong et al., 2017).

- **MultiNLI** - We compare on two test sets (*matched* and *mismatched*) which represent in-domain and out-domain performance. The main competitor on this dataset is the ESIM model, a powerful state-of-the-art SNLI baseline. We also compare with ESIM + read (Weissenborn, 2017).

- **SciTail** - This dataset only has one official setting. We compare against the reported results of ESIM (Chen et al., 2017a) and DecompAtt (Parikh et al., 2016) in the original paper. We also compare with DGEM, the new model proposed in (Khot et al., 2018).

Across all experiments and in the spirit of fair comparison, we only compare with works that (1) do not use extra training data and (2) do not use external resources (such as external knowledge bases etc.).

4.2 Implementation Details

We implement our model in Tensorflow (Abadi et al., 2015) and train them on Nvidia P100 GPUs. For the first setting (without cross sentence attention), we remove both the cross (inter) attention layers to abide to the rules of this setting. We use the Adam optimizer (Kingma and Ba, 2014) with an initial learning rate of 0.0003. Default L2 regularization is set to 10^{-6}. Dropout with a keep probability of 0.8 is applied after each fully-connected, recurrent or highway layer. The batch size is tuned amongst $\{128, 256, 512\}$. The number of latent factors k for the factorization layers is tuned amongst $\{5, 10, 50, 100, 150\}$. The size of the hidden layers of the highway layers are set to 300. All parameters are initialized with xavier initialization. Word embeddings are pre-loaded with $300d$ GloVE embeddings (Pennington et al., 2014) and fixed during training. Sequence length are padded to batch-wise maximums. The batch order are (randomly) sorted within buckets following (Parikh et al., 2016).

,

Method	$\|\theta\|$	Train	Test
Encoders (w/o Cross Sentence Attention)			
300D LSTM Encoder	3.0M	83.9	80.6
600D Gated BiLSTM + intra-att	12M	90.5	85.5
300D Gumbel LSTM TreeLSTM	2.9M	91.2	85.6
300D DISAN	2.4M	91.1	85.6
300D Residual Stacked Encoders	9.7M	89.8	85.7
600D Gumbel TreeLSTM	10M	93.1	**86.0**
300D CAFE (w/o Inter-Attention)	3.7M	87.3	<u>85.9</u>
Cross Sentence Attention (Single Models)			
100D LSTM with attention	250K	85.3	83.5
300D mLSTM	1.9M	92.0	86.1
200D DecompAtt	380K	89.5	86.3
200D DecompAtt + Intra-Att	580K	90.5	86.8
300D NTI-SLSTM-LSTM	3.2M	88.5	87.3
300D re-read LSTM	2.0M	90.7	87.5
BiMPM	1.6M	90.9	87.5
448D DIIN	4.4M	91.2	88.0
600D ESIM	4.3M	92.6	88.0
150D CAFE (Sum+2x200D MLP)	750K	88.2	87.7
200D CAFE (Sum+2x400D MLP)	1.4M	89.4	88.1
300D CAFE (Sum+2x600D MLP)	3.5M	89.2	<u>88.3</u>
300D CAFE (AvgMax+300D HN)	4.7M	89.8	**88.5**
Cross Sentence Attention (Ensemble Models)			
600D ESIM + 300D TreeLSTM	7.7M	93.5	88.6
BiMPM	6.4M	93.2	88.8
448D DIIN	17.0M	92.3	88.9
300D CAFE (Ensemble)	17.5M	92.5	**89.3**

Table 1: Performance comparison of all models on SNLI benchmark.

4.3 Results on SNLI

Table 1 reports our results on the SNLI benchmark. On the cross sentence (single model setting), the performance of our proposed CAFE model is extremely competitive. We report the test accuracies of CAFE at different extent of parameterization, varying the size of the LSTM encoder, width of the pre-softmax hidden layers and final pooling layer. CAFE obtains 88.5% accuracy on the SNLI test set, an extremely competitive score on the extremely popular benchmark. Notably, competitive results can be also achieved with a much smaller parameterization. For example, CAFE also achieves 88.3% and 88.1% test accuracy with only $3.5M$ and $1.5M$ respectively. This outperforms state-of-the-art ESIM and DIIN models with only a fraction of the parameter cost. Moreover, our lightweight adaption achieves 87.7% with only $750K$ parameters, which makes it extremely performant amongst models having the same amount of parameters such as the decomposable attention model (86.8%).

After removing the inter-attention layers from CAFE, the performance remains competitive to top performing encoder models. CAFE (w/o cross attention) achieves a respectible 85.9% accuracy in this setting. Notably, the best performing model, the Gumbel TreeLSTM (86.0%) has over $10M$ parameters. On the other hand, our CAFE model has a 300% less parameters, yet performs competitively to the Gumbel TreeLSTM.

