arXiv:1805.11360v1 [cs.CL] 29 May 2018

Semantic Sentence Matching with Densely-connected Recurrent and Co-attentive Information

Seonhoon Kim[1,2], **Jin-Hyuk Hong**[1,3], **Inho Kang**[1], **Nojun Kwak**[2]

Naver Corporation[1]
Seoul National University[2]
Gwangju Institute of Science and Technology[3]

`seonhoon.kim@navercorp.com, jh7.hong@gist.ac.kr`
`once.ihkang@navercorp.com, nojunk@snu.ac.kr`

Abstract

Sentence matching is widely used in various natural language tasks such as natural language inference, paraphrase identification, and question answering. For these tasks, understanding logical and semantic relationship between two sentences is required but it is yet challenging. Although attention mechanism is useful to capture the semantic relationship and to properly align the elements of two sentences, previous methods of attention mechanism simply use a summation operation which does not retain original features enough. Inspired by DenseNet, a densely connected convolutional network, we propose a *densely-connected co-attentive recurrent neural network*, each layer of which uses concatenated information of attentive features as well as hidden features of all the preceding recurrent layers. It enables preserving the original and the co-attentive feature information from the bottommost word embedding layer to the uppermost recurrent layer. To alleviate the problem of an ever-increasing size of feature vectors due to dense concatenation operations, we also propose to use an autoencoder after dense concatenation. We evaluate our proposed architecture on highly competitive benchmark datasets related to sentence matching. Experimental results show that our architecture, which retains recurrent and attentive features, achieves state-of-the-art performances for all the tasks.

1 Introduction

Semantic sentence matching, a fundamental technology in natural language processing, requires lexical and compositional semantics. In paraphrase identification, sentence matching is utilized to identify whether two sentences have identical meaning or not. In natural language inference also known as recognizing textual entailment, it determines whether a hypothesis sentence can reasonably be inferred from a given premise sentence. In question answering, sentence matching is required to determine the degree of matching 1) between a query and a question for question retrieval, and 2) between a question and an answer for answer selection. However identifying logical and semantic relationship between two sentences is not trivial due to the problem of the semantic gap [20].

Recent advances of deep neural network enable to learn textual semantics for sentence matching. Large amount of annotated data such as Quora [8], SNLI [2], and MultiNLI [41] have contributed significantly to learning semantics as well. In the conventional methods, a matching model can be trained in two different ways [10]. The first methods are sentence-encoding-based ones where each sentence is encoded to a fixed-sized vector in a complete isolated manner and the two vectors for the

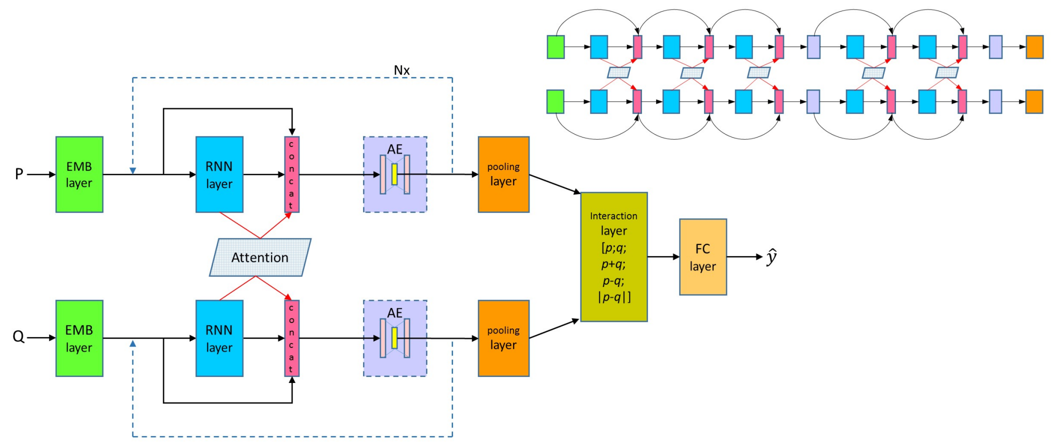


Figure 1: General architecture of our Densely-connected Recurrent and Co-attentive neural Network (DRCN). Dashed arrows indicate that a group of RNN-layer, concatenation and AE can be repeated multiple (N) times (like a repeat mark in a music score). The bottleneck component denoted as AE, inserted to prevent the ever-growing size of a feature vector, is optional for each repetition. The upper right diagram is our specific architecture for experiments with 5 RNN layers ($N = 4$).

corresponding sentences are used in predicting the degree of matching. The others are joint methods that allow to utilize interactive features like attentive information between the sentences.

In the former paradigm, because two sentences have no interaction, they can not utilize interactive information during the encoding procedure. In our work, we adopted a joint method which enables capturing interactive information for performance improvements. Furthermore, we employ a substantially deeper recurrent network for sentence matching like deep neural machine translator (NMT) [42]. Deep recurrent models are more advantageous for learning long sequences and outperform the shallower architectures. However, the attention mechanism is unstable in deeper models with the well-known vanishing gradient problem. Though GNMT [42] uses residual connection between recurrent layers to allow better information and gradient flow, there are some limitations. The recurrent hidden or attentive features are not preserved intact through residual connection because the summation operation may impede the information flow in deep networks.

Inspired by Densenet [15], we propose a densely-connected recurrent network where the recurrent hidden features are retained to the uppermost layer. In addition, instead of the conventional summation operation, the concatenation operation is used in combination with the attention mechanism to preserve co-attentive information better. The proposed architecture shown in Figure 1 is called DRCN which is an abbreviation for *Densely-connected Recurrent and Co-attentive neural Network*. The proposed DRCN can utilize the increased representational power of deeper recurrent networks and attentive information. Furthermore, to alleviate the problem of an ever-increasing feature vector size due to concatenation operations, we adopted an autoencoder and forwarded a fixed length vector to the higher layer recurrent module as shown in the figure.

We evaluate our model on three sentence matching tasks: *natural language inference*, *paraphrase identification* and *answer sentence selection*. Experimental results on five highly competitive benchmark datasets (SNLI, MultiNLI, QUORA, TrecQA and SelQA) show that our model significantly outperforms the current state-of-the-art results on all the tasks.

