Word n-gram attention models for sentence similarity and inference

I. Lopez-Gazpio[a,*], M. Maritxalar[a], M. Lapata[b], E. Agirre[a]

[a] *IXA NLP group, Computer Science faculty, University of the Basque Country (UPV/EHU), Manuel Lardizabal 1, 20018, Donostia, Basque Country*
[b] *Institute for Language, Cognition and Computation, School of Informatics, University of Edinburgh 10 Crichton Street, Edinburgh EH8 9AB*

Abstract

Semantic Textual Similarity and Natural Language Inference are two popular natural language understanding tasks used to benchmark sentence representation models where two sentences are paired. In such tasks sentences are represented as bag of words, sequences, trees or convolutions, but the attention model is based on word pairs. In this article we introduce the use of word n-grams in the attention model. Our results on five datasets show an error reduction of up to 41% with respect to the word-based attention model. The improvements are especially relevant with low data regimes and, in the case of natural language inference, on the recently released hard subset of Natural Language Inference datasets.

Keywords: Attention models, Deep Learning, Natural Language Understanding, Natural Language Inference, Semantic Textual Similarity

1. Introduction

A major challenge in Computational Linguistics is that of building meaning representation models to enable Natural Language Understanding (NLU). In order to train and evaluate those models the community has proposed several challenges and associated datasets, including Machine Comprehension

*Corresponding author
Email addresses: inigo.lopez@ehu.eus (I. Lopez-Gazpio), montse.maritxalar@ehu.eus (M. Maritxalar), mlap@inf.ed.ac.uk (M. Lapata), e.agirre@ehu.eus (E. Agirre)

(MC) (Rajpurkar et al., 2016), Question Answering (QA) (Yang et al., 2015), Automatic Short Answer Grading (ASAG) (Burrows et al., 2015), Natural Language Inference (NLI) (Bowman et al., 2015) and Semantic Textual Similarity (STS) (Agirre et al., 2012). In those tasks, the NLU system needs to pair two text snippets and then provide an output such as the relevance between a question and a text passage (MC), a question and an answer (QA), the two responses from a teacher and from a student (ASAG), the entailment relation between text and hypothesis (NLI) or the similarity score between two sentences (STS), respectively. In this paper we will focus on the latter two tasks, even the technique can be easily applied to the other tasks.

Computational linguists have used several approaches in the past, with deep learning systems getting consistently the best results when training data is available (Cer et al., 2017; Williams et al., 2017). These systems encode each of the input sentences into a vector using different methods, ranging from simple bag-of-words (BoW) (Parikh et al., 2016), convolutional neural networks (CNN) (Yin et al., 2016), recurrent neural nets such as LSTM (Nangia et al., 2017) to recursive tree LSTM (Tai et al., 2015). Some systems compare the vectors of the input sentences directly, and compute the output without access to the underlying information (Choi et al., 2017; Tai et al., 2015). The most successful systems, though, take also into account word alignment information, usually in the form of word attention models (Chen et al., 2016; Gong et al., 2017; Parikh et al., 2016). Those attention models capture the correspondences between words in the pair of sentences. On the other hand, Lopez-Gazpio et al. (2017) observe that alignments between linguistically motivated chunks[1] are very useful in order to capture the semantic relations between two sentences, in the framework of a shared task called Interpretable STS. Despite this observation, alignment and attention models continue to be limited to words.

This article proposes to extend the alignment information from pairs of words to pairs of word n-grams, motivated by the observation of Lopez-Gazpio et al. (2017). The use of word n-grams is common practice in statistical language models (Stolcke, 2002). More recently, sentence embedding models have complemented unigram (word) embeddings with bigram embeddings (Pagliardini et al., 2018). In our proposal we model attention as

[1]Chunks are similar to phrases, but do not require full parsing (Abney, 1991).

a weight for each possible word n-gram pair[2] instead of each possible word pair. We first extract sequences of contiguous words ranging from one single word to a maximum of N words for both sentence pairs, and build an attention matrix for all such n-gram pairs. In this work we use recurrent neural networks to represent n-grams, but other options like n-gram embeddings could be used (Zhao et al., 2017).

We explore the effect of the proposed attention model on a competitive BoW system called Decomposable Attention Model (DAM)(Parikh et al., 2016). We show that the n-gram alignment model improves results when compared to DAM with word attention, and that it is a better alternative than modeling context using LSTMs and CNNs. In addition, we train the attention model as a regression module, improving further the results. Our system is evaluated on multiple STS and NLI datasets. It is especially beneficial in datasets with lower amounts of training data and, in the case of NLI, on the hard subset of NLI datasets. Our system also compares well to the state-of-the-art, and shows promise for adding n-gram attention to other systems.

This article is structured as follows. We first lay out the background, including the STS and NLI tasks, followed by the definition of the Decomposable Attention Model. Section 3 introduces the proposal to extend the word alignment model. Section 4 describes the datasets and results. Section 5 presents the comparison to state-of-the-art systems. The final section draws the conclusions and mentions future work.

2. Background

In this section we review the the two sentence pairing tasks where we apply the proposed attention model, STS and NLI. In addition, we present the system which we will extend with our N-gram attention model.

2.1. STS and NLI

STS (Agirre et al., 2012) aims to measure the degree of semantic equivalence among two textual sentences. STS datasets are composed of input sentence pairs alongside their gold standard scores. Figure 1 shows a couple of examples extracted from two distinct STS sources: STS Benchmark (Cer

[2]For the sake of clarity we will use n-gram to mean word n-gram (as opposed to character n-gram) throughout this article.

```
Example 1
=========
A turtle walks over the ground.
A large turtle crawls in the grass.
Similarity score: 3.75

Example 2
=========
The children of a family are playing and waiting.
An Asian man is dancing and three kids are looking.
Similarity score: 1.9
```

Figure 1: Examples from Semantic Textual Similarity datasets. See text for further details.

```
Example 1
=========
Sentence 1 : A tiger cub is playing with a ball.
Sentence 2 : A baby is playing with a doll.
Relation label : Neutral

Example 2
=========
A white dog is chasing a stuffed animal.
Sentence 2: The animal is sleeping.
Relation label : Contradiction

Example 3
=========
Sentence 1: Please renew your commitment today.
Sentence 2: A renewal of commitment is required today.
Relation label : Entailment
```

Figure 2: Examples from Natural Language Inference datasets. See text for further details.

et al., 2017) and SICK textual similarity (Marelli et al., 2014). We review the cited datasets in further detail in Section 4.1. The gold standard scores are obtained by averaging the scores of several annotators, and ranges between 0 and 5. The highest value is for full semantic equivalence and the lowest value for no relation at all.

NLI datasets also comprise an input sentence pair and a manually assigned relation label, where the label establishes the entailment relation between the two sentences, which is usually one of entailment, neutral or contradiction (Bowman et al., 2015). NLI is also known as Textual Entailment (TE). Figure 2 shows three examples extracted from three distinct NLI sources: SICK Textual Entailment (Marelli et al., 2014), Stanford Natural Language Inference (Bowman et al., 2015) and Multi-Genre Natural Language Inference (Nangia et al., 2017). We also review the cited datasets in further detail in section 4.1. The first sentence in the pair is said to be the text (T) and the second sentence the hypothesis (H). The annotators need to decide whether T entails the hypothesis H, that is, whether reading T suggests that H is true (Dagan et al., 2006). If not, they need to decide whether T contradicts H. The remaining label is neutral, that is, T neither entails nor contradicts H.

Both STS and NLI are popular evaluation scenarios for semantic representation models, as similarity and entailment relations often involve complex linguistic phenomena. In fact, White et al. (2017) have converted several linguistically annotated datasets into entailment pairs. STS and SNLI datasets thus make it easy to judge the degree to which semantic representation models are able to effectively capture some aspects of the meaning of language.