Finally, an ensemble of 5 CAFE models achieves 89.3% test accuracy, the best test scores on the SNLI benchmark to date. Overall, we believe that the good performance of our CAFE can be attributed to (1) the effectiveness of the compare-propagate (i.e., providing word representations with global and local knowledge for better representation learning) and (2) the expressiveness of factorization layers that are used to decompose and compare word alignments. More details are given at the ablation study. Finally, we emphasize that CAFE is also relatively lightweight, efficient and fast to train given its performance. A single run on SNLI takes approximately 5 minutes per epoch with a batch size of 256. Overall, a single run takes $\approx$ 3 hours to get to convergence.

4.4 Results on MultiNLI and Scitail

Table 2 reports our results on the MultiNLI and SciTail datasets. On MultiNLI, CAFE significantly outperforms ESIM, a strong SoTA models on both settings. We also outperform the ESIM + Read model (Weissenborn, 2017). An ensemble of CAFE models achieve the best reported results on this MultiNLI to date (as of time of writing). On SciTail, Our proposed CAFE model achieves state-of-the-art performance. The performance gain over strong baselines such as DecompAtt and ESIM are $\approx 10\% - 13\%$ in terms of accuracy. CAFE also outperforms DGEM , which use a graph-based attention for improved performance, by a significant margin of 5%. As such, empirical results demonstrate the effectiveness of our proposed CAFE model on the challenging SciTail dataset.

	MultiNLI		
Model	Match	Mismatch	SciTail
Majority	36.5	35.6	60.3
NGRAM$^{\#}$	-	-	70.6
CBOW$^{\flat}$	65.2	64.8	-
BiLSTM$^{\flat}$	69.8	69.4	-
ESIM$^{\#,\flat}$	72.4	72.1	70.6
DecompAtt$^{\#}$ -	-	-	72.3
DGEM$^{\#}$	-	-	70.8
DGEM + Edge$^{\#}$	-	-	77.3
ESIM†	76.3	75.8	-
ESIM + Read†	77.8	77.0	-
CAFE	78.7	77.9	**83.3**
CAFE Ensemble	**80.2**	**79.0**	-

Table 2: Performance comparison (accuracy) on MultiNLI and SciTail. Models with $\dagger$, $\#$ and $\flat$ are reported from (Weissenborn, 2017), (Khot et al., 2018) and (Williams et al., 2017) respectively.

4.5 Ablation Study

Table 3 reports ablation studies on the MultiNLI development sets. Firstly, we replaced all FM functions with regular full-connected (FC) layers (1). We notice an decline in performance in both development sets. Ablation (2-3) explores the utility of using character and syntactic embeddings, which we find to have helped CAFE marginally. (4) Removes the Inter attention features, which naturally impacts the model performance significantly. (5-6) explores the effectiveness of the highway layers (in prediction layers and encoding layers) by replacing them to FC layers. Both highway layers have marginally helped overall performance. Finally, (7-9) removes the alignment features based on their composition type. We observe that the Sub feat and Concat compositions were more important than the Mul composition. However, removing any of the three will result in some performance degrade. Finally, we replace the LSTM encoder with a BiLSTM, observing that adding bidirectionality did not improve performance for our model.

	Match	Mismatch
Original Model	79.0	78.9
(1) Replace FM with FC	77.7	77.9
(2) Remove Char Embed	78.1	78.3
(3) Remove Syn Embed	78.3	78.4
(4) Remove Inter Att	75.2	75.6
(5) Replace Highway Pred. with FC	77.7	77.9
(6) Remove Highway Enc. with FC	78.7	78.7
(7) Remove Sub Feat	77.9	78.3
(8) Remove Mul Feat	78.7	78.6
(9) Remove Concat Feat	77.9	77.6
(10) Replace LSTM with BiLSTM	78.3	78.4

Table 3: Ablation study on MultiNLI development sets.