2 Related Work

Earlier approaches of sentence matching mainly relied on conventional methods such as syntactic features, transformations or relation extraction [30, 19, 39, 22]. These are restrictive in that they work only on very specific tasks.

The developments of large-scale annotated datasets [2, 41] and deep learning algorithms have led a big progress on matching natural language sentences. Furthermore, the well-established attention mechanisms endowed richer information for sentence matching by providing alignment and dependency relationship between two sentences. The release of the large-scale datasets also has encouraged the developments of the learning-centered approaches to semantic representation. The first type of these approaches is sentence-encoding-based methods [7, 6, 24, 34] where sentences are encoded into their own sentence representation without any cross-interaction. Then, a classifier such as a neural network is applied to decide the relationship based on these independent sentence representations. These sentence-encoding-based methods are simple to extract sentence representation and are able to be used for transfer learning to other natural language tasks [7]. On the other hand, the joint methods, which make up for the lack of interaction in the former methods, use cross-features as an attention mechanism to express the word- or phrase-level alignments for performance improvements [40, 4, 10, 43].

Recently, the architectural developments using deeper layers have led more progress in performance. The residual connection is widely and commonly used to increase the depth of a network stably [14, 42]. More recently, Huang *et al.* [15] enable the features to be connected from lower to upper layers using the concatenation operation without any loss of information on lower-layer features.

External resources are also used for sentence matching. Chen *et al.* [3, 4] used syntactic parse trees or WordNet to measure the semantic relationship among the words and Pavlick *et al.* [26] added interpretable semantics to the paraphrase database.

Unlike these, in this paper, we do not use any external resources. Our work belongs to the joint approaches which uses densely-connected recurrent and co-attentive information to enhance representation power for semantic sentence matching.

3 Methods

In this section, we describe our sentence matching architecture DRCN which is composed of the following three components: (1) word representation layer, (2) attentively connected RNN and (3) interaction and prediction layer. We denote two input sentences as $P = \{p_1, p_2, \cdots, p_I\}$ and $Q = \{q_1, q_2, \cdots, q_J\}$ where p_i/q_j is the i^{th}/j^{th} word of the sentence P/Q and I/J is the word length of P/Q. The overall architecture of the proposed DRCN is shown in Fig. 1.

3.1 Word Representation Layer

To construct the word representation layer, we concatenate word embedding, character representation and the exact matched flag which was used in [10].

In word embedding, each word is represented as a d-dimensional vector by using a pre-trained word embedding method such as GloVe [27], Word2vec [23] or Fasttext [17]. In our model, a word embedding vector can be updated or fixed during training. The strategy whether to make the pre-trained word embedding be trainable or not is heavily task-dependent. Trainable word embeddings capture the characteristics of the training data well but can result in overfitting. On the other hand, fixed (non-trainable) word embeddings lack flexibility on task-specific data, while it can be robust for overfitting, especially for less frequent words. We use both the trainable embedding $e^{tr}_{p_i}$ and the fixed (non-trainable) embedding $e^{fix}_{p_i}$ to let them play complementary roles in enhancing the performance of our model. This technique of mixing trainable and non-trainable word embeddings is simple but yet effective.

The character representation c_{p_i} is calculated by feeding randomly initialized character embeddings into a convolutional neural network with the max-pooling operation. The character embeddings and convolutional weights are jointly learned during training. Like [10], the exact match flag f_{p_i} is activated if the same word is found in the other sentence. Our final word representational feature p^w_i for the word p_i is composed of four components as follows:

$$\begin{gathered} e^{tr}_{p_i} = E^{tr}(p_i), \quad e^{fix}_{p_i} = E^{fix}(p_i) \quad c_{p_i} = \text{Char-Conv}(p_i) \\ p^w_i = [e^{tr}_{p_i}; e^{fix}_{p_i}; c_{p_i}; f_{p_i}]. \end{gathered} \tag{1}$$

Here, E^{tr} and E^{fix} are the trainable and non-trainable (fixed) word embeddings respectively. Char-Conv is the character-level convolutional operation and $[\cdot\, ; \,\cdot]$ is the concatenation operator. For each word in both sentences, the same above procedure is used to extract word features.

3.2 Densely connected Recurrent Networks

3.2.1 Recurrent Networks

We employ bidirectional LSTM (biLSTM) [11] for preserving sequential information of P and Q. BiLSTM consists of the forward and the backward LSTMs. The forward LSTM reads the input sentence from left to right and the backward LSTM reads the sentence in the reverse order (from right to left). Each LSTM calculates hidden states at each time step t as follows:

$$\begin{pmatrix} i_t \\ f_t \\ o_t \\ g_t \end{pmatrix} = \begin{pmatrix} \sigma \\ \sigma \\ \sigma \\ \tanh \end{pmatrix} \mathcal{T} \begin{pmatrix} x_t \\ h_{t-1} \end{pmatrix}, \quad c_t = f_t \circ c_{t-1} + i_t \circ g_t, \quad h_t = o_t \circ \tanh(c_t). \tag{2}$$

Here, the m-dimensional vectors i_t, f_t, o_t, c_t, g_t, and h_t are the input gate, forget gate, output gate, cell state (memory), candidate cell state and hidden state of the LSTM, respectively, at time step t. The σ, $\tanh$, and $\circ$ are sigmoid, hyper-tangent activation functions and element-wise multiplication, respectively. In the first recurrent layer, the d-dimensional input vector x_t is set to p_t^w or q_t^w for forward LSTM and p_{T-t}^w or q_{T-t}^w for backward LSTM, where T is the maximum length of the sentence. The function $\mathcal{T}$ denotes a simple affine transformation in an appropriate dimension. Here we suppress bias terms for readability.

3.2.2 Densely connected Recurrent Networks

The ordinal stacked RNNs (Recurrent Neural Networks) are composed of multiple RNN layers on top of each other, with the output sequence of previous layer forming the input sequence for the next. More concretely, let H_l be the l^{th} RNN layer in a stacked RNN. Note that in our implementation, we employ the bidirectional LSTM described in (2) as a base block of H_l. At the time step t, an ordinal stacked RNN is expressed as follows:

$$h_t^l = H_l(x_t^l, h_{t-1}^l), \quad x_t^l = h_t^{l-1}. \tag{3}$$

While this architecture enables us to build up higher level representation, deeper networks have difficulties in training due to the exploding or vanishing gradient problem [25].

