Attentive Convolution:
Equipping CNNs with RNN-style Attention Mechanisms

Wenpeng Yin
Department of Computer and Information
Science, University of Pennsylvania
wenpeng@seas.upenn.edu

Hinrich Schütze
Center for Information and Language
Processing, LMU Munich, Germany
inquiries@cislmu.org

Abstract

In NLP, convolutional neural networks (CNNs) have benefited less than recurrent neural networks (RNNs) from attention mechanisms. We hypothesize that this is because the attention in CNNs has been mainly implemented as attentive pooling (i.e., it is applied to pooling) rather than as attentive convolution (i.e., it is integrated into convolution). Convolution is the differentiator of CNNs in that it can powerfully model the higher-level representation of a word by taking into account its local fixed-size context in the input text t^x. In this work, we propose an attentive convolution network, ATTCONV. It extends the context scope of the convolution operation, deriving higher-level features for a word not only from local context, but also from information extracted from nonlocal context by the attention mechanism commonly used in RNNs. This nonlocal context can come (i) from parts of the input text t^x that are distant or (ii) from *extra* (i.e., external) contexts t^y. Experiments on sentence modeling with zero-context (sentiment analysis), single-context (textual entailment) and multiple-context (claim verification) demonstrate the effectiveness of ATTCONV in sentence representation learning with the incorporation of context. In particular, attentive convolution outperforms attentive pooling and is a strong competitor to popular attentive RNNs.[1]

1 Introduction

Natural language processing (NLP) has benefited greatly from the resurgence of deep neural networks (DNNs), thanks to their high performance with less need of engineered features. A DNN typically is composed of a stack of non-linear transformation layers, each generating a hidden representation for the input by projecting the output of a preceding layer into a new space. To date, building a single and static representation to express an input across diverse problems is far from satisfactory. Instead, it is preferable that the representation of the input vary in different application scenarios. In response, attention mechanisms (Graves, 2013; Graves et al., 2014) have been proposed to dynamically focus on parts of the input that are expected to be more specific to the problem. They are mostly implemented based on fine-grained alignments between two pieces of objects, each emitting a dynamic soft-selection to the components of the other, so that the selected elements dominate in the output hidden representation. Attention-based DNNs have demonstrated good performance on many tasks.

Convolutional neural networks (CNNs; LeCun et al., 1998) and recurrent neural networks (RNNs; Elman, 1990) are two important types of DNNs. Most work on attention has been done for RNNs. Attention-based RNNs typically take three types of inputs to make a decision at the current step: (i) the current input state, (ii) a representation of local context (computed unidirectionally or bidirectionally; Rocktäschel et al. [2016]), and (iii) the attention-weighted sum of hidden states corresponding to nonlocal context (e.g., the hidden states of the encoder in neural machine translation; Bahdanau et al. [2015]). An important question, therefore, is whether CNNs can benefit from such an attention mechanism as well, and how. This is our technical motivation.

Our second motivation is natural language understanding. In generic sentence modeling *without extra context* (Collobert et al., 2011; Kalchbrenner et al., 2014; Kim, 2014), CNNs learn sentence representations by composing word representations that are conditioned on a local context window. We believe that attentive convolution is needed

---

premise, modeled as context t^y	
Plant cells have structures that animal cells lack.	0
Animal cells do not have cell walls.	1
The cell wall is not a freestanding structure.	0
Plant cells possess a cell wall, animals never.	1

Table 1: Examples of four premises for the hypothesis t^x = "*A cell wall is not present in animal cells.*" in SCITAIL data set. Right column (hypothesis's label): "1" means *true*, "0" otherwise.

for some natural language understanding tasks that are essentially sentence modeling within contexts. Examples: textual entailment (is a hypothesis true given a premise as the single context?; Dagan et al. [2013]) and claim verification (is a claim correct given extracted evidence snippets from a text corpus as the context?; Thorne et al. [2018]). Consider the SCITAIL (Khot et al., 2018) textual entailment examples in Table 1; here, the input text t^x is the hypothesis and each premise is a context text t^y. And consider the illustration of claim verification in Figure 1; here, the input text t^x is the claim and t^y can consist of multiple pieces of context. In both cases, we would like the representation of t^x to be context-specific.

In this work, we propose attentive convolution networks, ATTCONV, to model a sentence (i.e., t^x) either in intra-context (where $t^y = t^x$) or extra-context (where $t^y \neq t^x$ and t^y can have many pieces) scenarios. In the **intra-context** case (sentiment analysis, for example), ATTCONV extends the local context window of standard CNNs to cover the entire input text t^x. In the **extra-context** case, ATTCONV extends the local context window to cover accompanying contexts t^y.

For a convolution operation over a window in t^x such as ($left_{\text{context}}$, *word*, $right_{\text{context}}$), we first compare the representation of *word* with all hidden states in the context t^y to obtain an attentive context representation att_{context}, then convolution filters derive a higher-level representation for *word*, denoted as $word_{\text{new}}$, by integrating *word* with three pieces of context: $left_{\text{context}}$, $right_{\text{context}}$, and att_{context}. We interpret this attentive convolution in two perspectives. (i) For intra-context, a higher-level word representation $word_{\text{new}}$ is learned by considering the local (i.e., $left_{\text{context}}$ and $right_{\text{context}}$) as well as nonlocal (i.e., att_{context}) context. (ii) For extra-context, $word_{\text{new}}$ is generated to represent *word*, together with its cross-text alignment att_{context}, in the context $left_{\text{context}}$ and $right_{\text{context}}$. In other words, the decision for the *word* is made based on the connected
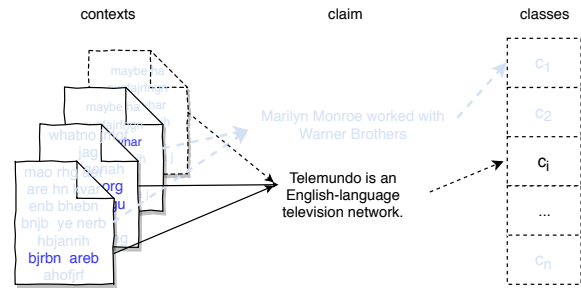

Figure 1: Verify claims in contexts.

hidden states of cross-text aligned terms, with local context.

We apply ATTCONV to three sentence modeling tasks with variable-size context: a large-scale Yelp sentiment classification task (Lin et al., 2017) (intra-context, i.e., no additional context), SCITAIL textual entailment (Khot et al., 2018) (single extra-context), and claim verification (Thorne et al., 2018) (multiple extra-contexts). ATTCONV outperforms competitive DNNs with and without attention and achieves state-of-the-art on the three tasks.

Overall, we make the following contributions:

- This is the first work that equips convolution filters with the attention mechanism commonly used in RNNs.
- We distinguish and build flexible modules—attention source, attention focus, and attention beneficiary—to greatly advance the expressivity of attention mechanisms in CNNs.
- ATTCONV provides a new way to broaden the originally constrained scope of filters in conventional CNNs. Broader and richer context comes from either external context (i.e., t^y) or the sentence itself (i.e., t^x).
- ATTCONV shows its flexibility and effectiveness in sentence modeling with variable-size context.

2 Related Work

In this section we discuss attention-related DNNs in NLP, the most relevant work for our paper.

2.1 RNNs with Attention

Graves (2013) and Graves et al. (2014) first introduced a differentiable attention mechanism that allows RNNs to focus on different parts of the input. This idea has been broadly explored in RNNs, shown in Figure 2, to deal with **text generation**, such as neural machine translation
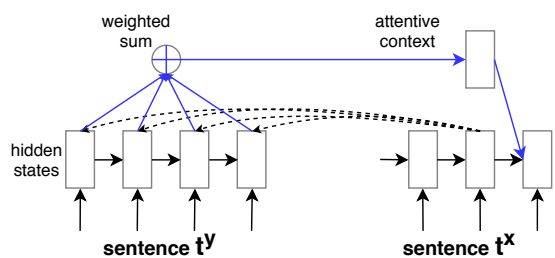
Figure 2: A simplified illustration of attention mechanism in RNNs.

(Bahdanau et al., 2015; Luong et al., 2015; Kim et al., 2017; Libovický and Helcl, 2017), response generation in social media (Shang et al., 2015), document reconstruction (Li et al., 2015), and document summarization (Nallapati et al., 2016); machine comprehension (Hermann et al., 2015; Kumar et al., 2016; Xiong et al., 2016; Seo et al., 2017; Wang and Jiang, 2017; Xiong et al., 2017; Wang et al., 2017a); and sentence relation classification, such as textual entailment (Cheng et al., 2016; Rocktäschel et al., 2016; Wang and Jiang, 2016; Wang et al., 2017b; Chen et al., 2017b) and answer sentence selection (Miao et al., 2016).

We try to explore the RNN-style attention mechanisms in CNNs—more specifically, in convolution.