STS is related to NLI, as argued in (Agirre et al., 2012). They both aim at capturing semantic relationships between the input sentence pairs. STS is symmetric and graded, while NLI is directional and categorical. They each are able to evaluate different traits of semantics, but both include desired requisites for any NLU system. An interesting example of the difference between similarity and inference is to consider the case between pairs of objects that hold the hypernym relation, e.g. two pairs like wildcat-cat and cat-animal. STS defines a similarity value for the pairs, higher for wildcat-cat than for cat-animal, but the same values as for the inverse pairs cat-wildcat and animal-cat. Inference is directional, and thus it captures entailment for wildcat-cat and neutrality for the inverse cat-wildcat, but does not differentiate the different strength of the association in wildcat-cat and animal-cat.

2.2. The Decomposable Attention Model (DAM)

There is a growing number of systems pushing the state-of-the-art results on STS and NLI upwards. In this work, we chose to add our n-gram attention model to the Decomposable Attention Model (Parikh et al., 2016) because of its simplicity, low number of parameters and high performance. DAM relies in the key concept that long sentences tend to be complex in structure, and, therefore, it is hard for computational models to construct a compact and reliable fixed-size representation that captures the entire meaning of the input. Furthermore, Parikh et al. (2016) state that most of the times the alignment among small parts of the content words can lead to successful entailment judgments. Although the system was originally designed for NLI,

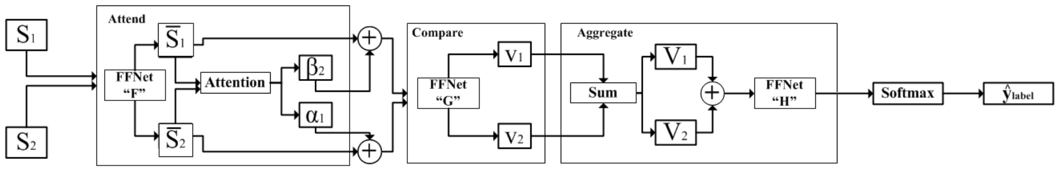

Figure 3: Architecture of the Decomposable Attention Model. The figure shows the concatenation of the three main layers of the model: the attention layer, the comparison layer and the aggregation layer. FFNet denotes a feed-forward neural network and the + operator denotes concatenation of vectors.

it is straightforward to adapt it to produce similarity scores, as we will see in the end of this section.

The architecture of DAM is shown in Figure 3. It consists of three feed-forward neural networks (F, G and H) structured in three consecutive layers as follows: the attention layer, the comparison layer and the aggregation layer. Each feed-forward network is composed of a single hidden layer and employ rectified linear units (ReLU) as non-linear functions[3]. Input and output dimensionality for all networks is kept constant and is defined by the global hidden size architectural setting[4].

The attention layer (*Attend* block of Figure 3) is where the soft-alignment between input words happen using a variation of neural attention. Given input sentences S_1 and S_2 represented as 2-dimensional tensors[5], the model first linearly transforms the input sentences applying the F network individually obtaining $\overline{S}_1$ and $\overline{S}_2$ respectively as output, following these equations: $\overline{S}_1 = F(S_1)$ and $\overline{S}_2 = F(S_2)$. Once the input sentences are transformed, the attention layer computes projection vectors Alpha (α_1) and Beta (β_2) as follows:

$$\beta_{2i} = \sum_{j=1}^{|\overline{S}_2|} \frac{\exp e_{ij}}{\sum_{k=1}^{|\overline{S}_2|} \exp e_{ik}} \overline{S}_{2j} \ , \ i \in |\overline{S}_1|$$

[3]For the sake of clarity we want to state that feed-forward networks (FFNet) consist of a total of 3 layers: input, hidden and output. Both hidden and output layers contain trainable parameters and the same non-linearity function (ReLU) after the linear transformation.

[4]Model hyper-parameters are accessible in tables 3, A.9, A.10, A.11 and A.12.

[5]The first dimension indexes word S_i from sentence S and the second dimension its corresponding word vector.

$$\alpha_{1j} = \sum_{i=1}^{|\overline{S}_1|} \frac{\exp e_{ij}}{\sum_{k=1}^{|\overline{S}_1|} \exp e_{kj}} \overline{S}_{1i} \ , \ j \in |\overline{S}_2|$$

where $|\overline{S}|$ denotes sentence length and e_{ij} denotes word to word attention computed as the dot product among normalized word vectors: $e_{ij} = \overline{S}_{1i} \cdot \overline{S}_{2j}$. As a result, β_2 contains the weighted sum of words from $\overline{S}_2$ projected onto the first sentence and α_1 contains the weighted sum of words from $\overline{S}_1$ projected onto the second sentence.

The comparison layer (*Compare* block of Figure 3) learns to compare the previously aligned words and projections, producing v_1 and v_2 vectors respectively:

$$v_{1i} = G([\overline{S}_{1i} \ ; \ \beta_{2i}]) \ , \ i \in |\overline{S}_1|$$

$$v_{2j} = G([\overline{S}_{2j} \ ; \ \alpha_{1j}]) \ , \ j \in |\overline{S}_2|$$

where the semicolon operation denotes vector concatenation.

The aggregation layer (*Aggregate* block of Figure 3) makes the final judgment based on the representation produced by the previous layers. It initially compacts and flattens the vectors containing the comparisons among words and projections, and obtains the final probability distribution estimates over the labels ($\hat{y}_{label}$) using the H network and softmax estimation. The final inference label (y_{label}) is obtained by picking the most probable class.

$$V_1 = \sum_{i=1}^{|\overline{S}_1|} \ v_{1i}$$

$$V_2 = \sum_{j=1}^{|\overline{S}_2|} \ v_{2j}$$

$$\hat{y}_{label} = \text{softmax}(H([V_1 \ ; \ V_2]))$$

$$y_{label} = argmax \ \hat{y}_{label}$$

We refer the reader to (Parikh et al., 2016) for further details on the DAM model. In this work, we re-implement the model to use it as a baseline. We call this baseline model DAM BoW. Our baseline model follows the same training criterion as Parikh et al. which uses negative log-likelihood as the loss function to be minimized.

DAM was proposed for NLI tasks. In order to adapt it to TS, we change the final layer so that the model performs regression instead of classification. We use the well-known approach by Tai et al. (2015) for this task. Following

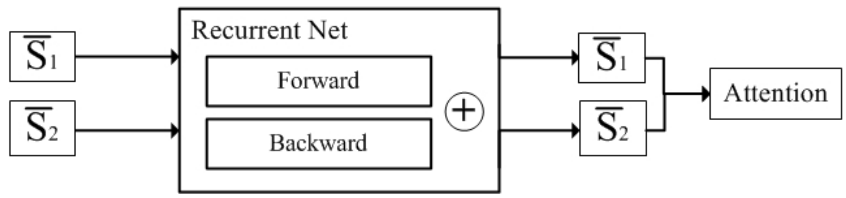

Figure 4: Addition to the attend module of DAM to introduce context using recurrence. See text for further details.

this work, during training the model predicts TS scores as if it were labels, optimizing the Kullback-Leibler divergence. For the task, the TS scores are converted into probability mass estimates over the discrete labels in the TS range. When testing the model to obtain the final predictions (y_{score}), the following formula is used to reconvert the probability mass estimates over the discrete labels into TS scores in the range $[0, 5]$:

$$y_{score} = r^T \cdot \hat{y}_{label}$$

where r is a row vector containing a value for each discrete number in the TS range.

3. Extensions to word alignment

The main contribution of this paper is the addition of n-gram attention to DAM. As one could argue that n-gram attention is merely adding context information into the attention model, we also implemented two extensions to the word attention model which add context information to tokens via recurrence and convolution. These two extensions are baselines which the n-gram attention model should outperform to show its value. We will first introduce these extensions and then present the n-gram attention model. In addition, our model benefits from a trainable attention model, which is presented last.