To encourage gradient to flow in the backward pass, residual connection [14] is introduced which bypasses the non-linear transformations with an identity mapping. Incorporating this into (3), it becomes

$$h_t^l = H_l(x_t^l, h_{t-1}^l), \quad \underline{x_t^l = h_t^{l-1} + x_t^{l-1}}. \tag{4}$$

However, the summation operation in the residual connection may impede the information flow in the network [15]. Motivated by Densenet [15], we employ direct connections using the concatenation operation from any layer to all the subsequent layers so that the features of previous layers are not to be modified but to be retained as they are as depicted in Figure 1. The densely connected recurrent neural networks can be described as

$$\boxed{h_t^l = H_l(x_t^l, h_{t-1}^l), \quad x_t^l = [h_t^{l-1}; x_t^{l-1}].} \tag{5}$$

The concatenation operation enables the hidden features to be preserved until they reach to the uppermost layer and all the previous features work for prediction as collective knowledge [15].

3.3 Densely-connected Co-attentive networks

Attention mechanism, which has largely succeeded in many domains [42, 38], is a technique to learn effectively where a context vector is matched conditioned on a specific sequence.

Given two sentences, a context vector is calculated based on an attention mechanism focusing on the relevant part of the two sentences at each RNN layer. The calculated attentive information represents soft-alignment between two sentences. In this work, we also use an attention mechanism. We incorporate co-attentive information into densely connected recurrent features using the concatenation operation, so as not to lose any information (Fig. 1). This concatenated recurrent and co-attentive features which are obtained by densely connecting the features from the undermost to the uppermost layers, enrich the collective knowledge for lexical and compositional semantics.

P	two bicyclists in spandex and helmets in a race pedaling uphill.	Several men in front of a white building.
H	A pair of humans are riding their bicycle with tight clothing, competing with each other.	Several people in front of a gray building.
L	{**entailment**; neutral; contradiction}	{entailment; neutral; **contradiction**}

Table 1: Examples of *natural language inference*. Top to bottom: Premise, Hypothesis and Label.

The attentive information a_{p_i} of the i^{th} word $p_i \in P$ against the sentence Q is calculated as a weighted sum of h_{q_j}'s which are weighted by the softmax weights as follows :

$$a_{p_i} = \sum_{j=1}^{J} \alpha_{i,j} h_{q_j}, \quad \alpha_{i,j} = \frac{exp(e_{i,j})}{\sum_{k=1}^{J} exp(e_{i,k})}, \quad e_{i,j} = \cos(h_{p_i}, h_{q_j}). \tag{6}$$

Similar to the densely connected RNN hidden features, we concatenate the attentive context vector a_{p_i} with triggered vector h_{p_i} so as to retain attentive information as an input to the next layer:

$$h_t^l = H_l(x_t^l, h_{t-1}^l), \quad x_t^l = [h_t^{l-1}; a_t^{l-1}; x_t^{l-1}]. \tag{7}$$

3.4 Bottleneck component

Our network uses all layers' outputs as a community of semantic knowledge. However, this network is a structure with increasing input features as layers get deeper, and has a large number of parameters especially in the fully-connected layer. To address this issue, we employ an autoencoder as a bottleneck component. Autoencoder is a compression technique that reduces the number of features while retaining the original information, which can be used as a distilled semantic knowledge in our model. Furthermore, this component increased the test performance by working as a regularizer.

3.5 Interaction and Prediction Layer

To extract a proper representation for each sentence, we apply the step-wise max-pooling operation over densely connected recurrent and co-attentive features (pooling in Fig. 1). More specifically, if the output of the final RNN layer is a 100d vector for a sentence with 30 words, a 30×100 matrix is obtained which is max-pooled column-wise such that the size of the resultant vector p or q is 100. Then, we aggregate these representations p and q for the two sentences P and Q in various ways in the interaction layer and the final feature vector v for semantic sentence matching is obtained as follows:

$$v = [p; q; p + q; p - q; |p - q|]. \tag{8}$$

Here, all the operations are performed element-wise. The element-wise subtraction $p - q$ is an asymmetric operator for one-way type tasks such as *natural language inference* or *answer sentence selection*.

Finally, based on previously aggregated features v, we use two fully-connected layers with ReLU activation followed by one fully-connected output layer. Then, the softmax function is applied to obtain a probability distribution of each class. The model is trained end-to-end by minimizing the multi-class cross entropy loss and the reconstruction loss of autoencoders.

4 Experiments

We evaluate our matching model on five popular and well-studied benchmark datasets for three challenging sentence matching tasks: (i) SNLI and MultiNLI for natural language inference; (ii) Quora Question Pair for paraphrase identification; and (iii) TrecQA and SelQA for answer sentence selection in question answering. Additional details about the above datasets as well as the implementation can be found in the supplementary materials.

4.1 Experimental Results

4.1.1 SNLI and MultiNLI

We evaluated our model on the natural language inference task over SNLI and MultiNLI datasets. Table 2 shows the results on SNLI dataset of our model with other published models. Among

Sentence encoding-based method			*Joint method (cross-features available)*		
Models	**Acc.**	**|θ|**	**Models**	**Acc. (single/ensemble)**	**|θ|**
BiLSTM-Max [7]	84.5	40m	DIIN [10]	88.0 / 88.9	4.4m
Gated Attn. BiLSTM [5]	85.5	12m	ESIM [4]	88.0 / 88.6	4.3m
Gumbel TreeLSTM [6]	85.6	2.9m	BCN+CoVe+Char [21]	88.1 / -	22m
CAFE [36]	85.9	3.7m	DR-BiLSTM [9]	88.5 / 89.3	7.5m
Gumbel TreeLSTM [6]	86.0	10m	CAFE [36]	88.5 / 89.3	4.7m
Residual stacked [24]	86.0	29m	KIM [3]	88.6 / 89.1	4.3m
Reinforced SAN [34]	86.3	3.1m	ESIM+ELMo [28]	88.7 / 89.3	8.0m
Distance SAN [16]	86.3	3.1m	**DRCN** (- AE)	**88.7** / -	20m
DRCN (- Attn, - Flag)	**86.5**	5.6m	**DRCN**	**88.9 / 90.1**	6.7m

Table 2: Classification accuracy (%) for natural language inference on SNLI test set. $|\theta|$ denotes the number of parameters in each model.