2.2 CNNs with Attention

In NLP, there is little work on attention-based CNNs. Gehring et al. (2017) propose an attention-based convolutional seq-to-seq model for machine translation. Both the encoder and decoder are hierarchical convolution layers. At the nth layer of the decoder, the output hidden state of a convolution queries each of the encoder-side hidden states, then a weighted sum of all encoder hidden states is added to the decoder hidden state, and finally this updated hidden state is fed to the convolution at layer $n + 1$. Their attention implementation relies on the existence of a multi-layer convolution structure—otherwise the weighted context from the encoder side could not play a role in the decoder. So essentially their attention is achieved *after* convolution. In contrast, we aim to modify the vanilla convolution, so that CNNs with attentive convolution can be applied more broadly.

We discuss two systems that are representative of CNNs that implement the attention *in pooling* (i.e., the convolution is still not affected): Yin et al. (2016) and dos Santos et al. (2016), illustrated in Figure 3. Specifically, these two systems work on two input sentences, each with a set of
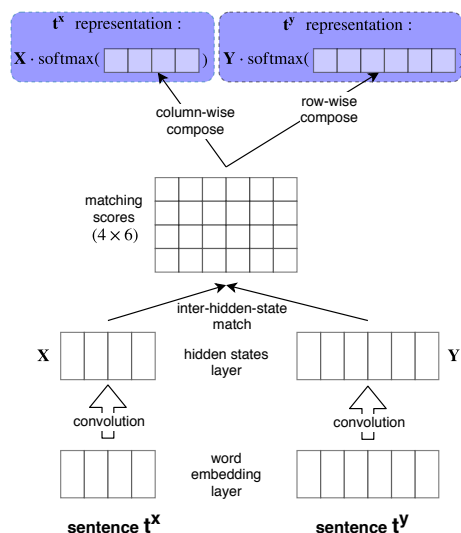

Figure 3: Attentive pooling, summarized from ABCNN (Yin et al., 2016) and APCNN (dos Santos et al., 2016).

hidden states generated by a convolution layer; then, each sentence will learn a weight for every hidden state by comparing this hidden state with all hidden states in the other sentence; finally, each input sentence obtains a representation by a weighted mean pooling over all its hidden states. The core component—weighted mean pooling—was referred to as "attentive pooling," aiming to yield the sentence representation.

In contrast to attentive convolution, attentive pooling does not connect directly the hidden states of cross-text aligned phrases in a fine-grained manner to the final decision making; only the matching scores contribute to the final weighting in mean pooling. This important distinction between attentive convolution and attentive pooling is further discussed in Section 3.3.

Inspired by the attention mechanisms in RNNs, we assume that it is the *hidden states of aligned phrases* rather than *their matching scores* that can better contribute to representation learning and decision making. Hence, our *attentive convolution differs from attentive pooling* in that it uses *attended hidden states from extra context* (i.e., t^y) *or broader-range context* within t^x to participate in the convolution. In experiments, we will show its superiority.

3 ATTCONV Model

We use bold uppercase (e.g., $\mathbf{H}$) for matrices; bold lowercase (e.g., $\mathbf{h}$) for vectors; bold lowercase with index (e.g., $\mathbf{h}_i$) for columns of $\mathbf{H}$; and non-bold lowercase for scalars.
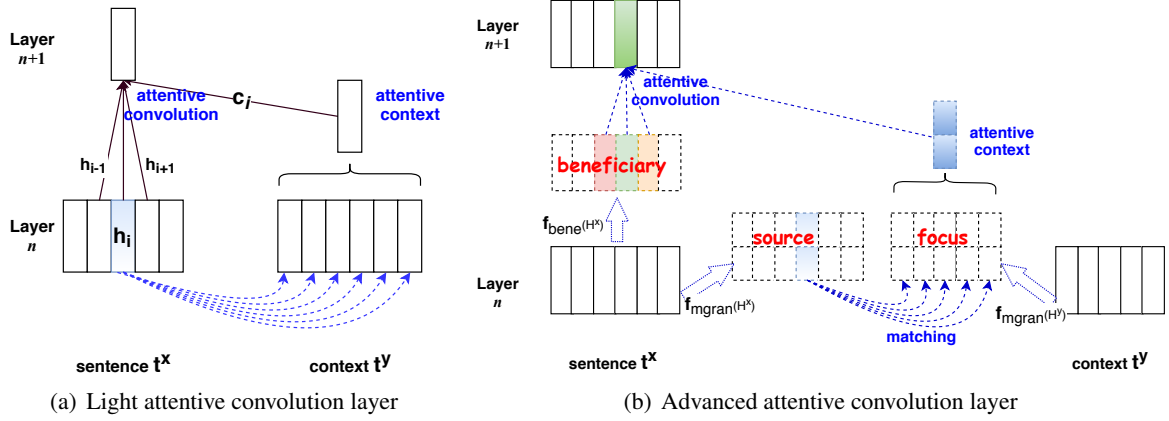
Figure 4: ATTCONV models sentence t^x with context t^y.

(a) Light attentive convolution layer   (b) Advanced attentive convolution layer

To start, we assume that a piece of text t ($t \in \{t^x, t^y\}$) is represented as a sequence of hidden states $\mathbf{h}_i \in \mathbb{R}^d$ ($i = 1, 2, \ldots, |t|$), forming feature map $\mathbf{H} \in \mathbb{R}^{d \times |t|}$, where d is the dimensionality of hidden states. Each hidden state $\mathbf{h}_i$ has its left context $\mathbf{l}_i$ and right context $\mathbf{r}_i$. In concrete CNN systems, contexts $\mathbf{l}_i$ and $\mathbf{r}_i$ can cover multiple adjacent hidden states; we set $\mathbf{l}_i = \mathbf{h}_{i-1}$ and $\mathbf{r}_i = \mathbf{h}_{i+1}$ for simplicity in the following description.

We now describe light and advanced versions of ATTCONV. Recall that ATTCONV aims to compute a representation for t^x in a way that convolution filters encode not only local context, but also attentive context over t^y.

3.1 Light ATTCONV

Figure 4(a) shows the light version of ATTCONV. It differs in two key points—(i) and (ii)—both from the basic convolution layer that models a single piece of text and from the Siamese CNN that models two text pieces in parallel. (i) A matching function determines how relevant each hidden state in the context t^y is to the current hidden state $\mathbf{h}_i^x$ in sentence t^x. We then compute an average of the hidden states in the context t^y, weighted by the matching scores, to get the attentive context $\mathbf{c}_i^x$ for $\mathbf{h}_i^x$. (ii) The convolution for position i in t^x integrates hidden state $\mathbf{h}_i^x$ with *three sources of context*: left context $\mathbf{h}_{i-1}^x$, right context $\mathbf{h}_{i+1}^x$, and attentive context $\mathbf{c}_i^x$.

Attentive Context. First, a function generates a matching score $e_{i,j}$ between a hidden state in t^x and a hidden state in t^y by (i) dot product:

$$e_{i,j} = (\mathbf{h}_i^x)^T \cdot \mathbf{h}_j^y \quad (1)$$

or (ii) bilinear form:

$$e_{i,j} = (\mathbf{h}_i^x)^T \mathbf{W}_e \mathbf{h}_j^y \quad (2)$$

(where $\mathbf{W}_e \in \mathbb{R}^{d \times d}$), or (iii) additive projection:

$$e_{i,j} = (\mathbf{v}_e)^T \cdot \tanh(\mathbf{W}_e \cdot \mathbf{h}_i^x + \mathbf{U}_e \cdot \mathbf{h}_j^y) \quad (3)$$

where $\mathbf{W}_e, \mathbf{U}_e \in \mathbb{R}^{d \times d}$ and $\mathbf{v}_e \in \mathbb{R}^d$.

Given the matching scores, the attentive context $\mathbf{c}_i^x$ for hidden state $\mathbf{h}_i^x$ is the weighted average of all hidden states in t^y:

$$\mathbf{c}_i^x = \sum_j \mathrm{softmax}(\mathbf{e}_i)_j \cdot \mathbf{h}_j^y \quad (4)$$

We refer to the concatenation of attentive contexts $[\mathbf{c}_1^x; \ldots; \mathbf{c}_i^x; \ldots; \mathbf{c}_{|t^x|}^x]$ as the feature map $\mathbf{C}^x \in \mathbb{R}^{d \times |t^x|}$ for t^x.