3.1. Adding context through recurrence

DAM BoW computes context-independent attention scores e_{ij} between words and, after that, re-weights the word vectors of the input sentences using the row-wise or column wise normalized e_{ij} values. As a consequence, the resulting tensors alpha and beta relate to input sentence 2 and input

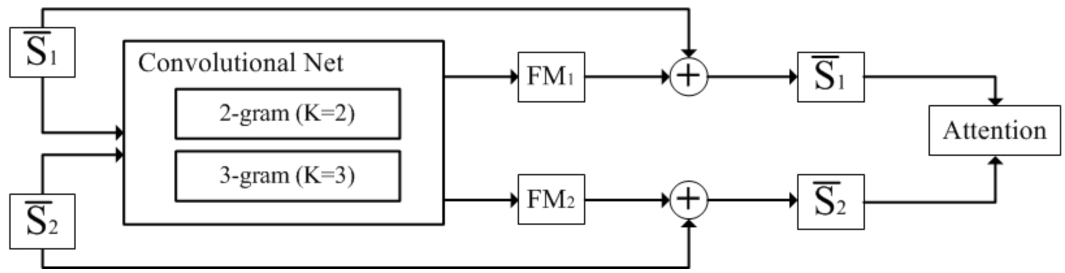

Figure 5: Addition to the attend module of DAM to introduce context using convolutions. See text for further details.

sentence 1 respectively based on e_{ij} values computed out of word to word interaction. In order to extend the word interaction between the input sentences, in this first extension we propose to run a recurrent neural network before the attention mechanism of DAM BoW in order to compute context-based representations of words. The context-dependent representations of words are then used to compute the attention scores e_{ij} in the same manner.

As shown in the schema of figure 4 in this extension we propose to modify the representation of every word on $\overline{S}_1$ and $\overline{S}_2$ formalizing it to be the concatenation of the forward and backward output states of a recurrent neural network for that word:

$$\overline{S}_i = [\mathrm{RNN}_i^f(\overline{S}) \; ; \; \mathrm{RNN}_i^b(\overline{S})] \; , \; i \in |\overline{S}|$$

where $\mathrm{RNN}_i^f(\overline{S})$ and $\mathrm{RNN}_i^b(\overline{S})$ denote the output state of the forward and backward passes respectively for word $i \in \overline{S}$. Note that by doing so the dimensionality required to represent each word in the sentence doubles.

3.2. Adding context through convolution

As an alternative to recurrence, one can exploit context through convolutions over nearby windows of words. We achieve this by concatenating the feature maps (FM) learned by convolution filters over input words. In this context, feature maps are defined as:

$$\mathrm{FM} = \mathrm{CNN}(\overline{S} \; , \; \text{filter size} = K)$$

We tested convolution filters (K) of sizes two and three respectively. A schema showing the changes required for this approach can be seen in Figure

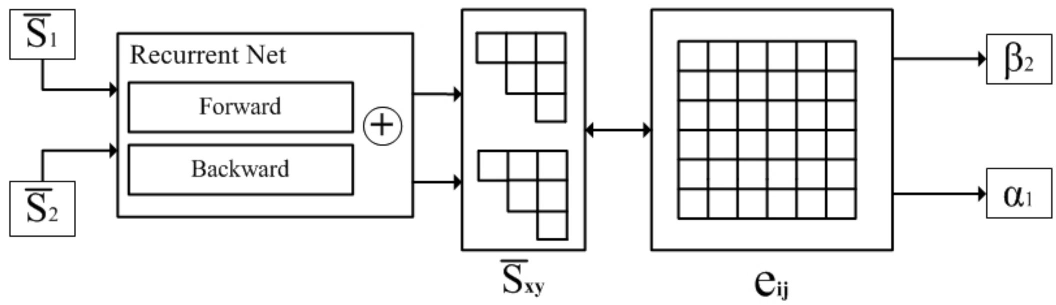

Figure 6: Addition to the attend module of DAM to introduce structure using arbitrary n-grams. See text for further details.

5. In a similar way to the extension based on recurrence, this time the dimensionality required to represent each word in the sentence also doubles as we concatenate the previous representation of the word with the learned feature map for every word in the sentence:

$$\overline{S}_i = [\overline{S}_i \ ; \ \mathrm{FM}_i(\overline{S})] \ , \ i \in |\overline{S}|$$

We apply padding when necessary to maintain the same dimensionality for the input and output vectors.

3.3. Word n-gram alignments

As mentioned in the introduction, this paper proposes to replace word alignment by n-gram alignment. Instead of enriching the representations of words using context (as done in the previous subsections), we hypothesize that explicitly representing word n-grams and computing attention between all possible n-gram pairs will perform better. Given the sentence $\overline{S}$ and assuming that $Ngram(\overline{S}, x, y)$ denotes the word n-gram starting from index x up to y in $\overline{S}$, the representation of the n-gram ($\overline{S}_{xy}$) is obtained as a bi-directional RNN which is run on that sequence as follows:

$$\overline{S}_{xy} = [RNN^f \ (\mathrm{Ngram}(\overline{S}, x, y)) \ ; \ RNN^b \ (Ngram(\overline{S}, x, y)) \ ]$$
$$1 \leq x \leq |\overline{S}| \ , \ x \leq y \leq |\overline{S}|$$

The resulting representation is an upper-diagonal matrix composed of all n-grams of $\overline{S}$, where the diagonal represents a 1-gram (single word) and subsequent squares to the right represent longer n-grams, which keep on adding words one by one at a time to the previous n-gram. The maximum size of n-gram to consider is defined by an hyper-parameter (N), $y - x < N$. Thus, the number of n-grams the model handles for sentence $\overline{S}$ is given by $|\overline{S}| \cdot N - \sum_{i=1}^{N-1} i$. When $N = 1$ the number of n-grams is equal to $|\overline{S}|$, that is, the number of elements in the diagonal, and if $N = |\overline{S}|$ the number of n-grams is equal to the number of elements in an upper triangular matrix of size $|\overline{S}|$ which is defined by $\sum_{i=1}^{|\overline{S}|} i$.

Figure 6 (middle box) shows the schema for the described architecture to represent n-grams. Given two sentences $\overline{S}_1$ and $\overline{S}_2$ the n-gram attention mechanism defines a matrix (e_{ij}) where i linearizes over the n-grams of $\overline{S}_1$ and j linearizes over the n-grams of $\overline{S}_2$. Figure 6 shows the full schema of the DAM N-gram approach. Note that the main difference with regard to DAM BoW resides in that, in this extension, each attention value e_{ij} captures the attention between n-gram i (corresponding to some n-gram $\overline{S}_{xy}$ spanning from x to y) from sentence $\overline{S}_1$ and n-gram j (corresponding to some n-gram $\overline{S}_{kz}$ spanning from k to z) from sentence $\overline{S}_2$. From another perspective, the attention model linearizes the triangular matrix of possible n-grams, that is, i is the linear index over possible (x,y) tuples and j is the linear index over possible (k,z) tuples.

3.4. Attention as an end-to-end trainable module

The usage of distinct attention mechanisms to attend to words has already been explored in the state-of-the art. For instance, Luong et al. (2015) define three well-known attention mechanisms. In the cited work the authors consider three distinct alternatives to score the attention between a pair of words: (1) neural attention, which is just the dot product (Bahdanau et al., 2015) $\overline{S}_i \cdot \overline{S}_j$ given $i \in \overline{S}_1$ and $j \in \overline{S}_2$; (2) general attention, which trains a weight matrix implemented as $\overline{S}_i \cdot W_1 \cdot \overline{S}_j$; and, (3) concat attention, which applies a 2-layer transformation to the concatenation of the representation for the words implied in the interaction, implemented as $W_2 \tanh(W_1[\overline{S}_i \, ; \, \overline{S}_j])$.

In this work we experiment with all the three variations above, but we adapted the concat attention model to use the same feed-forward neural network with ReLUs as in the DAM model (FFNet, see Section 2.2). We tested all three possibilities (cf. Section 4.4). We refer to our implementation of the concat attention as *FF attention*.

$$e_{ij} = \mathrm{FFNet}([\overline{S}_i \, ; \, \overline{S}_j])$$

Dataset	Train	Dev	Test	Total
STS Benchmark	5749	1500	1379	8628
SICK (TE /TS)	4439	495	4906	9840
SNLI (filtered)	549367	9842	9824	569033
MultiNLI (matched)	392702	9815	9796	412313

Table 1: Train, dev and test splits for all five datasets.