4.6 Linguistic Error Analysis

We perform a linguistic error analysis using the supplementary annotations provided by the MultiNLI dataset. We compare against the model outputs of the ESIM model across 13 categories

	Matched		Mismatched	
	ESIM	CAFE	ESIM	CAFE
Conditional	100	70	60	**85**
Word overlap	50	**82**	62	**87**
Negation	**76**	**76**	71	**80**
Antonym	67	**82**	58	**80**
Long Sentence	75	**79**	69	**77**
Tense Difference	73	**82**	79	**89**
Active/Passive	88	**100**	**91**	90
Paraphrase	**89**	88	84	**95**
Quantity/Time	33	**53**	54	**62**
Coreference	**83**	80	75	**83**
Quantifier	69	**75**	72	**80**
Modal	78	**81**	76	**81**
Belief	65	**77**	67	**83**

Table 4: Linguistic Error Analysis on MultiNLI dataset.

of linguistic[3] phenenoma. Table 4 reports the result of our error analysis. Firstly, we observe that our CAFE model generally outperforms ESIM on *most categories*. On the mismatched setting, CAFE outperforms ESIM in 12 out of 13 categories, losing only in one percentage point in *Active/Passive* category. On the matched setting, CAFE is outperformed by ESIM very marginally on coreference and paraphrase categories. Despite generally achieving much superior results, we noticed that CAFE performs poorly on *conditionals*[4] on the matched setting. Measuring the absolute ability of CAFE, we find that CAFE performs extremely well in handling linguistic patterns of *paraphrase detection* and *active/passive*. This is likely to be attributed by the alignment strategy that CAFE and ESIM both exploits.

4.7 Interpreting and Visualizing with CAFE

Finally, we also observed that the propagated features are highly interpretable, giving insights to the inner workings of the CAFE model. Figure 2 shows a visualization of the feature values from an example in the SNLI test set. The ground truth is *contradiction*. Based on the above example we make several observations. Firstly, *inter_mul* features mostly capture identical words (or semantically similar words), i.e., inter_mul features for *river* spikes in both sentences. Secondly, *inter_sub* spikes on conflicting words that might cause contradiction, e.g *sedan* and *land rover* are not the same vehicle. Another interesting observation is that we notice the inter_sub features for *driven* and *stuck* spiking. This also validates the observation of (Chen et al., 2017a), which shows what the *sub* vector in the ESIM model is looking out for contradictory information. However, our architecture allows the inspection of these vectors since they are compressed via factorization, leading to larger extents of explainability - a quality that neural models inherently lack. We also observed[5] that intra attention (e.g., intra_cat) features seem to capture the more important words in the sentence (river, sedan, landrover).

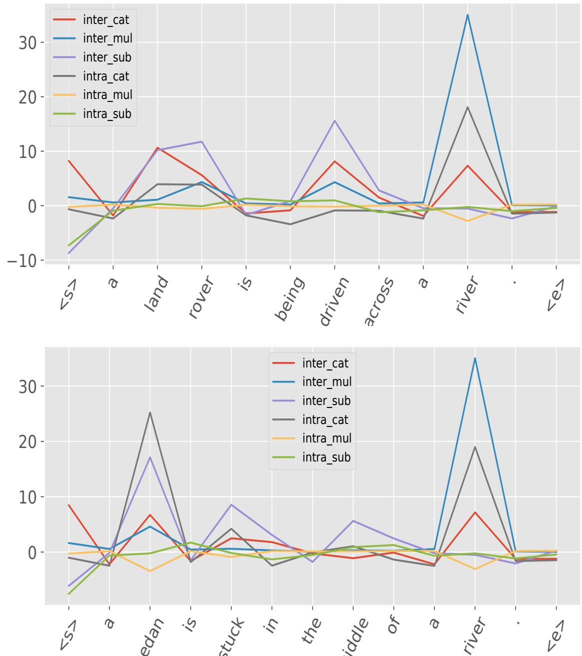

Figure 2: Visualization of *six* Propgated Features (*Best viewed in color*). Legend is denoted by {inter,intra} followed by the operations mul, sub or concat (cat).

5 Conclusion

We proposed a new neural architecture, CAFE for NLI. CAFE achieves state-of-the-art performance on three benchmark datasets. Moreover, a lightweight parameterization of CAFE outperforms other SoTA models such as ESIM and DINN while enjoying a 300% savings in parameter cost. The design of CAFE opens up new avenues for interpretability in neural models for NLI. Qualitatively, we show how different compositional operators (e.g., sub and mul) behave in NLI task and shed light on why subtractive composition helps in other models such as the ESIM.

[3] Due to the lack of space, we refer readers to http://www.nyu.edu/projects/bowman/multinli/multinli_1.0_annotations.zip for an explanation on each category.

[4] This only accounts for 5% of samples.

[5] Specific spikes in intra_mul and intra_sub was also observed in other samples but we leave them out due to the lack of space.

References

Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dan Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. 2015. TensorFlow: Large-scale machine learning on heterogeneous systems. Software available from tensorflow.org.

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 2015, Lisbon, Portugal, September 17-21, 2015*. pages 632–642. http://aclweb.org/anthology/D/D15/D15-1075.pdf.