Attentive Convolution. After attentive context has been computed, a position i in the sentence t^x has a hidden state $\mathbf{h}_i^x$, the left context $\mathbf{h}_{i-1}^x$, the right context $\mathbf{h}_{i+1}^x$, and the attentive context $\mathbf{c}_i^x$. Attentive convolution then generates the higher-level hidden state at position i:

$$\mathbf{h}_{i,\mathrm{new}}^x = \tanh(\mathbf{W} \cdot [\mathbf{h}_{i-1}^x, \mathbf{h}_i^x, \mathbf{h}_{i+1}^x, \mathbf{c}_i^x] + \mathbf{b}) \quad (5)$$

$$= \tanh(\mathbf{W}^1 \cdot [\mathbf{h}_{i-1}^x, \mathbf{h}_i^x, \mathbf{h}_{i+1}^x] + \mathbf{W}^2 \cdot \mathbf{c}_i^x + \mathbf{b}) \quad (6)$$

where $\mathbf{W} \in \mathbb{R}^{d \times 4d}$ is the concatenation of $\mathbf{W}^1 \in \mathbb{R}^{d \times 3d}$ and $\mathbf{W}^2 \in \mathbb{R}^{d \times d}$, $\mathbf{b} \in \mathbb{R}^d$.

As Equation (6) shows, Equation (5) can be achieved by summing up the results of two separate and parallel convolution steps before the non-linearity. The first is still a standard convolution-without-attention over feature map $\mathbf{H}^x$ by filter width 3 over the window ($\mathbf{h}_{i-1}^x$, $\mathbf{h}_i^x$, $\mathbf{h}_{i+1}^x$). The second is a convolution on the feature map $\mathbf{C}^x$ (i.e., the attentive context) with filter width 1 (i.e., over each $\mathbf{c}_i^x$); then we sum up the

690

role	text
premise	Three firefighters **come out** of subway station
hypothesis	Three firefighters **putting out** a fire **inside** of a subway station

Table 2: Multi-granular alignments required in textual entailment.

results element-wise and add a bias term and the non-linearity. This divide-then-compose strategy makes the attentive convolution easy to implement in practice, with no need to create a new feature map, as required in Equation (5), to integrate $\mathbf{H}^x$ and $\mathbf{C}^x$.

It is worth mentioning that $\mathbf{W}^1 \in \mathbb{R}^{d \times 3d}$ corresponds to the filter parameters of a vanilla CNN and the only added parameter here is $\mathbf{W}^2 \in \mathbb{R}^{d \times d}$, which only depends on the hidden size.

This light ATTCONV shows the basic principles of using RNN-style attention mechanisms in convolution. Our experiments show that this light version of ATTCONV—even though it incurs a limited increase of parameters (i.e., $\mathbf{W}^2$)—works much better than the vanilla Siamese CNN and some of the pioneering attentive RNNs. The following two considerations show that there is space to improve its expressivity.

(i) *Higher-level or more abstract representations are required in subsequent layers.* We find that directly forwarding the hidden states in t^x or t^y to the matching process does not work well in some tasks. Pre-learning some more high-level or abstract representations helps in subsequent learning phases.

(ii) *Multi-granular alignments are preferred in the interaction modeling between t^x and t^y.* Table 2 shows another example of textual entailment. On the unigram level, "out" in the premise matches with "out" in the hypothesis perfectly, whereas "out" in the premise is contradictory to "inside" in the hypothesis. But their context snippets—"come out" in the premise and "putting out a fire" in the hypothesis—clearly indicate that they are not semantically equivalent. And the gold conclusion for this pair is "neutral" (i.e., the hypothesis is possibly true). Therefore, matching should be conducted across phrase granularities.

We now present advanced ATTCONV. It is more expressive and modular, based on the two foregoing considerations (i) and (ii).

3.2 Advanced ATTCONV

Adel and Schütze (2017) distinguish between focus and source of attention. The **focus** of atten-

tion is the layer of the network that is reweighted by attention weights. The **source** of attention is the information source that is used to compute the attention weights. Adel and Schütze showed that increasing the scope of the attention source is beneficial. It possesses some preliminary principles of the query/key/value distinction by Vaswani et al. (2017). Here, we further extend this principle to define **beneficiary** of attention – the feature map (labeled "beneficiary" in Figure 4(b)) that is contextualized by *the attentive context* (labeled "attentive context" in Figure 4(b)). In the light attentive convolutional layer (Figure 4(a)), the source of attention is hidden states in sentence t^x, the focus of attention is hidden states of the context t^y, and the beneficiary of attention is again the hidden states of t^x; that is, it is identical to the source of attention.

We now try to distinguish these three concepts further to promote the expressivity of an attentive convolutional layer. We call it "advanced ATTCONV"; see Figure 4(b). It differs from the light version in three ways: (i) *attention source* is learned by function $f_{\text{mgran}}(\mathbf{H}^x)$, feature map $\mathbf{H}^x$ of t^x acting as input; (ii) *attention focus* is learned by function $f_{\text{mgran}}(\mathbf{H}^y)$, feature map $\mathbf{H}^y$ of context t^y acting as input; and (iii) *attention beneficiary* is learned by function $f_{\text{bene}}(\mathbf{H}^x)$, $\mathbf{H}^x$ acting as input. Both functions $f_{\text{mgran}}()$ and $f_{\text{bene}}()$ are based on a *gated convolutional function* $f_{\text{gconv}}()$:

$$\overline{\mathbf{o}}_i = \tanh(\mathbf{W}_h \cdot \mathbf{i}_i + \mathbf{b}_h) \quad (7)$$

$$\mathbf{g}_i = \text{sigmoid}(\mathbf{W}_g \cdot \mathbf{i}_i + \mathbf{b}_g) \quad (8)$$

$$f_{\text{gconv}}(\mathbf{i}_i) = \mathbf{g}_i \cdot \mathbf{u}_i + (1 - \mathbf{g}_i) \cdot \overline{\mathbf{o}}_i \quad (9)$$

where $\mathbf{i}_i$ is a composed representation, denoting a generally defined input phrase $[\cdots, \mathbf{u}_i, \cdots]$ of arbitrary length with $\mathbf{u}_i$ as the central unigram-level hidden state, and the gate $\mathbf{g}_i$ sets a trade-off between the unigram-level input $\mathbf{u}_i$ and the temporary output $\overline{\mathbf{o}}_i$ at the phrase-level. We elaborate these modules in the remainder of this subsection.

Attention Source. First, we present a general instance of generating source of attention by function $f_{\text{mgran}}(\mathbf{H})$, learning word representations in multi-granular context. In our system, we consider granularities 1 and 3, corresponding to unigram hidden state and trigram hidden state. For the uni-hidden state case, it is a gated convolution layer:

$$\mathbf{h}_{\text{uni},i}^x = f_{\text{gconv}}(\mathbf{h}_i^x) \quad (10)$$

For the tri-hidden state case:

$$\mathbf{h}_{\text{tri},i}^{x} = f_{\text{gconv}}([\mathbf{h}_{i-1}^{x}, \mathbf{h}_i^{x}, \mathbf{h}_{i+1}^{x}]) \qquad (11)$$

Finally, the overall hidden state at position i is the concatenation of $\mathbf{h}_{\text{uni},i}$ and $\mathbf{h}_{\text{tri},i}$:

$$\mathbf{h}_{\text{mgran},i}^{x} = [\mathbf{h}_{\text{uni},i}^{x}, \mathbf{h}_{\text{tri},i}^{x}] \qquad (12)$$

that is, $f_{\text{mgran}}(\mathbf{H}^x) = \mathbf{H}_{\text{mgran}}^x$.

Such a kind of comprehensive hidden state can encode the semantics of multigranular spans at a position, such as "out" and "come out of." Gating here implicitly enables cross-granular alignments in subsequent attention mechanism as it sets highway connections (Srivastava et al., 2015) between the input granularity and the output granularity.

Attention Focus. For simplicity, we use the same architecture for the *attention source* (just introduced) and for the *attention focus*, t^y (i.e., for the attention focus: $f_{\text{mgran}}(\mathbf{H}^y) = \mathbf{H}_{\text{mgran}}^y$; see Figure 4(b)). Thus, the focus of attention will participate in the matching process as well as be reweighted to form an attentive context vector. We leave exploring different architectures for attention source and focus for future work.

Another benefit of multi-granular hidden states in attention focus is to keep structure information in the context vector. In standard attention mechanisms in RNNs, all hidden states are average-weighted as a context vector, and the order information is missing. By introducing hidden states of larger granularity into CNNs that keep the local order or structures, we boost the attentive effect.

Attention Beneficiary. In our system, we simply use $f_{\text{gconv}}()$ over uni-granularity to learn a more abstract representation over the current hidden representations in $\mathbf{H}^x$, so that

$$f_{\text{bene}}(\mathbf{h}_i^x) = f_{\text{gconv}}(\mathbf{h}_i^x) \qquad (13)$$

Subsequently, the *attentive context vector* $\mathbf{c}_i^x$ is generated based on *attention source* feature map $f_{\text{mgran}}(\mathbf{H}^x)$ and *attention focus* feature map $f_{\text{mgran}}(\mathbf{H}^y)$, according to the description of the light ATTCONV. Then attentive convolution is conducted over *attention beneficiary* feature map $f_{\text{bene}}(\mathbf{H}^x)$ and the attentive context vectors $\mathbf{C}^x$ to get a higher-layer feature map for the sentence t^x.