4. Experiments

We now describe the experiments involving the original DAM and the proposed extensions, including the datasets, the evaluation metrics, the experimental setup and implementation details, development experiments and the main results.

4.1. Description of the datasets

We evaluated our systems on the most relevant textual similarity and natural language inference datasets. Table 1 shows the number of examples for each of the datasets.

Semantic Textual Similarity has been the focus of an annual task until 2017 (Agirre et al., 2012; Cer et al., 2017). STS contributed towards defining an unified framework and stimulate research for evaluating systems that measure the degree of sentence level semantic equivalence. Each year the challenge brought together numerous participants, with new datasets. Recently, the organizers released a dataset that comprises a selection of all datasets, in order to provide a standard benchmark to evaluate different models in a unified framework, the **STS Benchmark** dataset. The selection of datasets includes those in the domain of image captions, news headlines and user forums (see Table 2). Note that the development set is partially mismatched regarding the training and test sets. We refer the reader to the official website[6] for further information. An example of this dataset is available in Figure 1 (Example 1).

The **SICK** dataset (Marelli et al., 2014)[7], *Sentences Involving Compositional Knowledge*, comprises semantically challenging sentence pairs, which

[6]`http://ixa2.si.ehu.es/stswiki/index.php/STSbenchmark`
[7]`http://clic.cimec.unitn.it/composes/sick.html`

had been semi-automatically selected and manipulated to comprise phenomena such as lexically rich words, contextual synonymy, active and passive changes, syntactic alternations and negation. The dataset was annotated both for textual similarity (**SICK-TS**) and textual entailment (**SICK-TE**). Sentences come from ImageFlickr[8] and MSR-Video descriptions[9]. More information can be gathered in the official website[10]. We provide two examples of this dataset in figures 1 (Example 2) and 2 (Example 1), the first annotated with a similarity score and the second with an inference label.

The previous datasets have a relatively small number of training examples. In an effort to mitigate the lack of large-scale resources and scale-up existing resources for machine learning research, Bowman et al. (2015) introduced the **Stanford Natural Language Inference** corpus (SNLI[11]). In contrast to previous resources, sentences from SNLI were written by crowdsourcing in a grounded, naturalistic context, and labels were inferred automatically. Consisting of a total of 570k pairs it is two orders of magnitude larger than all previous resources. Following usual practice, we use the filtered version, where pairs that do not exhibit annotation agreement were removed. An example from this dataset can be read in Figure 2 (Example 2).

While SNLI focused on image captions, **MultiNLI**[12] (Nangia et al., 2017) introduces new genres, enlarging the diversity of linguistic phenomena, including temporal reasoning, belief and modality among others. The test subset contains only five of the genres present in train and development. We thus focus on the matched subset of MultiNLI, where the subsets of training and development coming from those five genres are used. An example from the dataset can be observed in Figure 2 (Example 3).

4.2. Evaluation metrics

Following usual practice we use Pearson product-moment correlation coefficient to report performance on TS datasets and accuracy to report performance on NLI datasets. Pearson measures the linear dependence between a pair of variables and outputs a value in the range $[-1, 1]$. Accuracy states

---

[8] `http://nlp.cs.illinois.edu/HockenmaierGroup/data.html`

[9] `http://www.cs.york.ac.uk/semeval-2012/task6/index.php?id=data`

[10] `http://clic.cimec.unitn.it/composes/sick.html`

[11] `https://nlp.stanford.edu/projects/snli/`

[12] `https://www.nyu.edu/projects/bowman/multinli/`

Genre	Train	Dev	Test
Microsoft Research Paraphrase	1000	250	250
SemEval news headlines	1999	250	250
SemEval DEFT news	300	0	0
Microsoft Research video captions	1000	250	250
SemEval Image captions (2014-2015)	1000	250	250
SemEval Image captions (2017)	0	125	125
SemEval DEFT forum crawl	450	0	0
SemEval question-answer pairs in forums	0	375	0
SemEval answer-answer pairs in forums	0	0	254

Table 2: Sources used in the STS Benchmark dataset, showing that development set is partially mismatched with regards to the training and test sets.

the number of predicted examples that hold the same label with regards to the gold standard annotation divided by the total number of samples in the dataset and outputs a value in the range $[0, 1]$.

4.3. Implementation details

We used Pytorch in the implementation[13]. The texts were tokenized and punctuation removed. Regarding hyper-parameters and design options, we run experiments on the development datasets alone.

Following (Parikh et al., 2016) we use pre-trained Glove word embeddings in the input. Glove word embeddings[14] have been broadly used to initialize a wide range of neural network architectures and are based on word co-occurrence counts (Pennington et al., 2014). We tested several versions on development data, with the best results for the embeddings trained on with 840 Billion tokens.

Feed-forward networks used ReLU non-linearity. We tried several approaches for the recurrent neural networks employed in sections 3.1 and 3.3, including simple Recurrent Neural Networks (RNNs), Gated Recurrent Units (GRUs) (Cho et al., 2014) and Long Short-Term Memory networks (LSTMs) (Hochreiter & Schmidhuber, 1997). We empirically found out that LSTMs and GRUs outperform RNNs by large margin, whereas the performance between LSTMs and GRUs was similar, slightly in favor of GRUs. We opted

[13]Code and models to be made available upon acceptance.

[14]`https://nlp.stanford.edu/projects/glove/`

	SICK-TS	STS-B	Multi NLI	SNLI	SICK-TE
Embedding size	300	300	300	300	300
Hidden size	450	450	500	600	1050
Weight decay	5e-5	5e-5	5e-5	5e-5	5e-5
Max grad norm	5.0	5.0	5.0	5.0	5.0
Dropout	0.5	0.5	0.25	0.15	0.15
Param init	1e-2	1e-2	1e-2	1e-2	1e-2
Learning rate	9e-4	1e-4	1e-4	7.5e-5	1.9e-5
Epochs	87	46	98	95	72
Optimizer	Adam	Adam	Adam	Adam	Adam
Max n-gram size	2	2	4	4	4

Table 3: Hyper-parameters for the proposed system, DAM N-gram with FF attention. See appendix for the hyper-parameters of the baseline systems.

in favor of GRUs as the defined neuron unit is simpler while still keeps the memory gate that makes the difference with respect to standard RNNs. The election of GRUs over LSTMs also favors the time required to train models by large margin. We also follow (Tai et al., 2015) for Textual Similarity tasks so that instead of concatenating word embedding vectors $\vec{A}$ and $\vec{B}$ we concatenate their element-wise difference (distance) defined as $|\vec{A}-\vec{B}|$ and their element-wise product (angle) defined as $\vec{A}\odot\vec{B}$. For NLI tasks, we empirically observed that concatenating $\vec{A}$, $\vec{B}$, $|\vec{A}-\vec{B}|$ and $\vec{A}\odot\vec{B}$ yields slightly better results.

We included dropout in all layers. We noted that high dropout ratios (40% - 50%) are useful in Textual Similarity datasets as the training set is reduced in size, and complex models can easily overfit them. In tasks with larger available resources we did not find dropout to be among the most important hyper-parameters to tune. We tested both Adagrad (Duchi et al., 2010) and Adam (Kingma & Ba, 2014) optimizers.

We optimize all the hyper-parameters using random search (Bergstra & Bengio, 2012) which is stated to find better settings in a limited amount of time compared to grid search. The hyper-parameters were tuned using the available development set for each dataset separately. The hyper-parameters of our proposed model are described in Table 3, while the hyper-parameters for the rest of baselines are detailed in the appendix.