Models	Accuracy (%)	
	matched	**mismatched**
ESIM [41]	72.3	72.1
DIIN [10]	78.8	77.8
CAFE [36]	78.7	77.9
DRCN	**79.1**	**78.4**
DIIN* [10]	80.0	78.7
CAFE* [36]	80.2	79.0
DRCN*	**80.6**	**79.5**

Table 3: Classification accuracy for natural language inference on MultiNLI test set. * denotes ensemble methods.

Models	Accuracy (%)
MP-LSTM [40]	83.21
L.D.C. [40]	85.55
BiMPM [40]	88.17
pt-DecAttchar.c [37]	88.40
DIIN [10]	89.06
DRCN	**90.15**
DIIN* [10]	89.84
DRCN*	**91.30**

Table 4: Classification accuracy for paraphrase identification on Quora question pair test set. * denotes ensemble methods.

them, KIM and ESIM+ELMo are the current state-of-the-art models. However, KIM uses external knowledge and ESIM+ELMo uses additional embeddings from language models. The proposed DRCN obtains an accuracy of 88.9% which is a competitive score although we do not use any external resources like KIM and ESIM+ELMo. The ensemble model achieves an accuracy of 90.1%, which sets the new state-of-the-art performance. Furthermore, in case of the encoding-based method, we obtain the best performance of 86.5% without the co-attention and exact match flag. Table 3 shows the results on MATCHED and MISMATCHED of MultiNLI dataset. DRCN also outperforms DIIN and CAFE which are the current state-of-the-art models for both cases of single and ensemble models.

4.1.2 Quora Question Pair

Table 4 shows our results on the Quora question pair dataset. BiMPM using the multi-perspective matching technique between two sentences reports baseline performance of a L.D.C. network and basic multi-perspective models [40]. We obtained accuracies of 90.15% and 91.30% in single and ensemble methods, respectively, surpassing the previous state-of-the-art model of DIIN.

4.1.3 TrecQA and SelQA

Table 5 shows the performance of different models on TrecQA and SelQA datasets for answer sentence selection task that aims to select a set of candidate answer sentences given a question. Most

Models	MAP	MRR		MAP	MRR
aNMM [43]	0.750	0.811	HyperQA [35]	0.801	0.877
PWIM [13]	0.758	0.822	PR+CNN [29]	0.801	0.877
MP CNN [12]	0.762	0.830	BiMPM [40]	0.802	0.875
HyperQA [35]	0.770	0.825	Comp.-Aggr. [1]	0.821	0.899
PR+CNN [29]	0.780	0.834	IWAN [33]	0.822	0.889
DRCN	**0.804**	**0.862**	**DRCN**	**0.830**	**0.908**

(a) TrecQA: *raw* and *clean*

Models	MAP	MRR
RNN:attn-pool [18]	0.864	0.876
CNN-DAN [31]	0.866	0.873
CNN-hinge [31]	0.876	0.881
ACNN [32]	0.874	0.880
AdaQA [32]	0.891	0.898
DRCN	**0.925**	**0.930**

(b) SelQA

Table 5: Performance for answer sentence selection on TrecQA and selQA test set.

Models	Accuracy (%)
(1) DRCN	89.4
(2) − autoencoder	89.1
(3) − E^{tr}	88.7
(4) − E^{fix}	88.9
(5) − dense(Attn.)	88.7
(6) − dense(Rec.)	88.8
(7) − dense(Rec. & Attn.)	88.5
(8) − dense(Rec. & Attn.) + res(Rec. & Attn.)	88.7
(9) − dense(Rec. & Attn. & Emb) + res(Rec. & Attn.)	88.4
(10) − dense(Rec. & Attn. & Emb)	87.8
(11) − dense(Rec. & Attn. & Emb) - Attn.	85.3

Table 6: Ablation study results on the SNLI dev sets.

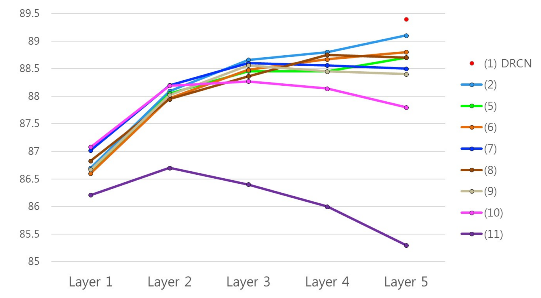


Figure 2: Comparison of models on every layer in ablation study. (best viewed in color)

competitive models [33, 1, 40, 32] also use attention methods for words alignment between question and candidate answer sentences. However, the proposed DRCN using collective attentions over multiple layers, achieves the new state-of-the-art performance, exceeding the current state-of-the-art performance significantly on both datasets.