Qian Chen, Xiaodan Zhu, Zhen-Hua Ling, Si Wei, Hui Jiang, and Diana Inkpen. 2017a. Enhanced LSTM for natural language inference. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 1: Long Papers*. pages 1657–1668. https://doi.org/10.18653/v1/P17-1152.

Qian Chen, Xiaodan Zhu, Zhen-Hua Ling, Si Wei, Hui Jiang, and Diana Inkpen. 2017b. Recurrent neural network-based sentence encoder with gated attention for natural language inference. In *Proceedings of the 2nd Workshop on Evaluating Vector Space Representations for NLP, RepEval@EMNLP 2017, Copenhagen, Denmark, September 8, 2017*. pages 36–40. http://aclanthology.info/papers/W17-5307/w17-5307.

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* .

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*. Springer-Verlag, Berlin, Heidelberg, MLCW'05, pages 177–190.

Yichen Gong, Heng Luo, and Jian Zhang. 2017. Natural language inference over interaction space. *CoRR* abs/1709.04348. http://arxiv.org/abs/1709.04348.

Adrian Iftene and Alexandra Balahur-Dobrescu. 2007. Hypothesis transformation and semantic variability rules used in recognizing textual entailment. In *Proceedings of the ACL-PASCAL Workshop on Textual Entailment and Paraphrasing*. Association for Computational Linguistics, Stroudsburg, PA, USA, RTE '07, pages 125–130.

Tushar Khot, Ashish Sabharwal, and Peter Clark. 2018. Scitail: A textual entailment dataset from science question answering. In *AAAI*.

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

Bill Maccartney. 2009. *Natural Language Inference*. Ph.D. thesis, Stanford, CA, USA. AAI3364139.

Bill MacCartney and Christopher D. Manning. 2008. Modeling semantic containment and exclusion in natural language inference. In *Proceedings of the 22Nd International Conference on Computational Linguistics - Volume 1*. Association for Computational Linguistics, Stroudsburg, PA, USA, COLING '08, pages 521–528.

Ankur P. 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 2016, Austin, Texas, USA, November 1-4, 2016*. pages 2249–2255.

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 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*. pages 1532–1543.

Steffen Rendle. 2010. Factorization machines. In *Data Mining (ICDM), 2010 IEEE 10th International Conference on*. IEEE, pages 995–1000.

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

Lei Sha, Baobao Chang, Zhifang Sui, and Sujian Li. 2016. Reading and thinking: Re-read LSTM unit for textual entailment recognition. In *COLING 2016, 26th International Conference on Computational Linguistics, Proceedings of the Conference: Technical Papers, December 11-16, 2016, Osaka, Japan*. pages 2870–2879. http://aclweb.org/anthology/C/C16/C16-1270.pdf.

Tao Shen, Tianyi Zhou, Guodong Long, Jing Jiang, Shirui Pan, and Chengqi Zhang. 2017. Disan: Directional self-attention network for rnn/cnn-free language understanding. *CoRR* abs/1709.04696. http://arxiv.org/abs/1709.04696.

Rupesh Kumar Srivastava, Klaus Greff, and Jürgen Schmidhuber. 2015. Highway networks. *CoRR* abs/1505.00387. http://arxiv.org/abs/1505.00387.

Shuohang Wang and Jing Jiang. 2016a. A compare-aggregate model for matching text sequences. *CoRR* abs/1611.01747. http://arxiv.org/abs/1611.01747.

Shuohang Wang and Jing Jiang. 2016b. Learning natural language inference with LSTM. In *NAACL HLT 2016, The 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, San Diego California, USA, June 12-17, 2016*. pages 1442–1451. http://aclweb.org/anthology/N/N16/N16-1170.pdf.

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 2017, Melbourne, Australia, August 19-25, 2017*. pages 4144–4150. https://doi.org/10.24963/ijcai.2017/579.

Dirk Weissenborn. 2017. Reading twice for natural language understanding. *CoRR* abs/1706.02596. http://arxiv.org/abs/1706.02596.

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

Jun Xiao, Hao Ye, Xiangnan He, Hanwang Zhang, Fei Wu, and Tat-Seng Chua. 2017. Attentional factorization machines: Learning the weight of feature interactions via attention networks. *arXiv preprint arXiv:1708.04617* .

Hong Yu and Tsendsuren Munkhdalai. 2017. Neural semantic encoders. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics, EACL 2017, Valencia, Spain, April 3-7, 2017, Volume 1: Long Papers*. pages 397–407. http://aclanthology.info/papers/E17-1038/neural-semantic-encoders.