System	Train	Dev		Train	Dev
DAM BoW	.927	.746	+ FF att.	.946	.765
DAM RNN	.935	.757	+ FF att.	.922	.780
DAM CNN_2	.915	.747			
DAM CNN_3	.971	.774	+ FF att.	.972	.771
DAM N-gram	.930	**.801**	+ FF att.	.928	**.817**

Table 4: Development results (Pearson) in the STS Benchmark dataset for distinct approaches. The rightmost columns correspond to the respective DAM versions with FF attention (cf. Section 3.4).

4.4. Development of the systems on STS-B

In order to develop the system proposed in Section 3, we decided to do some development experiments on the STS Benchmark development dataset first. We chose STS Benchmark because it is smaller than the Natural Language Inference datasets, and, compared to the SICK datasets, it contains a wider range of topics and the development set is partially mismatched regarding the training and test sets (see Table 2).

Table 4 shows the development results. In the first row we show the DAM architecture (DAM BoW, cf. Section 2.2). In the rows below we show the results for the two baseline methods to encode context in the attention model (DAM RNN, cf. Section 3.1, and DAM CNN, cf. Section 3.2) as well as our proposed model (DAM N-gram, cf. Section 3.3). The DAM CNN model includes two rows, as we tested filters of maximum width 2 and 3. The results show that all approaches to encode context improve the results with respect to the original DAM, with RNNs yielding a weak gain, CNNs with width 3 performing better, and with the best results for the n-gram attention model.

The table also shows, in the rightmost columns (denoted by *+ FF att.*), the results when adding the FF attention module (described in Section 3.4) to the systems in the rows. We report the results for the most significant systems. The FF attention module yields improvements between 2.3 and 1.6 points, except for CNNs, where it does not improve results. We also tested the general attention model (cf. Section 3.4), but found that it is below the Feed-Forward attention model by 3 - 1.5 absolute points.

All in all, the best results are obtained by our n-gram attention model with FF attention, with improvements of around 5 points with respect to the original word-based attention model. The results also show that the n-gram

System	SICK-TS Dev	SICK-TS Test	STS-B Dev	STS-B Test	MultiNLI Dev	MultiNLI Test	SNLI Dev	SNLI Test	SICK-TE Dev	SICK-TE Test
DAM BoW	.768	.771	.746	.679	.717	.725	.854	.852	.745	.727
DAM BoW$_{FFatt}$	.802	.794	.765	.726	.681	.676	.855	.854	.765	.766
DAM RNN$_{FFatt}$	.836	.826	.780	.742	.719	.720	.857	.850	.796	.787
DAM CNN$_3$	.811	.814	.774	.741	.722	.721	.852	.856	.789	.781
DAM N-gram$_{FFatt}$	**.860**	**.857**	**.817**	**.773**	**.750**	**.748**	**.867**	**.863**	**.844**	**.840**

Table 5: Results for baselines and proposed model in textual similarity (Pearson) and inference datasets (Accuracy). FFatt for FF attention, STS-B for STS-Benchmark.

attention model is superior to the alternative baselines (RNNs or CNNs) to infuse context information into a word-based attention model. We will confirm these development results when testing on all five datasets.

4.5. Main results

We now evaluate the most representative systems on all five test datasets, including textual similarity and inference, as seen in Table 5. All hyper-parameters were set using the respective development dataset (cf. Section 4.3). Regarding the performance of our implementation of DAM (DAM BoW), it is better by around half a point on both MultiNLI and SNLI over the performance of the implementation reported on Gururangan et al. (2018), although it is one point below the performance reported by the original authors on SNLI (Parikh et al., 2016).

The table includes also DAM BoW with FF attention (DAM BoW$_{FFatt}$), and the best RNN, CNN and n-gram attention system as reported in the development experiments. The results confirm the trends observed in development: the two baseline methods to encode context in the attention model (DAM RNN, cf. Section 3.1, and DAM CNN, cf. Section 3.2) improve over the original DAM model, and that both models perform very similarly, although RNNs require the FF attention model to match CNNs. Our proposed model (DAM N-gram, cf. Section 3.3) yields the best results in all five datasets. The improvements vary from dataset to dataset, with the biggest gains on SICK-TE (11.3 absolute points and a relative error reduction of 41%), SICK-TS (8.6 absolute, 38% error reduction) and STS Benchmark (9.4 and 29%, respectively). The gains for the SNLI and MultiNLI datasets are smaller, 1.1 and 2.3 absolute points, 7.4% and 8.4% error reduction, respectively.

System	MultiNLI hard	SNLI hard
DAM BoW	.563	.712
DAM BoW$_{FFatt}$	.496	.711
DAM RNN$_{FFatt}$	.551	.704
DAM CNN$_3$	.563	.717
DAM N-gram$_{FFatt}$	**.611**	**.734**

Table 6: Results (accuracy) for baselines and proposed model in the hard subsets for SNLI and MultiNLI (Gururangan et al., 2018).

We examined the reason for the smaller differences in MultiNLI and SNLI. Gururangan et al. (2018) found that significant portions in SNLI (67%) and MultiNLI (53%) could be solved based on the hypothesis text alone, ignoring the premise sentence. This large portion of trivial pairs can make the differences in performance smaller, and they thus released two subsets of MultiNLI and SNLI, the so-called hard subsets. We evaluated our systems on the hard subsets, and found that the ranking of systems does not vary, but the differences in performance are larger (see Table 6). The absolute difference between our proposed n-gram attention model with learnable attention and the BoW model is of 2.2 and 4.8 points for SNLI and MultiNLI, respectively, and the error reduction of 7.6% and 11.1%, confirming that the trivial parts of SNLI and MultiNLI dilute performance differences. These results show that our system is specially effective for the more realistic hard pairs of the NLI datasets.

In addition, we studied whether the amount of training data is an important factor in the performance differences. Figure 7 shows the performance on the hard subset of SNLI of relevant systems with smaller subsets of the training data. The figure clearly shows that our proposal is more effective on the smaller subsets. The fact that the performance differences are also larger on the three datasets with smaller amounts of training data (STS-B and the two SICK datasets) seems to confirm that our proposed algorithm is specially effective on low data regimes.

In summary, the results across the five datasets confirm the development results. Our n-gram attention model combined with FF attention is able to provide large performance gains with respect to word-based attention models, including those models using RNNs or CNNs to add context information into the word-based attention model.

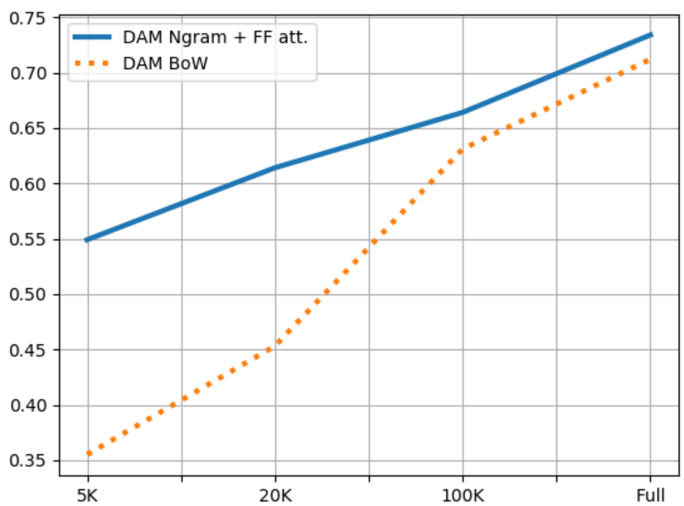

Figure 7: Results (accuracy) for different training set sizes in the hard subset for the baseline and proposed model.

5. Comparison to the state-of-the-art

Representing and comparing two text snippets as in STS and NLI is a usual benchmark for testing NLU architectures. We will review the most relevant state-of-the-art systems for our work, with emphasis on the attention model that they use. Head-to-head empirical comparison of specific components across architectures is difficult, as the final results of complex systems are affected by several design decisions, including pre-processing, sentence representation, attention model, or final classification/regression layer.