4.2 Analysis

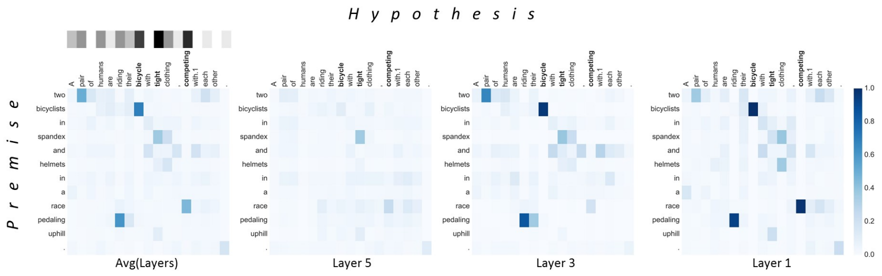
4.2.1 Ablation study

We conducted an ablation study on the SNLI dev set as shown in Table 6, where we aim to examine the effectiveness of our word embedding technique as well as the proposed densely-connected recurrent and co-attentive features. Firstly, we verified the effectiveness of the autoencoder as a bottleneck component in (2). Although the number of parameters in the DRCN significantly decreased as shown in Table 2, we could see that the performance was rather higher because of the regularization effect. Secondly, we study how the technique of mixing trainable and fixed word embeddings contributes to the performance in models (3-4). After removing E^{tr} or E^{fix} in eq. (1), the performance degraded, slightly. The trainable embedding E^{tr} seems more effective than the fixed embedding E^{fix}. Next, the effectiveness of dense connections was tested in models (5-9). In (5-6), we removed dense connections only over co-attentive or recurrent features, respectively. The result shows that the dense connections over attentive features are more effective. In (7), we removed dense connections over both co-attentive and recurrent features, and the performance degraded to 88.5%. In (8), we replace dense connection with residual connection only over recurrent and co-attentive features. It means that only the word embedding features are densely connected to the uppermost layer while recurrent and attentive features are connected to the upper layer using the residual connection. In (9), we removed additional dense connection over word embedding features from (8). The results of (8-9) demonstrate that the dense connection using concatenation operation over deeper layers, has more powerful capability retaining collective knowledge to learn textual semantics. The model (10) is the basic 5-layer RNN with attention and (11) is the one without attention. The result of (10) shows that the connections among the layers are important to help gradient flow. And, the result of (11) shows that the attentive information functioning as a soft-alignment is significantly effective in semantic sentence matching.

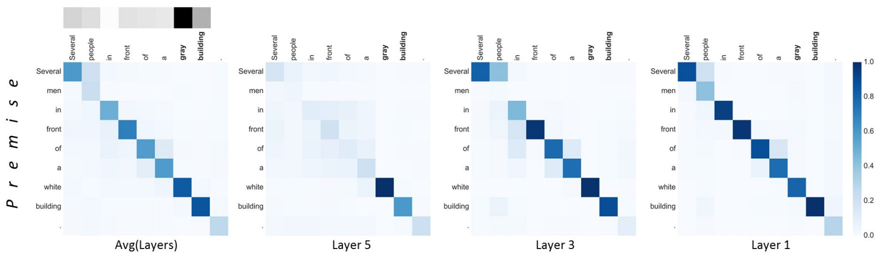
The performances of models having different number of recurrent layers are also reported in Fig. 2. The models (5-9) which have connections between layers, are more robust to the increased depth of network, however, the performances of (10-11) tend to degrade as layers get deeper. In addition, the models with dense connections rather than residual connections, have higher performance in general. Figure 2 shows that the connection between layers is essential, especially in deep models, endowing more representational power, and the dense connection is more effective than the residual connection.

4.2.2 Word Alignment and Importance

Our densely-connected recurrent and co-attentive features are connected to the classification layer through the max pooling operation such that all max-valued features of every layer affect the loss function and perform a kind of deep supervision [15]. Thus, we could cautiously interpret the classification results using our attentive weights and max-pooled positions. The attentive weights



(a) entailment



(b) contradiction

Figure 3: Visualization of attentive weights and the rate of max-pooled position. The darker, the higher. See supplementary materials for a comparison with other models regarding attention maps.

contain information on how two sentences are aligned and the numbers of max-pooled positions in each dimension play an important role in classification.

Figure 3 shows the attention map ($\alpha_{i,j}$ in eq. (6)) on each layer of the samples in Table 1. The Avg(Layers) is the average of attentive weights over 5 layers and the gray heatmap right above the Avg(Layers) is the rate of max-pooled positions. The darker indicates the higher importance in classification. In the figure, we can see that *tight*, *competing* and *bicycle* are more important words than others in classifying the label. The word *tight clothing* in the hypothesis can be inferred from *spandex* in the premise. And *competing* is also inferred from *race*. Other than that, the *riding* is matched with *pedaling*, and *pair* is matched with *two*. Judging by the matched terms, the model is undoubtedly able to classify the label as an entailment, correctly.

In Figure 3 (b), most of words in both the premise and the hypothesis coexist except *white* and *gray*. In attention map of layer 1, the same or similar words in each sentence have a high correspondence (*gray* and *white* are not exactly matched but have a linguistic relevance). However, as the layers get deeper, the relevance between *white building* and *gray building* is only maintained as a clue of classification (See layer 5). Because *white* is clearly different from *gray*, our model determines the label as a contradiction.

The densely connected recurrent and co-attentive features are well-semanticized over multiple layers as collective knowledge. And the max pooling operation selects the soft-positions that may extract the clues on inference correctly.

5 Conclusion

In this paper, we introduce a densely-connected recurrent and co-attentive network (DRCN) for semantic sentence matching. We connect the recurrent and co-attentive features from the bottom to the top layer without any deformation. These intact features over multiple layers compose a community of semantic knowledge and outperform the previous deep RNN models using residual connections. In doing so, bottleneck components are inserted to reduce the size of the network. We additionally show the interpretability of our model using the attentive weights and the rate of max-pooled positions. Our model achieves the state-of-the-art performance on five benchmark datasets of three highly challenging natural language tasks.

References

[1] Weijie Bian, Si Li, Zhao Yang, Guang Chen, and Zhiqing Lin. A compare-aggregate model with dynamic-clip attention for answer selection. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, pages 1987–1990. ACM, 2017.

[2] Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. 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, 2015.

[3] Qian Chen, Xiaodan Zhu, Zhen-Hua Ling, and Diana Inkpen. Natural language inference with external knowledge. *arXiv preprint arXiv:1711.04289*, 2017.

[4] Qian Chen, Xiaodan Zhu, Zhen-Hua Ling, Si Wei, Hui Jiang, and Diana Inkpen. Enhanced lstm for natural language inference. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 1657–1668, 2017.

[5] Qian Chen, Xiaodan Zhu, Zhen-Hua Ling, Si Wei, Hui Jiang, and Diana Inkpen. Recurrent neural network-based sentence encoder with gated attention for natural language inference. *arXiv preprint arXiv:1708.01353*, 2017.

[6] Jihun Choi, Kang Min Yoo, and Sang goo Lee. Learning to compose task-specific tree structures. AAAI, 2017.