3.3 Analysis

Compared with the standard attention mechanism in RNNs, ATTCONV has a similar matching function and a similar process of computing context vectors, but differs in three ways. (i) The discrimination of attention source, focus, and beneficiary improves expressivity. (ii) In CNNs, the surrounding hidden states for a concrete position are available, so the attention matching is able to encode the left context as well as the right context. In RNNs, however, we need bidirectional RNNs to yield both left and right context representations. (iii) As attentive convolution can be implemented by summing up two separate convolution steps (Equations 5 and 6), this architecture provides both attentive representations and representations computed without the use of attention. This strategy is helpful in practice to use richer representations for some NLP problems. In contrast, such a clean modular separation of representations computed with and without attention is harder to realize in attention-based RNNs.

Prior attention mechanisms explored in CNNs mostly involve attentive pooling (dos Santos et al., 2016; Yin et al., 2016); namely, the weights of the post-convolution pooling layer are determined by attention. These weights come from the matching process between hidden states of two text pieces. However, a weight value is not informative enough to tell the relationships between aligned terms. Consider a textual entailment sentence pair for which we need to determine whether "inside $\longrightarrow$ outside" holds. The matching degree (take cosine similarity as example) of these two words is high: for example, ≈ 0.7 in Word2Vec (Mikolov et al., 2013) and GloVe (Pennington et al., 2014). On the other hand, the matching score between "inside" and "in" is lower: 0.31 in Word2Vec, 0.46 in GloVe. Apparently, the higher number 0.7 does not mean that "outside" is more likely than "in" to be entailed by "inside." Instead, joint representations for aligned phrases [$\mathbf{h}_{\text{inside}}$, $\mathbf{h}_{\text{outside}}$], [$\mathbf{h}_{\text{inside}}$, $\mathbf{h}_{\text{in}}$] are more informative and enable finer-grained reasoning than a mechanism that can only transmit information downstream by matching scores. We modify the conventional CNN filters so that "inside" can make the entailment decision by looking at the representation of the counterpart term ("outside" or "in") rather than a matching score.

A more damaging property of attentive pooling is the following. Even if matching scores could convey the phrase-level entailment degree to some extent, matching weights, in fact, are not leveraged to make the entailment decision directly;

instead, they are used to weight the sum of the output hidden states of a convolution as the global sentence representation. In other words, fine-grained entailment degrees are likely to be lost in the summation of many vectors. This illustrates why attentive context vectors participating in the convolution operation are expected to be more effective than post-convolution attentive pooling (more explanations in §4.3, paragraph "Visualization").

Intra-context attention and extra-context attention. Figures 4(a) and 4(b) depict the modeling of a sentence t^x with its context t^y. This is a common application of attention mechanism in the literature; we call it extra-context attention. But ATTCONV can also be applied to model a single text input, that is, intra-context attention. Consider a sentiment analysis example: "*With the 2017 NBA All-Star game in the books I think we can all agree that this was definitely one to remember. Not because of the three-point shootout, the dunk contest, or the game itself but because of the ludicrous trade that occurred after the festivities.*" This example contains informative points at different locations ("remember" and "ludicrous"); conventional CNNs' ability to model nonlocal dependency is limited because of fixed-size filter widths. In ATTCONV, we can set $t^y = t^x$. The attentive context vector then accumulates all related parts together for a given position. In other words, our intra-context attentive convolution is able to connect all related spans together to form a comprehensive decision. This is a new way to broaden the scope of conventional filter widths: A filter now covers not only the local window, but also those spans that are related, but are beyond the scope of the window.

Comparison to Transformer.[2] The "focus" in ATTCONV corresponds to "key" and "value" in Transformer; that is, our versions of "key" and "value" are the same, coming from the context sentence. The "query" in Transformer corresponds to the "source" and "beneficiary" of ATTCONV; namely, our model has two perspectives to utilize the context: one acts as a real query (i.e., "source") to attend the context, the other (i.e., "beneficiary") takes the attentive con-

text back to improve the learned representation of itself. If we reduce ATTCONV to unigram convolutional filters, it is pretty much a single Transformer layer (if we neglect the positional encoding in Transformer and unify the "query-key-value" and "source-focus-beneficiary" mechanisms).

4 Experiments

We evaluate ATTCONV on sentence modeling in three scenarios: (i) Zero-context, that is, intra-context; the same input sentence acts as t^x as well as t^y; (ii) Single-context, that is, textual entailment—hypothesis modeling with a single premise as the extra-context; and (iii) Multiple-context, namely, claim verification—claim modeling with multiple extra-contexts.

4.1 Common Set-up and Common Baselines

All experiments share a **common set-up**. The input is represented using 300-dimensional publicly available Word2Vec (Mikolov et al., 2013) embeddings; out of vocabulary embeddings are randomly initialized. The architecture consists of the following four layers in sequence: embedding, attentive convolution, max-pooling, and logistic regression. The context-aware representation of t^x is forwarded to the logistic regression layer. We use AdaGrad (Duchi et al., 2011) for training. Embeddings are fine-tuned during training. Hyperparameter values include: learning rate 0.01, hidden size 300, batch size 50, filter width 3.

All experiments are designed to explore comparisons in three aspects: (i) within ATTCONV, "light" vs. "advanced"; (ii) "attentive convolution" vs. "attentive pooling"/"attention only"; and (iii) "attentive convolution" vs. "attentive RNN".

To this end, we always report "light" and "advanced" ATTCONV performance and compare against five types of **common baselines**: (i) w/o context; (ii) w/o attention; (iii) w/o convolution: Similar to the Transformer's principle (Vaswani et al., 2017), we discard the convolution operation in Equation (5) and forward the addition of the attentive context $\mathbf{c}_i^x$ and the $\mathbf{h}_i^x$ into a fully connected layer. To keep enough parameters, we stack in total four layers so that "w/o convolution" has the same size of parameters as light-ATTCONV; (iv) with attention: RNNs with attention and CNNs with attentive pooling; and (v) prior state of the art, *typeset in italics*.

---

[2]Our "source-focus-beneficiary" mechanism was inspired by Adel and Schütze (2017). Vaswani et al. (2017) later published the Transformer model, which has a similar "query-key-value" mechanism.

	systems	acc
w/o attention	Paragraph Vector	58.43
	Lin et al. Bi-LSTM	61.99
	Lin et al. CNN	62.05
	MultichannelCNN (Kim)	*64.62*
with attention	CNN+internal attention	61.43
	ABCNN	61.36
	APCNN	61.98
	Attentive-LSTM	63.11
	Lin et al. RNN Self-Att.	64.21
ATT CONV	light	66.75
	w/o convolution	61.34
	advanced	**67.36***

Table 3: System comparison of sentiment analysis on Yelp. Significant improvements over state of the art are marked with $*$ (test of equal proportions, $p < 0.05$).
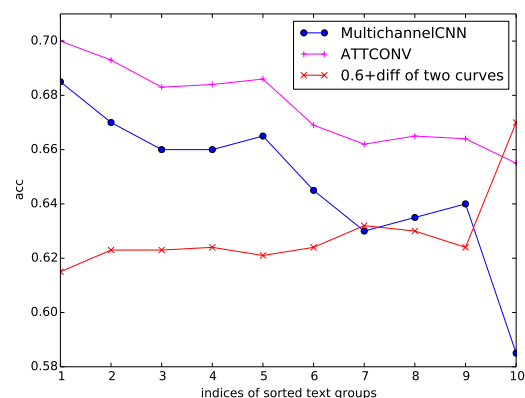

Figure 5: ATTCONV vs. MultichannelCNN for groups of Yelp text with ascending text lengths. ATTCONV performs more robustly across different lengths of text.

4.2 Sentence Modeling with Zero-context: Sentiment Analysis

We evaluate sentiment analysis on a Yelp benchmark released by Lin et al. (2017): review-star pairs in sizes 500K (train), 2,000 (dev), and 2,000 (test). Most text instances in this data set are long: 25%, 50%, 75% percentiles are 46, 81, and 125 words, respectively. The task is five-way classification: 1 to 5 stars. The measure is accuracy. We use this benchmark because the predominance of long texts lets us evaluate the system performance of encoding long-range context, and the system by Lin et al. is directly related to ATTCONV in intra-context scenario.