The goal of this section is to show that the performance of the DAM system with our n-gram attention model is competitive with respect to comparable systems, that is, systems which have comparable modules on all layers but attention. We first group the best-known systems, then compare their performance head-to-head in a table, and finally discuss the differences with respect to the two best-performing systems.

We can classify existing systems according to their representation for the input texts: transfer models, recurrent models, convolutional models and recursive models. Finally, we group ensemble models.

Transfer models employ external learning objectives to train distributed representations of sentences. *Sent2Vec* (Pagliardini et al., 2018), for instance, is an extension of CBOW (Mikolov et al., 2013) that learns word representations such that each word in the sentence can be predicted based on the average of the representations for the rest of the words in the sentence. Similarity is computed as the cosine between those vectors. To our knowledge it has not been applied to NLI. *SkipThought* (Kiros et al., 2015) is another example of this kind in which the training objective is to maximize the reconstruction of neighboring sentences based on the recurrent representation (using LSTMs) of the current sentence. A classification and regression layer is trained on the respective similarity or NLI dataset, in order to fine-tune the representations to the task. Note that none of these two methods use any attention layer. As an alternative to transfer models, the methods presented below learn the representations directly on the provided training data, with the exception of word embeddings, which are often initialized with pre-trained values.

RNN models also encode the meaning of the sentence into a single vector and compute the final label out of it using some kind of recurrent structures such as bidirectional LSTMs. Williams et al. (2017) present a baseline system, *Bi-LSTM*. which uses a bidirectional LSTM to encode the meaning of the sentences, and then compute distance and angle vector features between the two sentences, which are fed to a single non-linear layer. This system does not use any attention model. $ESIM_{seq}$ (Chen et al., 2016) is based on bidirectional LSTMS, and introduces a word-based attention layer and an extra layer of bidirectional LSTMs on top of the word-based attention layer. *DINN* (Gong et al., 2017) uses additional features in the input, where each word is represented as a concatenation of a word embedding, character features and syntactic features. In addition, they use multi-head attention, which is an extension of standard word-based attention to a 3D tensor, where the attention between two words is represented as a vector instead of a single scalar. The attention tensor is exploited using deep convolutional networks.

Tree models employ recursive tree-structured neural networks such as Tree-LSTMs to learn to compose the appropriate structure out of the input. *Constituency* and *dependency Tree-LSTM* (Tai et al., 2015) generalize regular linear LSTM chains into Tree-structured LSTM chains. The *Gumbel TreeLSTM* approach (Choi et al., 2017) uses an alternative tree-learning algorithm which dynamically selects candidate nodes using Straight-Through Gumbel-Softmax estimation. The previous two models do not use any attention model.

FF models use feed-forward networks to encode the meaning of the sentence. They include DAM, which uses a word attention model, and therefore our proposal is also a member of this family. More recently, self-attention has emerged as a powerful tool to model intra-sentence dependencies. *Reinforced self-learning* (Shen et al., 2018) combine soft and hard attention with an emphasis on self-attention and also include multi-head attention. The hard attention module trims the input for the soft attention module, while the soft attention module feeds back signals in the form of rewards to the hard attention module. Both are combined via reinforcement learning. They claim to extract efficiently the sparse dependencies between selected token pairs without involving recurrence or convolutions.

Feature-based models are based on sets of manually designed heuristics encoded as features which are fed to a machine learning algorithm. For instance, *ECNU* (Zhao et al., 2014) combines a total of seventy two features including length differences, word overlap measures weighted with tf.idf, matrix factorization of distributional vectors, overlap of dependencies, antonyms from WordNet, string similarity and co-occurrence-based distributional models. Support vector machines are used to classify (or regress) the target entailment (or similarity) label.

Finally, **ensemble models** obtain improvements combining the output of several models. For instance, $ESIM_{seq+tree}$ (Chen et al., 2016) combines the recurrent model mentioned above with another system based on Tree-LSTMs. *DINN* (Gong et al., 2017) does the majority vote of the predictions given by multiple runs of the same model (see Single DINN above) under different random parameter initialization. Alternatively, models which are substantially different can also be combined. *BiLSTM-Max+AIINLI* (Conneau et al., 2017) combines two different recurrent models (LSTMs and GRUs), self-attentive networks and hierarchical convolutional networks.

Table 7 shows the results of the systems mentioned above, with Table 8 reporting the results on the hard subset of SNLI and MultiNLI. Given that systems have been evaluated in different datasets, the comparison between two systems is limited to common datasets. Ensemble methods, as expected yield the best results in all datasets. We include them for completeness, but we are mainly interested in the comparison between single systems.

The best system in each dataset varies, with one different winner in each dataset, except our proposed system which is the best on SICK-TE and STS Benchmark. Our system performs better than Sent2Vec, Bi-LSTMs, Gumbel Tree-LSTM and ECNU in all datasets in common. The comparison with the

System	Type	Attention	MNLI	SNLI	S-TE	S-TS	STSB
DAM BoW	FF	Word	.725	.852	.727	.771	.679
DAM N-gram$_{FFatt}$	FF	N-gram	.748	.863	**.840**	.857	**.773**
Sent2vec	Transfer	-				.620	.755
SkipThought	Transfer	-			.823	.858	
BiLSTM	RNN	-	.669	.815			
ESIM$_{seq}$	RNN	Word	.723[a]	**.867**[a]			
Single DINN	RNN	3D	**.788**	.865[b]			
Constituency Tree-LSTM	Tree	-				.868	.719
Gumbel Tree-LSTM	Tree	-		.860			
Reinforced self-attention	FF	Self		.863		**.872**	
ECNU	Feature	-			.836	.828	
ESIM$_{seq+tree}$	Ensemble	Word		.886			
DINN	Ensemble	3D	.800	.889			
BiLSTM-Max+AIINLI	Ensemble	-			.863	.884	

Table 7: Results (accuracy) for our models (first two rows) and representative state-of-the-art models on STS and NLI datasets (see text for references). Best non-ensemble systems in bold, second best underlined. MNLI for MultiNLI, S-TE for SICK-TE, S-TS for SICK-TS and STSB for STS Benchmark. Source of results are the original papers (see text for references), with the following exceptions: [a] (Williams et al., 2017), [b] Gururangan et al. (2018).

rest of the systems is not clear, as our system wins in one dataset but not in the other. The only exceptions are Single DINN, which is better than our system in both MultiNLI and SNLI, and Reinforced self-attention, which is better on SICK-TS and equal on SNLI.

The qualitative comparison between competing systems and ours shows that our n-gram attention model is a module which could be complementary to the components of the other systems and vice-versa. We will now focus on those differences, system by system. For instance, SkipThought trains the LSTM in the input layer on a very large unsupervised task, and reuses it for STS and NLI. The addition of a n-gram attention model could further improve results, and, on the opposite direction, transfer learning could improve the sentence representations of our system.

In the case of ESIM$_{seq}$, they use Bi-LSTMs both in the input layer and after attention, in the inference layer. This double use of recurrence is complementary to the use of our n-gram attention model, and adding the recurrent networks to our model could further improve results. In any case, the

System	SNLI hard	MultiNLI hard
DAM N-gram$_{FFatt}$	**.734**	.611
ESIM$_{seq}$	.713	.593
Single DINN	.727	**.641**

Table 8: Results (accuracy) for proposed model and two competing models in the hard subsets of SNLI and MultiNLI (Gururangan et al., 2018). ESIN and DINN results taken from (Gururangan et al., 2018).

results on the hard subsets (Table 8) shows that our system beats ESIM$_{seq}$ on both SNLI and MultiNLI when trivial examples are ruled out.

Regarding Single DINN, it uses a richer input layer, a multi-head attention layer, and convolution and pooling layers. The comparison on the hard subset (Table 8) shows that our system is better on SNLI, and reduces the difference on MultiNLI. We think that enriching their attention layer with n-grams such as ours, or, conversely, adding multi-head attention to our n-gram attention are promising directions for future research.