[7] Alexis Conneau, Douwe Kiela, Holger Schwenk, Loic Barrault, and Antoine Bordes. Supervised learning of universal sentence representations from natural language inference data. *arXiv preprint arXiv:1705.02364*, 2017.

[8] Kornél Csernai. Quora question pair dataset, 2017.

[9] Reza Ghaeini, Sadid A Hasan, Vivek Datla, Joey Liu, Kathy Lee, Ashequl Qadir, Yuan Ling, Aaditya Prakash, Xiaoli Z Fern, and Oladimeji Farri. Dr-bilstm: Dependent reading bidirectional lstm for natural language inference. *arXiv preprint arXiv:1802.05577*, 2018.

[10] Yichen Gong, Heng Luo, and Jian Zhang. Natural language inference over interaction space. In *International Conference on Learning Representations*, 2018.

[11] Alex Graves. Bidirectional lstm networks for improved phoneme classification and recognition. In *In Proceedings of the 2005 International Conference on Artificial Neural Networks*, 2006.

[12] Hua He, Kevin Gimpel, and Jimmy Lin. Multi-perspective sentence similarity modeling with convolutional neural networks. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1576–1586, 2015.

[13] Hua He and Jimmy Lin. Pairwise word interaction modeling with deep neural networks for semantic similarity measurement. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 937–948, 2016.

[14] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[15] Gao Huang, Zhuang Liu, Kilian Q Weinberger, and Laurens van der Maaten. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, volume 1, page 3, 2017.

[16] Jinbae Im and Sungzoon Cho. Distance-based self-attention network for natural language inference. *arXiv preprint arXiv:1712.02047*, 2017.

[17] Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. Bag of tricks for efficient text classification. *arXiv preprint arXiv:1607.01759*, 2016.

[18] Tomasz Jurczyk, Michael Zhai, and Jinho D Choi. Selqa: A new benchmark for selection-based question answering. In *Tools with Artificial Intelligence (ICTAI), 2016 IEEE 28th International Conference on*, pages 820–827. IEEE, 2016.

[19] Zornitsa Kozareva and Andrés Montoyo. Paraphrase identification on the basis of supervised machine learning techniques. In *Advances in natural language processing*, pages 524–533. Springer, 2006.

[20] Pengfei Liu, Xipeng Qiu, Jifan Chen, and Xuanjing Huang. Deep fusion lstms for text semantic matching. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 1034–1043, 2016.

[21] Bryan McCann, James Bradbury, Caiming Xiong, and Richard Socher. Learned in translation: Contextualized word vectors. In *Advances in Neural Information Processing Systems*, pages 6297–6308, 2017.

[22] Yashar Mehdad, Alessandro Moschitti, and Fabio Massimo Zanzotto. Syntactic/semantic structures for textual entailment recognition. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 1020–1028. Association for Computational Linguistics, 2010.

[23] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119, 2013.

[24] Yixin Nie and Mohit Bansal. Shortcut-stacked sentence encoders for multi-domain inference. *arXiv preprint arXiv:1708.02312*, 2017.

[25] Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. On the difficulty of training recurrent neural networks. In *International Conference on Machine Learning*, pages 1310–1318, 2013.

[26] Ellie Pavlick, Johan Bos, Malvina Nissim, Charley Beller, Benjamin Van Durme, and Chris Callison-Burch. Adding semantics to data-driven paraphrasing. 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)*, volume 1, pages 1512–1522, 2015.

[27] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, 2014.

[28] Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. Deep contextualized word representations. In *Proc. of NAACL*, 2018.

[29] Jinfeng Rao, Hua He, and Jimmy Lin. Noise-contrastive estimation for answer selection with deep neural networks. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*, pages 1913–1916. ACM, 2016.

[30] Lorenza Romano, Milen Kouylekov, Idan Szpektor, Ido Dagan, and Alberto Lavelli. Investigating a generic paraphrase-based approach for relation extraction. In *11th Conference of the European Chapter of the Association for Computational Linguistics*, 2006.

[31] Cicero Nogueira dos Santos, Kahini Wadhawan, and Bowen Zhou. Learning loss functions for semi-supervised learning via discriminative adversarial networks. *arXiv preprint arXiv:1707.02198*, 2017.

[32] Dinghan Shen, Martin Renqiang Min, Yitong Li, and Lawrence Carin. Adaptive convolutional filter generation for natural language understanding. *arXiv preprint arXiv:1709.08294*, 2017.

[33] Gehui Shen, Yunlun Yang, and Zhi-Hong Deng. Inter-weighted alignment network for sentence pair modeling. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1179–1189, 2017.

[34] Tao Shen, Tianyi Zhou, Guodong Long, Jing Jiang, Sen Wang, and Chengqi Zhang. Reinforced self-attention network: a hybrid of hard and soft attention for sequence modeling. *arXiv preprint arXiv:1801.10296*, 2018.

[35] Yi Tay, Anh Tuan Luu, and Siu Cheung Hui. Enabling efficient question answer retrieval via hyperbolic neural networks. *CoRR abs/1707.07847*, 2017.

[36] Yi Tay, Luu Anh Tuan, and Siu Cheung Hui. A compare-propagate architecture with alignment factorization for natural language inference. *arXiv preprint arXiv:1801.00102*, 2017.

[37] Gaurav Singh Tomar, Thyago Duque, Oscar Täckström, Jakob Uszkoreit, and Dipanjan Das. Neural paraphrase identification of questions with noisy pretraining. *arXiv preprint arXiv:1704.04565*, 2017.

[38] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 6000–6010, 2017.

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

[40] Zhiguo Wang, Wael Hamza, and Radu Florian. Bilateral multi-perspective matching for natural language sentences. *arXiv preprint arXiv:1702.03814*, 2017.

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

[42] Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, et al. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*, 2016.

[43] Liu Yang, Qingyao Ai, Jiafeng Guo, and W Bruce Croft. anmm: Ranking short answer texts with attention-based neural matching model. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*, pages 287–296. ACM, 2016.

6 Supplementary Material

6.1 Datasets

A. SNLI is a collection of 570k human written sentence pairs based on image captioning, supporting the task of natural language inference [2]. The labels are composed of entailment, neutral and contradiction. The data splits are provided in [2].