Baselines. (i) w/o attention. Three baselines from Lin et al. (2017): Paragraph Vector (Le and Mikolov, 2014) (unsupervised sentence representation learning), BiLSTM, and CNN. We also reimplement MultichannelCNN (Kim, 2014), recognized as a simple but surprisingly strong sentence modeler. (ii) with attention. A vanilla "Attentive-LSTM" by Rocktäschel et al. (2016). "RNN Self-Attention" (Lin et al., 2017) is directly comparable to ATTCONV: it also uses intra-context attention. "CNN+internal attention" (Adel and Schütze, 2017), an intra-context attention idea similar to, but less complicated than, Lin et al. (2017). ABCNN & APCNN – CNNs with attentive pooling.

Results and Analysis. Table 3 shows that advanced-ATTCONV surpasses its "light" counterpart, and obtains significant improvement over the state of the art.

In addition, ATTCONV surpasses attentive pooling (ABCNN&APCNN) with a big margin (>5%) and outperforms the representative attentive-LSTM (>4%).

Furthermore, it outperforms the two self-attentive models: CNN+internal attention (Adel and Schütze, 2017) and RNN Self-Attention (Lin et al., 2017), which are specifically designed for single-sentence modeling. Adel and Schütze (2017) generate an attention weight for each CNN hidden state by a linear transformation of the same hidden state, then compute weighted average over all hidden states as the text representation. Lin et al. (2017) extend that idea by generating *a group of* attention weight vectors, then RNN hidden states are averaged by those diverse weighted vectors, allowing extracting different aspects of the text into multiple vector representations. Both works are essentially weighted mean pooling, similar to the attentive pooling in Yin et al. (2016) and dos Santos et al. (2016).

Next, we compare ATTCONV with MultichannelCNN, the strongest baseline system ("w/o attention"), for different length ranges to check whether ATTCONV can really encode long-range context effectively. We sort the 2,000 test instances by length, then split them into 10 groups, each consisting of 200 instances. Figure 5 shows performance of ATTCONV vs. MultichannnelCNN.

We observe that ATTCONV consistently outperforms MultichannelCNN for all lengths. Furthermore, the improvement over MultichannelCNN generally increases with length. This is evidence that ATTCONV more effectively models long text.

	#instances	#entail	#neutral
train	23,596	8,602	14,994
dev	1,304	657	647
test	2,126	842	1,284
total	27,026	10,101	16,925

Table 4: Statistics of SCITAIL data set.

	systems	acc
w/o attention	Majority Class	60.4
	w/o Context	65.1
	Bi-LSTM	69.5
	NGram model	70.6
	Bi-CNN	74.4
with attention	Enhanced LSTM	70.6
	Attentive-LSTM	71.5
	Decomp-Att	72.3
	DGEM	77.3
	APCNN	75.2
	ABCNN	75.8
	ATTCONV-light	78.1
	w/o convolution	75.1
	ATTCONV-advanced	**79.2**

Table 5: ATTCONV vs. baselines on SCITAIL.

This is likely because of ATTCONV's capability to encode broader context in its filters.

4.3 Sentence Modeling with a Single Context: Textual Entailment

Data Set. SCITAIL (Khot et al., 2018) is a textual entailment benchmark designed specifically for a real-world task: multi-choice question answering. All hypotheses t^x were obtained by rephrasing (question, correct answer) pairs into single sentences, and premises t^y are relevant Web sentences retrieved by an information retrieval method. Then the task is to determine whether a hypothesis is *true or not*, given a premise as context. All (t^x, t^y) pairs are annotated via crowdsourcing. Accuracy is reported. Table 1 shows examples and Table 4 gives statistics.

By this construction, a substantial performance improvement on SCITAIL *is equivalent to a better QA performance* (Khot et al., 2018). The hypothesis t^x is the target sentence, and the premise t^y acts as its context.

Baselines. Apart from the common baselines (see Section 4.1), we include systems covered by Khot et al. (2018): (i) n-gram Overlap: An overlap baseline, considering lexical granularity

such as unigrams, one-skip bigrams, and one-skip trigrams. (ii) Decomposable Attention Model (Decomp-Att) (Parikh et al., 2016): Explore attention mechanisms to decompose the task into subtasks to solve in parallel. (iii) Enhanced LSTM (Chen et al., 2017b): Enhance LSTM by taking into account syntax and semantics from parsing information. (iv) DGEM (Khot et al., 2018): A decomposed graph entailment model, the current state-of-the-art.

Table 5 presents results on SCITAIL. (i) Within ATTCONV, "advanced" beats "light" by 1.1%; (ii) "w/o convolution" and attentive pooling (i.e., ABCNN & APCNN) get lower performances by 3%–4%; (iii) More complicated attention mechanisms equipped into LSTM (e.g., "attentive-LSTM" and "enhanced-LSTM") perform even worse.

Error Analysis. To better understand the ATTCONV in SCITAIL, we study some error cases listed in Table 6.

Language conventions. Pair #1 uses sequential commas (i.e., in "the egg, larva, pupa, and adult") or a special symbol sequence (i.e., in "egg $->$ larva $->$ pupa $->$ adult") to form a set or sequence; pair #2 has "A (or B)" to express the equivalence of A and B. This challenge is expected to be handled by DNNs with specific training signals.

Knowledge beyond the text t^y. In #3, "*because smaller amounts of water evaporate in the cool morning*" cannot be inferred from the premise t^y directly. The main challenge in #4 is to distinguish "weight" from "force," which requires background physical knowledge that is beyond the presented text here and beyond the expressivity of word embeddings.

Complex discourse relation. The premise in #5 has an "or" structure. In #6, the inserted phrase "*with about 16,000 species*" makes the connection between "*nonvascular plants*" and "*the mosses, liverworts, and hornworts*" hard to detect. Both instances require the model to decode the discourse relation.

ATTCONV on SNLI. Table 7 shows the comparison. We observe that: (i) classifying hypotheses without looking at premises, that is, "w/o context" baseline, results in a large improvement over the "majority baseline." This verifies the strong bias in the hypothesis construction of the SNLI data set (Gururangan et al., 2018; Poliak et al., 2018). (ii) ATTCONV (advanced) surpasses

695

#	(Premise t^y, Hypothesis t^x) Pair	G/P	Challenge
1	(t^y) These insects have 4 life stages, the egg, larva, pupa, and adult. (t^x) The sequence egg $->$ larva $->$ pupa $->$ adult shows the life cycle of some insects.	1/0	language conventions
2	(t^y) ... the notochord forms the backbone (or vertebral column). (t^x) Backbone is another name for the vertebral column.	1/0	language conventions
3	(t^y) Water lawns early in the morning ... prevent evaporation. (t^x) Watering plants and grass in the early morning is a way to conserve water because smaller amounts of water evaporate in the cool morning.	1/0	beyond text
4	(t^y) ... the SI unit ... for force is the Newton (N) and is defined as (kg·m/s^{-2}). (t^x) Newton (N) is the SI unit for weight.	0/1	beyond text
5	(t^y) Heterotrophs get energy and carbon from living plants or animals (consumers) or from dead organic matter (decomposers). (t^x) Mushrooms get their energy from decomposing dead organisms.	0/1	discourse relation
6	(t^y) ... are a diverse assemblage of three phyla of nonvascular plants, with about 16,000 species, that includes the mosses, liverworts, and hornworts. (t^x) Moss is best classified as a nonvascular plant.	1/0	discourse relation

Table 6: Error cases of ATTCONV in SCITAIL. "...": truncated text. "G/P": gold/predicted label.

	Systems	#para	acc
w/o attention	majority class	0	34.3
	w/o context (i.e., hypothesis only)	270K	68.7
	Bi-LSTM (Bowman et al., 2015)	220K	77.6
	Bi-CNN	270K	80.3
	Tree-CNN (Mou et al., 2016)	3.5M	82.1
	NES (Munkhdalai and Yu, 2017)	6.3M	84.8
with attention	Attentive-LSTM (Rocktäschel)	250K	83.5
	Self-Attentive (Lin et al., 2017)	95M	84.4
	Match-LSTM (Wang and Jiang)	1.9M	86.1
	LSTMN (Cheng et al., 2016)	3.4M	86.3
	Decomp-Att (Parikh)	580K	86.8
	Enhanced LSTM (Chen et al., 2017b)	7.7M	88.6
	ABCNN (Yin et al., 2016)	834K	83.7
	APCNN (dos Santos et al., 2016)	360K	83.9
	ATTCONV – light	360K	86.3
	w/o convolution	360K	84.9
	ATTCONV – advanced	900K	87.8
	State-of-the-art (Peters et al., 2018)	8M	**88.7**

Table 7: Performance comparison on SNLI test. Ensemble systems are not included.

all "w/o attention" baselines and "with attention" CNN baselines (i.e., attentive pooling), obtaining a performance (87.8%) that is close to the state of the art (88.7%).

We also report the parameter size in SNLI as most baseline systems did. Table 7 shows that, in comparison to these baselines, our ATTCONV (light and advanced) has a more limited number of parameters, yet its performance is competitive.

Visualization. In Figure 6, we visualize the attention mechanisms explored in attentive convolution (Figure 6(a)) and attentive pooling (Figure 6(b)).