Regarding recursive encoders, the comparison to our method shows that using n-grams instead of syntax yields slightly better results (better on STS Benchmark by 5 points, worse on SICK-TS by 1 point) with less complexity. We think that these comparative results show that the n-gram attention model is able to partially capture syntactic information.

Finally, the system based on self-attention coupled with hard and soft attention has obtained slightly better results (same results on SNLI, 1 point better on SICK-TS). The use of self-attention is a promising direction of research, which could complement the good results of our n-gram attention model.

6. Conclusions and future work

In this work we extend attention models from pairs of words to pairs of word n-grams of variable length. We plugged our attention model on the well-known Decomposable Attention Model system (Parikh et al., 2016), which is known for obtaining strong results on Natural Language Inference datasets. Our n-gram attention model improves results on five textual similarity and inference datasets, with up to 41% error reduction and 11 points of absolute gain. The gains are especially large for datasets with small training data, and the hard subsets of MultiNLI and SNLI datasets (Gururangan et al., 2018). Our experiments show that the alternative means to infuse context information into a word-to-word attention model (e.g. using a CNN or RNN over the context of occurrence) also improve results, but our method is the most effective. We also show that a trainable attention model increases results in all cases.

We think that the better results compared to recursive tree-based systems shows that n-grams are capturing some syntactic information. Our proposal can be seen as an intermediate step between learning a latent grammar (Tai et al., 2015) and staying at the flat word level: we add some structure in the form of a flat set of possible word n-grams, but do not require a full-fledged tree.

From another perspective, our work can be additional evidence on the benefits of aligning chunks defended by Lopez-Gazpio et al. (2017). In their work, a linguistically motivated software identifies chunks and then aligns them across the target sentences. The system solving the task uses the provided training data to learn how to relate and align pairs of chunks. In our case, our n-gram attention model can be seen as inducing chunks (n-grams) and alignments between pairs of chunks without any direct supervision. We would like to explore whether chunk alignment corpora can be used to better train our n-gram attention model. Alternatively, our n-gram attention model might help improve systems solving the Interpretable STS task, including short answer grading (Riordan et al., 2017).

Code and models are publicly available[15]. The analysis of state-of-the-art systems shows that our n-gram attention layer could be also beneficial (Chen et al., 2016; Gong et al., 2017; Shen et al., 2018), as all top-scoring systems use word-to-word attention models. The benefits of our attention model could be also extended to other problems where the standard word attention model is used (Luong et al., 2015; Rajpurkar et al., 2016; Yang et al., 2015). Finally, we would also like to explore whether the n-gram attention model trained in one task can be transferred to tasks with less training data.

Acknowledgments

This research is supported by a doctoral grant from MINECO. We also thank for technical and human support provided by IZO-SGI SGIker of UPV/EHU and European funding (ERDF and ESF), and, also, we gratefully

[15]To be made available on github upon acceptance

acknowledge the support of NVIDIA Corporation with the donation of a Pascal Titan X GPU used for this research. Finally, we would like to thank Siva Reddy[16] for his time, interest and fruitful comments made to the present work.

References

Abney, S. (1991). Parsing by chunks. In *Principle-based parsing: Computation and psycholinguistics. Robert Berwick and Steven Abney and Carol Tenny(eds.)* (pp. 257–278). Springer Science & Business Media.

Agirre, E., Bos, J., iab, M., Manandhar, S., Marton, Y., & Yuret, D. (2012). *sem 2012: The first joint conference on lexical and computational semantics – volume 1: Proceedings of the main conference and the shared task, and volume 2: Proceedings of the sixth international workshop on semantic evaluation (semeval 2012). In **SEM 2012: The First Joint Conference on Lexical and Computational Semantics – Volume 1: Proceedings of the main conference and the shared task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation (SemEval 2012)*. Association for Computational Linguistics. URL: `http://www.aclweb.org/anthology/S12-1000`.

Bahdanau, D., Cho, K., & Bengio, Y. (2015). Neural machine translation by jointly learning to align and translate. *In Proceedings of ICLR*, .

Bergstra, J., & Bengio, Y. (2012). Random search for hyper-parameter optimization. *Journal of Machine Learning Research*, *13*, 281–305.

Bowman, S. R., Angeli, G., Potts, C., & Manning, C. D. (2015). A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics.

Burrows, S., Gurevych, I., & Stein, B. (2015). The eras and trends of automatic short answer grading. *International Journal of Artificial Intelligence in Education*, *25*, 60–117.

[16] `http://sivareddy.in/`

Cer, D., Diab, M., Agirre, E., Lopez-Gazpio, I., & Specia, L. (2017). Semeval-2017 task 1: Semantic textual similarity multilingual and crosslingual focused evaluation. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)* (pp. 1–14). Association for Computational Linguistics. URL: `http://www.aclweb.org/anthology/S17-2001`. doi:`10.18653/v1/S17-2001`.

Chen, Q., Zhu, X., Ling, Z.-H., Wei, S., & Jiang, H. (2016). Enhancing and combining sequential and tree lstm for natural language inference. *CoRR*, *abs/1609.06038*. URL: `http://dblp.uni-trier.de/db/journals/corr/corr1609.html#ChenZLWJ16`.

Cho, K., van Merrienboer, B., Bahdanau, D., & Bengio, Y. (2014). On the properties of neural machine translation: Encoder–decoder approaches. In *Proceedings of SSST-8, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation* (pp. 103–111). Doha, Qatar: Association for Computational Linguistics. URL: `http://www.aclweb.org/anthology/W14-4012`.

Choi, J., Yoo, K. M., & Lee, S. (2017). Unsupervised learning of task-specific tree structures with tree-lstms. *CoRR*, *abs/1707.02786*. URL: `http://arxiv.org/abs/1707.02786`.

Conneau, A., Kiela, D., Schwenk, H., Barrault, L., & Bordes, A. (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* (pp. 670–680). Copenhagen, Denmark: Association for Computational Linguistics. URL: `https://www.aclweb.org/anthology/D17-1070`.

Dagan, I., Glickman, O., & Magnini, B. (2006). The pascal recognising textual entailment challenge. In *Machine learning challenges. evaluating predictive uncertainty, visual object classification, and recognising tectual entailment* (pp. 177–190). Springer.

Duchi, J., Hazan, E., & Singer, Y. (2010). *Adaptive Subgradient Methods for Online Learning and Stochastic Optimization*. Technical Report UCB/EECS-2010-24 EECS Department, University of California, Berkeley. URL: `http://www2.eecs.berkeley.edu/Pubs/TechRpts/2010/EECS-2010-24.html`.

Gong, Y., Luo, H., & Zhang, J. (2017). Natural language inference over interaction space. *CoRR*, *abs/1709.04348*. URL: `http://arxiv.org/abs/1709.04348`.

Gururangan, S., Swayamdipta, S., Levy, O., Schwartz, R., Bowman, S., & Smith, N. A. (2018). Annotation artifacts in natural language inference data. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)* (pp. 107–112). New Orleans, Louisiana: Association for Computational Linguistics. URL: `http://www.aclweb.org/anthology/N18-2017`.

Hochreiter, S., & Schmidhuber, J. (1997). Lstm can solve hard long time lag problems. In *Advances in neural information processing systems* (pp. 473–479).

Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. *CoRR*, *abs/1412.6980*. URL: `http://dblp.uni-trier.de/db/journals/corr/corr1412.html#KingmaB14`.

Kiros, R., Zhu, Y., Salakhutdinov, R. R., Zemel, R., Urtasun, R., Torralba, A., & Fidler, S. (2015). Skip-thought vectors. In *Advances in neural information processing systems* (pp. 3294–3302).

Lopez-Gazpio, I., Maritxalar, M., Gonzalez-Agirre, A., Rigau, G., Uria, L., & Agirre, E. (2017). Interpretable semantic textual similarity: Finding and explaining differences between sentences. *Knowledge-Based Systems*, *119*, 186–199.