B. MultiNLI, also known as Multi-Genre NLI, has 433k sentence pairs whose size and mode of collection are modeled closely like SNLI. MultiNLI offers ten distinct genres (FACE-TO-FACE, TELEPHONE, 9/11, TRAVEL, LETTERS, OUP, SLATE, VERBATIM, GOVERNMENT and FICTION) of written and spoken English data. Also, there are matched dev/test sets which are derived from the same sources as those in the training set, and mismatched sets which do not closely resemble any seen at training time. The data splits are provided in [41].

C. Quora Question Pair consists of over 400k question pairs based on actual `quora.com` questions. Each pair contains a binary value indicating whether the two questions are paraphrase or not. The training-dev-test splits for this dataset are provided in [40].

D. TrecQA provided in [39] was collected from TREC Question Answering tracks 8-13. There are two versions of data due to different pre-processing methods, namely clean and raw [29]. We evaluate our model on both data and follow the same data split as provided in [39]. We use official evaluation metrics of MAP (Mean Average Precision) and MRR (Mean Reciprocal Rank), which are standard metrics in information retrieval and question answering tasks.

E. SelQA consists of questions generated through crowdsourcing and the answer senteces are extracted from the ten most prevalent topics (Arts, Country, Food, Historical Events, Movies, Music, Science, Sports, Travel and TV) in the English Wikipedia. We also use MAP and MRR for our evaluation metrics, and the data splits are provided in [18].

6.2 Implementation Details

We initialized word embedding with 300d GloVe vectors pre-trained from the 840B Common Crawl corpus [27], while the word embeddings for the out-of-vocabulary words were initialized randomly. We also randomly initialized character embedding with a 16d vector and extracted 32d character representation with a convolutional network. For the densely-connected recurrent layers, we stacked 5 layers each of which have 100 hidden units. We set 1000 hidden units with respect to the fully-connected layers. The dropout was applied after the word and character embedding layers with a keep rate of 0.5. It was also applied before the fully-connected layers with a keep rate of 0.8. For the bottleneck component, we set 200 hidden units as encoded features of the autoencoder with a dropout rate of 0.2. The batch normalization was applied on the fully-connected layers, only for the one-way type datasets. The RMSProp optimizer with an initial learning rate of 0.001 was applied. The learning rate was decreased by a factor of 0.85 when the dev accuracy does not improve. All weights except embedding matrices are constrained by L2 regularization with a regularization constant $\lambda = 10^{-6}$. The sequence lengths of the sentence are all different for each dataset: 35 for SNLI, 55 for MultiNLI, 25 for Quora question pair and 50 for TrecQA. The learning parameters were selected based on the best performance on the dev set. We employed 8 different randomly initialized sets of parameters with the same model for our ensemble approach.

6.3 Visualization on the comparable models

We study how the attentive weights flow as layers get deeper in each model using the dense or residual connection. We used the samples of the SNLI dev set in Table 1.

Figure 4 and 5 show the attention map on each layer of the models of DRCN, Table 6 (8), and Table 6 (9). In the model of Table 6 (8), we replaced the dense connection with the residual connection only over recurrent and co-attentive features. And, in the model of Table 6 (9), we removed additional dense connection over word embedding features from Table 6 (8). We denote the model of Table 6 (9) as Res1 and the model of Table 6 (8) as Res2 for convenience.

In Figure 4, DRCN does not try to find the right alignments at the upper layer if it already finds the rationale for the prediction at the relatively lower layer. This is expected that the DRCN use the features of all the preceding layers as a collective knowledge. While Res1 and Res2 have to find correct alignments at the top layer, however, there are some misalignments such as *competing* and *bicyclists* rather than *competing* and *race* in Res2 model.

In the second example in Figure 5, although the DRCN couldn't find the clues at the lower layer, it gradually finds the alignments, which can be a rationale for the prediction. At the 5th layer of DRCN, the attentive weights of *gray building* and *white building* are significantly higher than others. On the other hand, the attentive weights are spread in several positions in both Res1 and Res2 which use residual connection.