Figure 6(a) explores the visualization of two kinds of features learned by light ATTCONV in SNLI data set (most are short sentences with rich phrase-level reasoning): (i) $e_{i,j}$ in Equation (1) (after softmax), which shows the *attention distribution* over context t^y by the hidden state h_i^x in sentence t^x; (ii) $\mathbf{h}_{i,\text{new}}^x$ in Equation (5) for $i = 1, 2, \cdots, |t^x|$; it shows the context-aware word features in t^x. By the two visualized features, we can identify which parts of the context t^y are more important for a word in sentence t^x, and a max-pooling, over those context-driven word representations, selects and forwards dominant *(word, left*$_{\text{context}}$, *right*$_{\text{context}}$, *att*$_{\text{context}})$ combinations to the final decision maker.

Figure 6(a) shows the features[3] of sentence t^x = "*A dog jumping for a Frisbee in the snow*" conditioned on the context t^y = "*An animal is outside in the cold weather, playing with a plastic toy.*" Observations: (i) The right figure shows that the attention mechanism successfully aligns some cross-sentence phrases that are informative to the textual entailment problem, such as "dog" to "animal" (i.e., $\mathbf{c}_{dog}^x \approx$ "animal"), "Frisbee" to "plastic toy" and "playing" (i.e., $\mathbf{c}_{Frisbee}^x \approx$ "plastic toy"+"playing"); (ii) The left figure shows a max-pooling over the generated features of filter_1 and filter_2 will focus on the context-aware phrases (A, dog, jumping, $\mathbf{c}_{dog}^x$) and (a,

---

[3]For simplicity, we show 2 out of 300 ATTCONV filters.
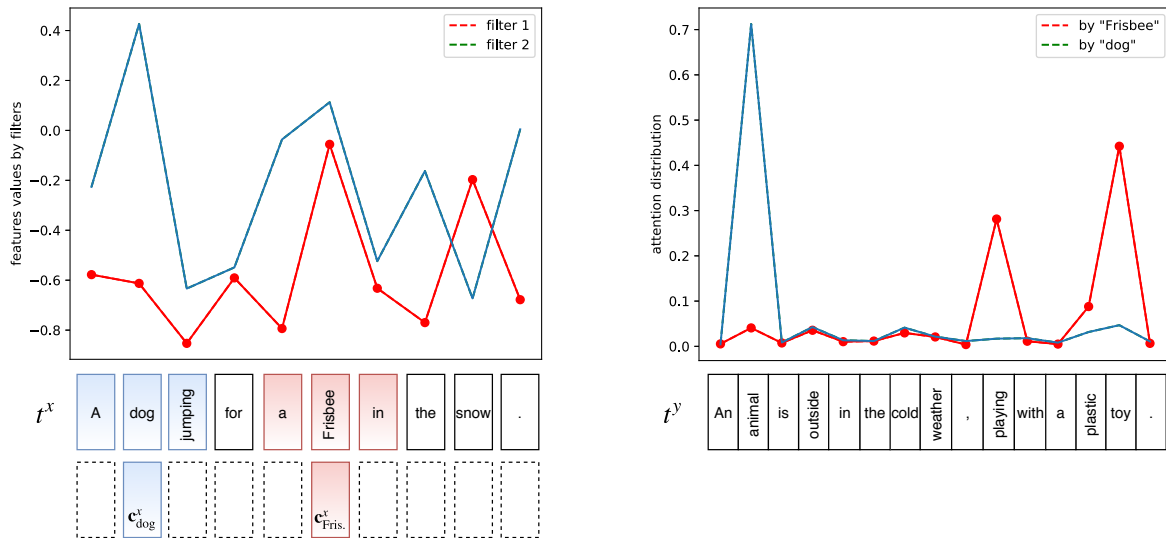
(a) Visualization for features generated by ATTCONV's filters on sentence t^x and t^y. A max-pooling, over filter_1, locates the phrase (A, dog, jumping, $\mathbf{c}^x_{dog}$), and locates the phrase (a, Frisbee, in, $\mathbf{c}^x_{Frisbee}$) via filter_2. "$\mathbf{c}^x_{dog}$" (resp. $\mathbf{c}^x_{Fris.}$)—the attentive context of "dog" (resp. "Frisbee") in t^x—mainly comes from "animal" (resp. "toy" and "playing") in t^y.
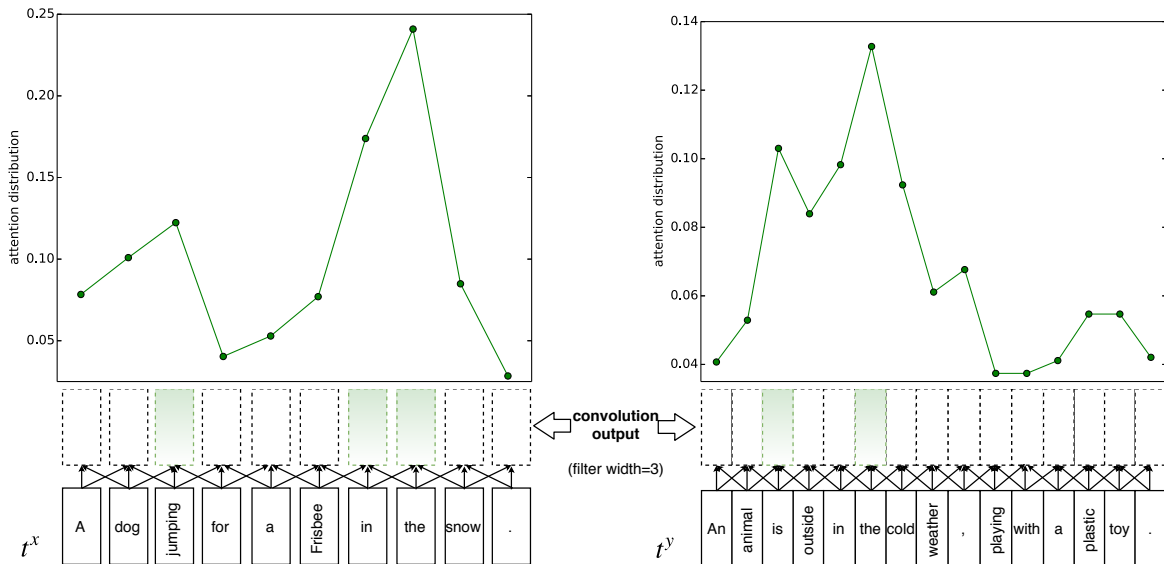
(b) Attention visualization for attentive pooling (ABCNN). Based on the words in t^x and t^y, first, a convolution layer with filter width 3 outputs hidden states for each sentence, then each hidden state will obtain an attention weight for how well this hidden state matches towards all the hidden states in the other sentence, and finally all hidden states in each sentence will be weighted and summed up as the sentence representation. This visualization shows that the spans "dog jumping for" and "in the snow" in t^x and the spans "animal is outside" and "in the cold" in t^y are most indicative to the entailment reasoning.

Figure 6: Attention visualization for attentive convolution (top) and attentive pooling (bottom) between sentence t^x = "*A dog jumping for a Frisbee in the snow*" (left) and sentence t^y = "*An animal is outside in the cold weather, playing with a plastic toy*" (right).

Frisbee, in, $\mathbf{c}^x_{Frisbee}$) respectively; the two phrases are crucial to the entailment reasoning for this (t^y, t^x) pair.

Figure 6(b) shows the phrase-level (i.e., each consecutive trigram) attentions after the convolution operation. As Figure 3 shows, a subsequent pooling step will weight and sum up those phrase-level hidden states as an overall sentence representation. So, even though some phrases such as "in the snow" in t^x and "in the cold" in t^y show importance in this pair instance, the final sentence representation still (i) lacks a fine-grained phrase-to-phrase reasoning, and (ii) underestimates some indicative phrases such as "A dog" in t^x and "An animal" in t^y.

Briefly, attentive convolution first performs phrase-to-phrase, inter-sentence reasoning, then composes features; attentive pooling composes

	#Supported	#Refuted	#NEI
train	80,035	29,775	35,639
dev	3,333	3,333	3,333
test	3,333	3,333	3,333

Table 8: Statistics of claims in the FEVER data set.

phrase features as sentence representations, then performs reasoning. Intuitively, attentive convolution better fits the way humans conduct entailment reasoning, and our experiments validate its superiority—it is the hidden states of the aligned phrases rather than their matching scores that support better representation learning and decision-making.

The comparisons in both SCITAIL and SNLI show that:

- CNNs with attentive convolution (i.e., ATTCONV) outperform the CNNs with attentive pooling (i.e., ABCNN and APCNN);
- Some competitors got over-tuned on SNLI while demonstrating mediocre performance in SCITAIL—a real-world NLP task. Our system ATTCONV shows its robustness in both benchmark data sets.