Luong, T., Pham, H., & Manning, C. D. (2015). Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing* (pp. 1412–1421). Lisbon, Portugal: Association for Computational Linguistics. URL: `http://aclweb.org/anthology/D15-1166`.

Marelli, M., Bentivogli, L., Baroni, M., Bernardi, R., Menini, S., & Zamparelli, R. (2014). Semeval-2014 task 1: Evaluation of compositional distributional semantic models on full sentences through semantic relatedness and textual entailment. In *Proceedings of the 8th International Workshop* *on Semantic Evaluation (SemEval 2014)* (pp. 1–8). Dublin, Ireland: Association for Computational Linguistics and Dublin City University. URL: http://www.aclweb.org/anthology/S14-2001.

Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). Efficient estimation of word representations in vector space. *CoRR*, *abs/1301.3781*. URL: http://arxiv.org/abs/1301.3781. arXiv:1301.3781.

Nangia, N., Williams, A., Lazaridou, A., & Bowman, S. (2017). The repeval 2017 shared task: Multi-genre natural language inference with sentence representations. In *Proceedings of the 2nd Workshop on Evaluating Vector Space Representations for NLP* (pp. 1--10). Copenhagen, Denmark: Association for Computational Linguistics. URL: http://www.aclweb.org/anthology/W17-5301.

Pagliardini, M., Gupta, P., & Jaggi, M. (2018). Unsupervised Learning of Sentence Embeddings using Compositional n-Gram Features. In *NAACL 2018 - Conference of the North American Chapter of the Association for Computational Linguistics*.

Parikh, A., Täckström, O., Das, D., & Uszkoreit, J. (2016). A decomposable attention model for natural language inference. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing* (pp. 2249--2255). Austin, Texas: Association for Computational Linguistics. URL: https://aclweb.org/anthology/D16-1244.

Pennington, J., Socher, R., & Manning, C. D. (2014). Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)* (pp. 1532--1543). URL: http://www.aclweb.org/anthology/D14-1162.

Rajpurkar, P., Zhang, J., Lopyrev, K., & Liang, P. (2016). Squad: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing* (pp. 2383--2392). Association for Computational Linguistics. URL: http://www.aclweb.org/anthology/D16-1264. doi:10.18653/v1/D16-1264.

Riordan, B., Horbach, A., Cahill, A., Zesch, T., & Lee, C. M. (2017). Investigating neural architectures for short answer scoring. In *Proceedings of the 12th Workshop on Innovative Use of NLP for Building Educational Applications* (pp. 159--168).

Shen, T., Zhou, T., Long, G., Jiang, J., Wang, S., & Zhang, C. (2018). Reinforced self-attention network: a hybrid of hard and soft attention for sequence modeling. *arXiv preprint arXiv:1801.10296*, .

Stolcke, A. (2002). Srilm-an extensible language modeling toolkit. In *Seventh international conference on spoken language processing*.

Tai, K. S., Socher, R., & Manning, C. D. (2015). Improved semantic representations from tree-structured long short-term memory networks. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)* (pp. 1556--1566). Beijing, China: Association for Computational Linguistics. URL: http://www.aclweb.org/anthology/P15-1150.

White, A. S., Rastogi, P., Duh, K., & Van Durme, B. (2017). Inference is everything: Recasting semantic resources into a unified evaluation framework. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)* (pp. 996--1005). Taipei, Taiwan: Asian Federation of Natural Language Processing. URL: http://www.aclweb.org/anthology/I17-1100.

Williams, A., Nangia, N., & Bowman, S. R. (2017). A broad-coverage challenge corpus for sentence understanding through inference. *CoRR*, *abs/1704.05426*. URL: http://arxiv.org/abs/1704.05426.

Yang, Y., Yih, W.-t., & Meek, C. (2015). Wikiqa: A challenge dataset for open-domain question answering. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing* (pp. 2013--2018). Association for Computational

Linguistics. URL: http://www.aclweb.org/anthology/D15-1237. doi:10.18653/v1/D15-1237.

Yin, W., Schtze, H., Xiang, B., & Zhou, B. (2016). Abcnn: Attention-based convolutional neural network for modeling sentence pairs. *Transactions of the Association for Computational Linguistics*, *4*, 259--272. URL: https://tacl2013.cs.columbia.edu/ojs/index.php/tacl/article/view/831.

Zhao, J., Zhu, T., & Lan, M. (2014). Ecnu: One stone two birds: Ensemble of heterogenous measures for semantic relatedness and textual entailment. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)* (pp. 271--277). Dublin, Ireland: Association for Computational Linguistics and Dublin City University. URL: http://www.aclweb.org/anthology/S14-2044.

Zhao, Z., Liu, T., Li, S., Li, B., & Du, X. (2017). Ngram2vec: Learning improved word representations from ngram co-occurrence statistics. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing* (pp. 244--253). Copenhagen, Denmark: Association for Computational Linguistics. URL: https://www.aclweb.org/anthology/D17-1023.

Appendix A. Hyper-parameter values

While Table 3 shows the hyper-parameters for the proposed technique, the tables in this appendix (A.9, A.10, A.11 and A.12) summarize the hyper-parameters for the baseline models tested in this work.

	Multi NLI	SNLI	SICK-TE	SICK-TS	STSBenchmark
Embedding size	300	300	300	300	300
Hidden size	300	300	300	200	250
Weight decay	5e-5	5e-5	5e-5	5e-5	5e-5
Max grad norm	5.0	5.0	5.0	5.0	5.0
Dropout	0.1	0.1	0.2	0.2	0.2
Param init	1e-2	1e-2	1e-2	1e-2	1e-2
Learning rate	5e-2	5e-2	5e-2	2.5e-2	2.5e-2
Epochs	97	76	29	23	11
Optimizer	Adagrad	Adagrad	Adagrad	Adagrad	Adagrad

Table A.9: Hyper-parameters for DAM BoW.

	Multi NLI	SNLI	SICK-TE	SICK-TS	STSBenchmark
Embedding size	300	300	300	300	300
Hidden size	500	500	500	400	200
Weight decay	5e-5	5e-5	5e-5	2e-5	2e-5
Max grad norm	5	5	5	8	8
Dropout	0.25	0.25	0.5	0.5	0.5
Param init	1e-2	1e-2	1e-2	1e-2	1e-2
Learning rate	4e-2	6e-2	2.5e-2	3.5e-2	3e-2
Epochs	71	63	25	33	9
Optimizer	Adagrad	Adagrad	Adagrad	Adagrad	Adagrad

Table A.10: Hyper-parameters for DAM BoW with FF attention.

	Multi NLI	SNLI	SICK-TE	SICK-TS	STSBenchmark
Embedding size	300	300	300	300	300
Hidden size	300	1150	550	350	600
Weight decay	5e-5	4.9e-5	5.9e-5	4e-5	5e-5
Max grad norm	5.0	5.0	5.0	5.0	5.0
Dropout	0.15	0.25	0.25	0.25	0.25
Param init	1e-2	1e-2	1e-2	1e-2	1e-2
Learning rate	5e-2	6e-2	1.8e-2	1.5e-2	7e-2
Epochs	97	74	18	22	10
Optimizer	Adagrad	Adagrad	Adagrad	Adagrad	Adagrad

Table A.11: Hyper-parameters for DAM RNN with FF attention.

	Multi NLI	SNLI	SICK-TE	SICK-TS	STSBenchmark
Embedding size	300	300	300	300	300
Hidden size	385	300	250	600	250
Weight decay	6.7e-5	5.5e-5	5e-5	5e-6	5e-6
Max grad norm	8.0	8.0	5.0	5.0	1.0
Dropout	0.15	0.15	0.15	0.4	0.4
Param init	8e-2	4.5e-2	1e-1	5e-2	1e-1
Learning rate	3.4e-2	3.7e-2	2e-2	1e-2	1.5e-2
Epochs	35	57	57	15	6
Optimizer	Adagrad	Adagrad	Adagrad	Adagrad	Adagrad

Table A.12: Hyper-parameters for DAM RNN with FF attention.