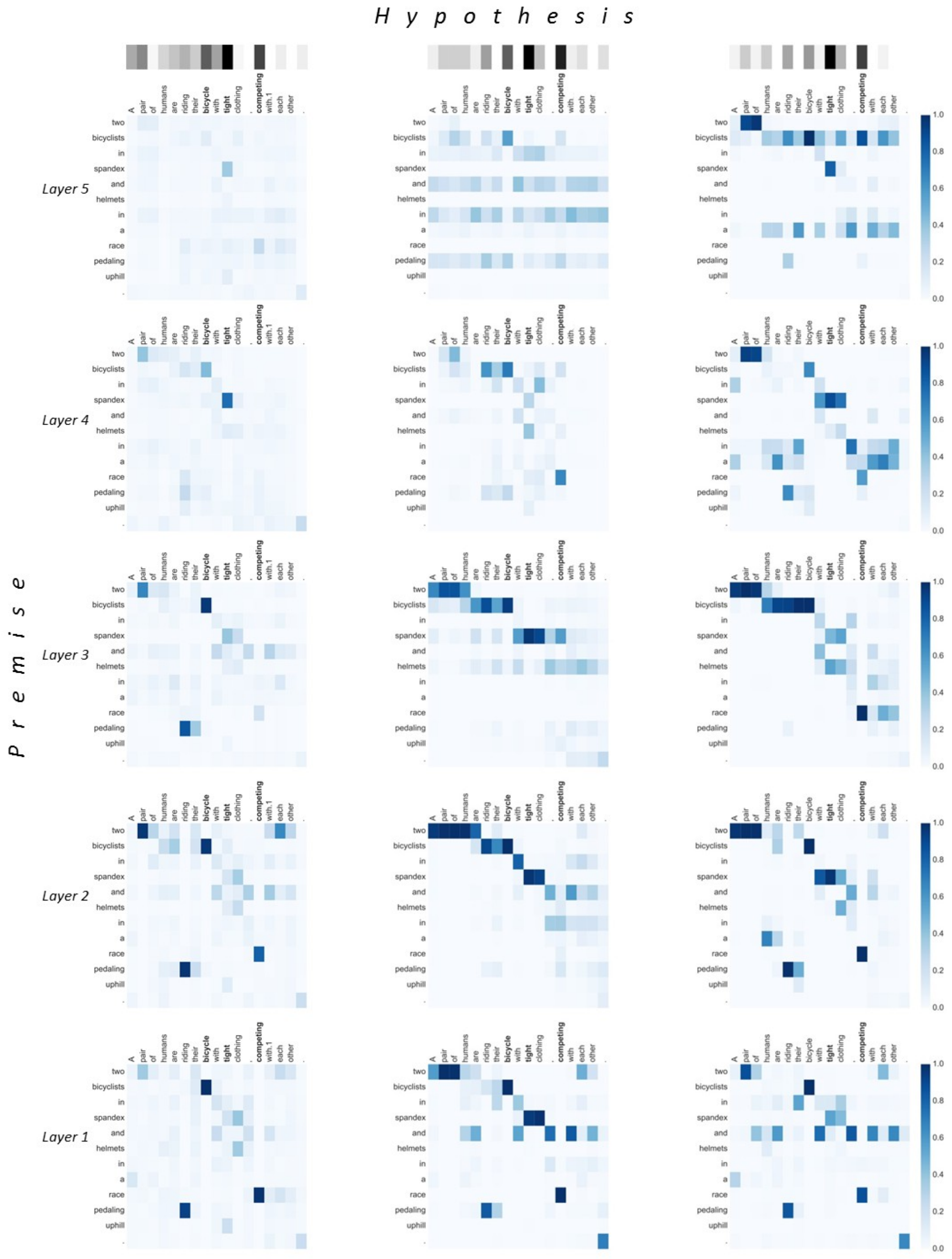


Figure 4: Visualization of attentive weights on the *entailment* example. The premise is "*two bicyclists in spandex and helmets in a race pedaling uphill.*" and the hypothesis is "*A pair of humans are riding their bicycle with tight clothing, competing with each other.*". The attentive weights of DRCN, Res1, and Res2 are presented from left to right.

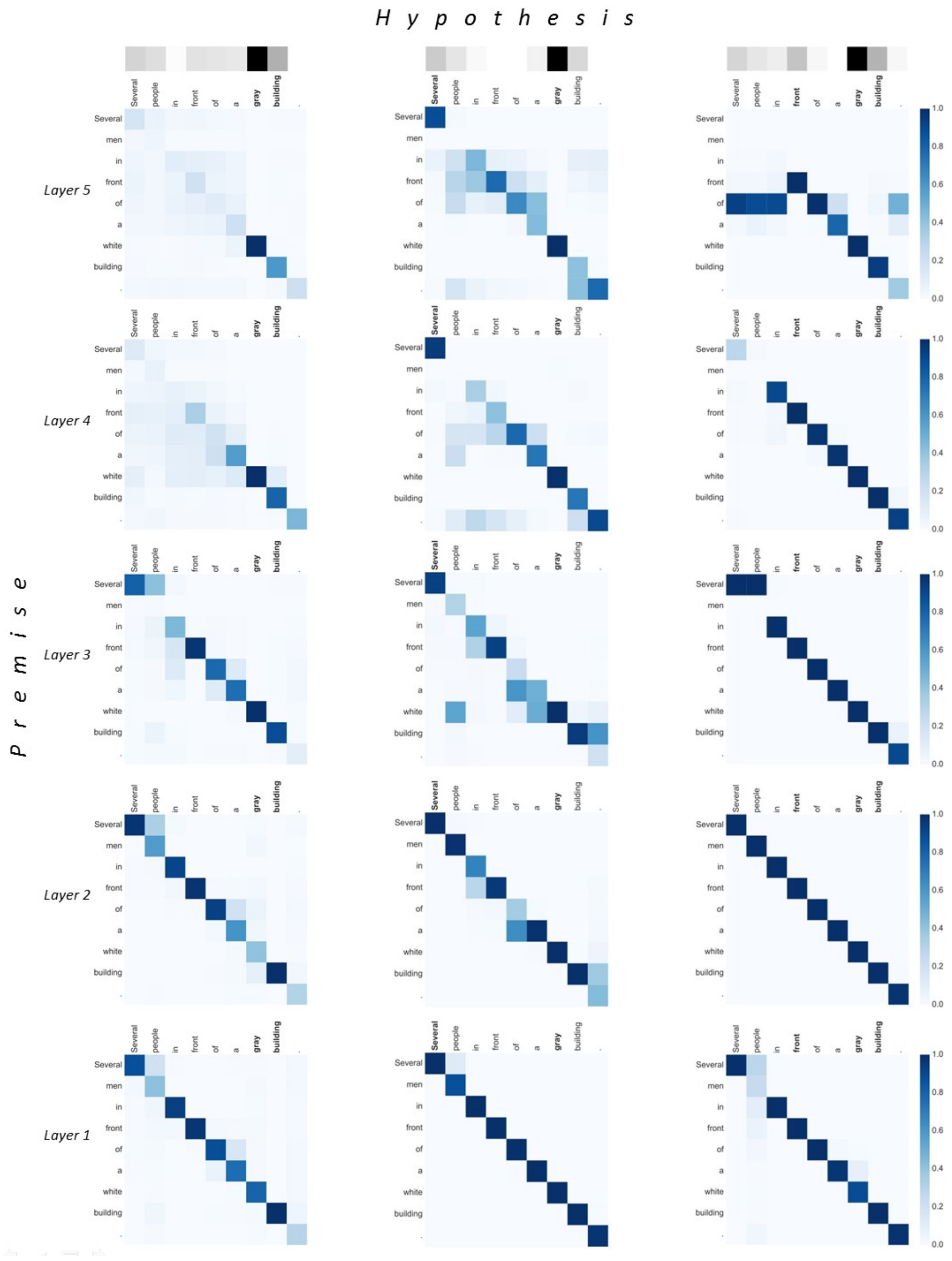


Figure 5: Visualization of attentive weights on the *contradiction* example. The premise is "*Several men in front of a white building.*" and the hypothesis is "*Several people in front of a gray building.*". The attentive weights of DRCN, Res1, and Res2 are presented from left to right.