4.4 Sentence Modeling with Multiple Contexts: Claim Verification

Data Set. For this task, we use FEVER (Thorne et al., 2018); it infers the truthfulness of claims by extracted evidence. The claims in FEVER were manually constructed from the introductory sections of about 50K popular Wikipedia articles in the June 2017 dump. Claims have 9.4 tokens on average. Table 8 lists the claim statistics.

In addition to claims, FEVER also provides a Wikipedia corpus of approximately 5.4 million articles, from which gold evidences are gathered and provided. Figure 7 shows the distributions of sentence sizes in FEVER's ground truth evidence set (i.e., the context size in our experimental set-up). We can see that roughly 28% of evidence instances cover more than one sentence and roughly 16% cover more than two sentences.

Each claim is labeled as SUPPORTED, REFUTED, or NOTENOUGHINFO (NEI) given the gold evidence. The standard FEVER task also explores the performance of evidence extraction, evaluated by F_1 between extracted evidence and gold evidence. This work focuses on the claim entailment part, assuming the evidences are provided (extracted or gold). More specifically, we treat a claim as t^x, and its evidence sentences as context t^y.
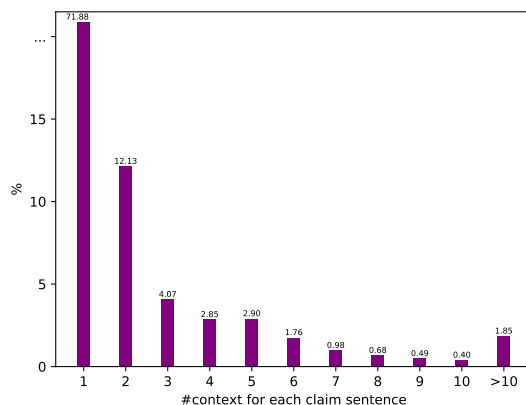

Figure 7: Distribution of #sentence in FEVER evidence.

This task has two evaluations: (i) ALL— accuracy of claim verification regardless of the validness of evidence; (ii) SUBSET—verification accuracy of a subset of claims, in which the gold evidence for SUPPORTED and REFUTED claims must be fully retrieved. We use the official evaluation toolkit.[4]

Set-ups. (i) We adopt the same retrieved evidence set (i.e, contexts t^y) as Thorne et al. (2018): top-5 most relevant sentences from top-5 retrieved wiki pages by a document retriever (Chen et al., 2017a). The quality of this evidence set against the ground truth is: 44.22 (recall), 10.44 (precision), 16.89 (F_1) on dev, and 45.89 (recall), 10.79 (precision), 17.47 (F_1) on test. This set-up challenges our system with potentially unrelated or even misleading context. (ii) We use the ground truth evidence as context. This lets us determine how far our ATTCONV can go for this claim verification problem once the accurate evidence is given.

Baselines. We first include the two systems explored by Thorne et al. (2018): (i) MLP: A multilayer perceptron baseline with a single hidden layer, based on tf-idf cosine similarity between the claim and the evidence (Riedel et al., 2017); (ii) Decomp-Att (Parikh et al., 2016): A decomposable attention model that is tested in SCITAIL and SNLI before. Note that both baselines first relied on an information retrieval system to extract the top-5 relevant sentences from the retrieved top-5 wiki pages as *evidence* for claims, then concatenated all evidence sentences as a longer context for a claim.

---

[4] https://github.com/sheffieldnlp/fever-scorer.

698

| | | retrie. evi. | | gold |
	system	ALL	SUB	evi.
dev	MLP	41.86	19.04	65.13
	Bi-CNN	47.82	26.99	75.02
	APCNN	50.75	30.24	78.91
	ABCNN	51.39	32.44	77.13
	Attentive-LSTM	52.47	33.19	78.44
	Decomp-Att	52.09	32.57	80.82
	ATTCONV			
	light,context-wise	57.78	34.29	83.20
	w/o conv.	47.29	25.94	73.18
	light,context-conc	59.31	37.75	84.74
	w/o conv.	48.02	26.67	73.44
	advan.,context-wise	60.20	37.94	84.99
	advan.,context-conc	**62.26**	**39.44**	**86.02**
test	(Thorne et al., 2018)	50.91	31.87	–
	ATTCONV	**61.03**	**38.77**	**84.61**

Table 9: Performance on *dev* and *test* of FEVER. In "gold evi." scenario, ALL SUBSET are the same.

We then consider two variants of our ATTCONV in dealing with modeling of t^x with variable-size context t^y. (i) *Context-wise*: we first use all evidence sentences one by one as context t^y to guide the representation learning of the claim t^x, generating a group of context-aware representation vectors for the claim, then we do element-wise max-pooling over this vector group as the final representation of the claim. (ii) *Context-conc*: concatenate all evidence sentences as a single piece of context, then model the claim based on this context. This is the same preprocessing step as Thorne et al. (2018) did.

Results. Table 9 compares our ATTCONV in different set-ups against the baselines. First, ATTCONV surpasses the top competitor "Decomp-Att," reported in Thorne et al. (2018), with big margins in *dev* (ALL: 62.26 vs. 52.09) and *test* (ALL: 61.03 vs. 50.91). In addition, "advanced-ATTCONV" consistently outperforms its "light" counterpart. Moreover, ATTCONV surpasses attentive pooling (i.e., ABCNN & APCNN) and "attentive-LSTM" by >10% in ALL, >6% in SUB and >8% in "*gold evi.*"

Figure 8 further explores the fine-grained performance of ATTCONV for different sizes of gold evidence (i.e., different sizes of context t^y). The system shows comparable performances for sizes 1 and 2. Even for context sizes larger than 5, it only drops by 5%.
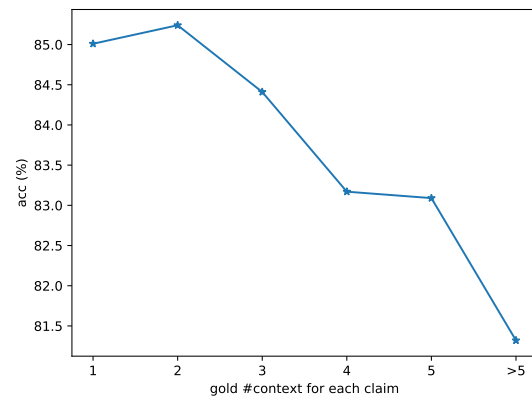

Figure 8: Fine-grained ATTCONV performance given variable-size golden FEVER evidence as claim's context.

These experiments on claim verification clearly show the effectiveness of ATTCONV in sentence modeling with variable-size context. This should be attributed to the attention mechanism in ATTCONV, which enables a word or a phrase in the claim t^x to "see" and accumulate all related clues even if those clues are scattered across multiple contexts t^y.

Error Analysis. We do error analysis for "retrieved evidence" scenario.

Error case #1 is due to the failure of fully retrieving all evidence. For example, a successful *support* of the claim "*Weekly Idol has a host born in the year 1978*" requires the information composition from three evidence sentences, two from the wiki article "Weekly Idol," and one from "Jeong Hyeong-don." However, only one of them is retrieved in the top-5 candidates. Our system predicts REFUTED. This error is more common in instances for which no evidence is retrieved.

Error case #2 is due to the insufficiency of representation learning. Consider the wrong claim "Corsica belongs to Italy" (i.e., in REFUTED class). Even though good evidence is retrieved, the system is misled by noise evidence: "*It is located ... west of the Italian Peninsula, with the nearest land mass being the Italian island ...*".

Error case #3 is due to the lack of advanced data preprocessing. For a human, it is very easy to "refute" the claim "Telemundo is an English-language television network" by the evidence "*Telemundo is an American Spanish-language terrestrial television ...*" (from the "Telemundo" wikipage), by checking the keyphrases: "Spanish-language" vs. "English-language." Unfortunately, both tokens are unknown words in our system; as a result,

they do not have informative embeddings. A more careful data preprocessing is expected to help.

5 Summary

We presented ATTCONV, the first work that enables CNNs to acquire the attention mechanism commonly used in RNNs. ATTCONV combines the strengths of CNNs with the strengths of the RNN attention mechanism. On the one hand, it makes broad and rich context available for prediction, either context from external inputs (extra-context) or internal inputs (intra-context). On the other hand, it can take full advantage of the strengths of convolution: It is more order-sensitive than attention in RNNs and local-context information can be powerfully and efficiently modeled through convolution filters. Our experiments demonstrate the effectiveness and flexibility of ATTCONV when modeling sentences with variable-size context.

Acknowledgments

References

Heike Adel and Hinrich Schütze. 2017. Exploring different dimensions of attention for uncertainty detection. In *Proceedings of EACL*, pages 22–34, Valencia, Spain.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *Proceedings of ICLR*, San Diego, USA.

Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. 2015. A large annotated corpus for learning natural language inference. In *Proceedings of EMNLP*, pages 632–642, Lisbon, Portugal.

Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. 2017a. Reading Wikipedia to answer open-domain questions. In *Proceedings of ACL*, pages 1870–1879, Vancouver, Canada.

Qian Chen, Xiaodan Zhu, Zhen-Hua Ling, Si Wei, Hui Jiang, and Diana Inkpen. 2017b. Enhanced LSTM for natural language inference. In *Proceedings of ACL*, pages 1657–1668, Vancouver, Canada.

Jianpeng Cheng, Li Dong, and Mirella Lapata. 2016. Long short-term memory-networks for machine reading. In *Proceedings of EMNLP*, pages 551–561, Austin, USA.

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel P. Kuksa. 2011. Natural language processing (almost) from scratch. *Journal of Machine Learning Research*, 12:2493–2537.

Ido Dagan, Dan Roth, Mark Sammons, and Fabio Massimo Zanzotto. 2013. *Recognizing Textual Entailment: Models and Applications*. Synthesis Lectures on Human Language Technologies. Morgan & Claypool.

John Duchi, Elad Hazan, and Yoram Singer. 2011. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12:2121–2159.

Jeffrey L. Elman. 1990. Finding structure in time. *Cognitive Science*, 14(2):179–211.

Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, and Yann N. Dauphin. 2017. Convolutional sequence to sequence learning. In *Proceedings of ICML*, pages 1243–1252, Sydney, Australia.

Alex Graves. 2013. Generating sequences with recurrent neural networks. *CoRR*, abs/1308.0850.

Alex Graves, Greg Wayne, and Ivo Danihelka. 2014. Neural turing machines. *CoRR*, abs/1410.5401.

Suchin Gururangan, Swabha Swayamdipta, Omer Levy, Roy Schwartz, Samuel R. Bowman, and Noah A. Smith. 2018. Annotation artifacts in natural language inference data. In *Proceedings of NAACL-HLT*, pages 107–112, New Orleans, USA.

Karl Moritz Hermann, Tomás Kociský, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. In *Proceedings of NIPS*, pages 1693–1701, Montreal, Canada.

Nal Kalchbrenner, Edward Grefenstette, and Phil Blunsom. 2014. A convolutional neural network for modelling sentences. In *Proceedings of ACL*, pages 655–665, Baltimore, USA.

Tushar Khot, Ashish Sabharwal, and Peter Clark. 2018. SciTaiL: A textual entailment dataset from science question answering. In *Proceedings of AAAI*, pages 5189–5197, New Orleans, USA.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *Proceedings of EMNLP*, pages 1746–1751, Doha, Qatar.

Yoon Kim, Carl Denton, Luong Hoang, and Alexander M. Rush. 2017. Structured attention networks. In *Proceedings of ICLR*, Toulon, France.

Ankit Kumar, Ozan Irsoy, Peter Ondruska, Mohit Iyyer, James Bradbury, Ishaan Gulrajani, Victor Zhong, Romain Paulus, and Richard Socher. 2016. Ask me anything: Dynamic memory networks for natural language processing. In *Proceedings of ICML*, pages 1378–1387, New York City, USA.

Quoc Le and Tomas Mikolov. 2014. Distributed representations of sentences and documents. In *Proceedings of ICML*, pages 1188–1196, Beijing, China.

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. 1998. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324.

Jiwei Li, Minh-Thang Luong, and Dan Jurafsky. 2015. A hierarchical neural autoencoder for paragraphs and documents. In *Proceedings of ACL*, pages 1106–1115, Beijing, China.

Jindrich Libovický and Jindrich Helcl. 2017. Attention strategies for multi-source sequence-to-sequence learning. In *Proceedings of ACL*, pages 196–202, Vancouver, Canada.

Zhouhan Lin, Minwei Feng, Cícero Nogueira dos Santos, Mo Yu, Bing Xiang, Bowen Zhou, and Yoshua Bengio. 2017. A structured self-attentive sentence embedding. In *Proceedings of ICLR*, Toulon, France.

Minh-Thang Luong, Hieu Pham, and Christopher D. Manning. 2015. Effective approaches to attention-based neural machine translation. In *Proceedings of EMNLP*, pages 1412–1421, Lisbon, Portugal.

Yishu Miao, Lei Yu, and Phil Blunsom. 2016. Neural variational inference for text processing. In *Proceedings of ICML*, pages 1727–1736, New York City, USA.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Gregory S. Corrado, and Jeffrey Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Proceedings of NIPS*, pages 3111–3119, Lake Tahoe, USA.

Lili Mou, Rui Men, Ge Li, Yan Xu, Lu Zhang, Rui Yan, and Zhi Jin. 2016. Natural language inference by tree-based convolution and heuristic matching. In *Proceedings of ACL*, pages 130–136, Berlin, Germany.

Tsendsuren Munkhdalai and Hong Yu. 2017. Neural semantic encoders. In *Proceedings of EACL*, pages 397–407, Valencia, Spain.

Ramesh Nallapati, Bowen Zhou, Cícero Nogueira dos Santos, Çaglar Gülçehre, and Bing Xiang. 2016. Abstractive text summarization using sequence-to-sequence rnns and beyond. In *Proceedings of CoNLL*, pages 280–290, Berlin, Germany.

Ankur P. Parikh, Oscar Täckström, Dipanjan Das, and Jakob Uszkoreit. 2016. A decomposable attention model for natural language inference. In *Proceedings of EMNLP*, pages 2249–2255, Austin, USA.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings of EMNLP*, pages 1532–1543, Doha, Qatar.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of NAACL-HLT*, pages 2227–2237, New Orleans, USA.

Adam Poliak, Jason Naradowsky, Aparajita Haldar, Rachel Rudinger, and Benjamin Van Durme. 2018. Hypothesis only baselines in natural

language inference. In *Proceedings of *SEM*, pages 180–191, New Orleans, USA.

Benjamin Riedel, Isabelle Augenstein, Georgios P. Spithourakis, and Sebastian Riedel. 2017. A simple but tough-to-beat baseline for the fake news challenge stance detection task. *CoRR*, abs/1707.03264.

Tim Rocktäschel, Edward Grefenstette, Karl Moritz Hermann, Tomáš Kočiský, and Phil Blunsom. 2016. Reasoning about entailment with neural attention. In *Proceedings of ICLR*, San Juan, Puerto Rico.

Cícero Nogueira dos Santos, Ming Tan, Bing Xiang, and Bowen Zhou. 2016. Attentive pooling networks. *CoRR*, abs/1602.03609.

Min Joon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hannaneh Hajishirzi. 2017. Bidirectional attention flow for machine comprehension. In *Proceedings of ICLR*, Toulon, France.

Lifeng Shang, Zhengdong Lu, and Hang Li. 2015. Neural responding machine for short-text conversation. In *Proceedings of ACL*, pages 1577–1586, Beijing, China.

Rupesh Kumar Srivastava, Klaus Greff, and Jürgen Schmidhuber. 2015. Training very deep networks. In *Proceedings of NIPS*, pages 2377–2385, Montreal, Canada.

James Thorne, Andreas Vlachos, Christos Christodoulopoulos, and Arpit Mittal. 2018. FEVER: A large-scale dataset for fact extraction and verification. In *Proceedings of NAACL-HLT*, pages 809–819, New Orleans, USA.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Proceedings of NIPS*, pages 6000–6010, Long Beach, USA.

Shuohang Wang and Jing Jiang. 2016. Learning natural language inference with LSTM. In *Proceedings of NAACL-HLT*, pages 1442–1451, San Diego, USA.

Shuohang Wang and Jing Jiang. 2017. Machine comprehension using match-LSTM and answer pointer. In *Proceedings of ICLR*, Toulon, France.

Wenhui Wang, Nan Yang, Furu Wei, Baobao Chang, and Ming Zhou. 2017a. Gated self-matching networks for reading comprehension and question answering. In *Proceedings of ACL*, pages 189–198, Vancouver, Canada.

Zhiguo Wang, Wael Hamza, and Radu Florian. 2017b. Bilateral multi-perspective matching for natural language sentences. In *Proceedings of IJCAI*, pages 4144–4150, Melbourne, Australia.

Caiming Xiong, Stephen Merity, and Richard Socher. 2016. Dynamic memory networks for visual and textual question answering. In *Proceedings of ICML*, pages 2397–2406, New York City, USA.

Caiming Xiong, Victor Zhong, and Richard Socher. 2017. Dynamic coattention networks for question answering. In *Proceedings of ICLR*, Toulon, France.

Wenpeng Yin, Hinrich Schütze, Bing Xiang, and Bowen Zhou. 2016. ABCNN: Attention-based convolutional neural network for modeling sentence pairs. *TACL*, 4:259–272.