EXPLORING ASYMMETRIC ENCODER-DECODER STRUCTURE FOR CONTEXT-BASED SENTENCE REPRESENTATION LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Context information plays an important role in human language understanding, and it is also useful for machines to learn vector representations of language. In this paper, we and explore an asymmetric encoder-decoder structure for unsupervised context-based sentence representation learning. As a result, we build an encoder-decoder architecture with an RNN encoder and a CNN decoder. We further combine a suite of effective designs to significantly improve model efficiency while also achieve better performance. Our model is trained on two different large unlabeled corpora, and in both cases transferability is evaluated on a set of downstream language understanding tasks. We empirically show that our model is simple and fast while producing rich sentence representations that excel in downstream tasks.

1 INTRODUCTION

Learning distributed representations of sentences is an important and hard topic in both the deep learning and natural language processing communities, since it requires machines to encode a sentence with rich language content into a fixed-dimension vector filled with continuous values. We are interested in learning to build a distributed sentence encoder in an unsupervised fashion by exploiting the structure and relationship in a large unlabeled corpus. Since humans interpret sentences by composing from the meanings of the words, we decompose the task of learning a sentence encoder into two essential components, which are learning distributed word representations, and learning how to compose a sentence representation from the representations of words in the given sentence.

Numerous studies in human language processing have claimed that the context in which words and sentences are understood plays an important role in human language understanding (Altmann & Mirkovic, 2009; Binder & Desai, 2011). The idea of learning from the context information was first successfully applied to vector representation learning for words in Mikolov et al. (2013b) and learning from the occurrence of words also succeeded in Pennington et al. (2014).

Inspired by the prior work on incorporating context information into representation learning, Kiros et al. (2015) proposed the Skip-thought model, which is an asymmetric encoder-decoder model for unsupervised sentence representation learning. The paper exploits the semantic similarity within a tuple of adjacent sentences as supervision, and successfully built a generic, distributed sentence encoder. Rather than applying the conventional autoencoder model, the skip-thought model tries to reconstruct the surrounding 2 sentences instead of the input sentence. The learned sentence representation encoder outperforms previous unsupervised pretrained models on the evaluation tasks with no finetuning, and the results are comparable to the models which were trained directly on the datasets in a supervised fashion.

The usage of 2 independent decoders in Skip-thought model matches our intuition that, given the current sentence, inferring the previous sentence and inferring the next one should be different. Recently, Tang et al. (2017) proposed the Neighborhood Hypothesis, which showed that given the current sentence, inferring the previous one and inferring the next one both provide the same supervision power for learning sentence representation. They also experimentally showed that the model with one encoder and one decoder for the reconstruction of the next sentence performs as well as their implementation of the Skip-thought model does on 7 downstream tasks, thus we adopted the idea of using only one encoder and one decoder, which largely decreases the training time. The

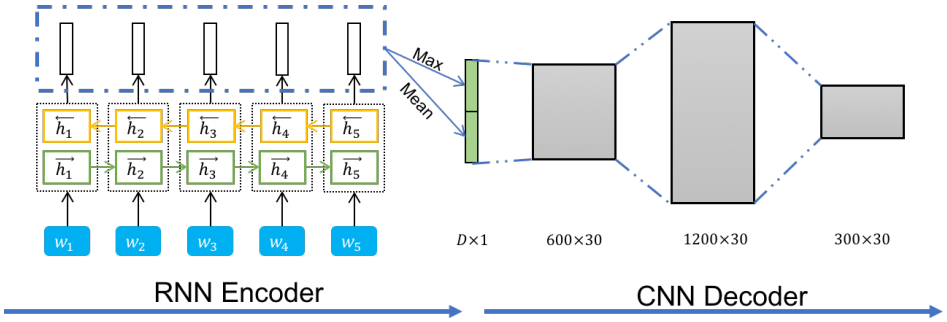

Figure 1: Our proposed model is composed of an RNN encoder, and a CNN decoder. During training, a batch of sentences are sent to the model, and the RNN encoder computes a vector representation for each of sentences; then the CNN decoder needs to reconstruct the paired target sequence, which contains 30 contiguous words right after the input sentence, given the vector representation. 300 is the dimension of word vectors. D is the dimension of sentence representation, and it varies along with the change of the RNN encoder size. (Better view in color.)

resulting model of the Neighborhood Hypothesis is also an asymmetric encoder-decoder model, while the encoder and decoder are both RNNs, the input and the target for reconstruction are different.

In this paper, we follow the design principle in the Neighborhood Hypothesis, which exploits the right context information for learning representation, and aim to bring asymmetry into structure design as well. Our proposed model has an asymmetric encoder-decoder structure, which keeps an RNN as the encoder and has a CNN as the decoder, and it is called as "RNN-CNN" model. The key components of our model design can be summarized as:

1. a bidirectional RNN encodes the input sentence, and a CNN decodes the paired target sequence, which speeds up the training process;
2. the supervision for training comes from inferring the next contiguous words given the current sentence, which helps the model to learn from the context in an unsupervised fashion;
3. the mean+max pooling captures complex interactions among words, which augments the transferability of the proposed model;
4. tying word embeddings with the word prediction layer constrains the input and output space to be the same, which d reduces the large number of parameters and regularizes the search space.

We demonstrate the transferability of our model by evaluation on various downstream tasks, and the performance shows that our model improves both results and efficiency in training.

2 RNN-CNN Model

Our model is highly asymmetric in terms of both training pairs and model structure. Specifically, our model has an RNN as the encoder, and a CNN as the decoder. During training, the encoder takes a sentence (s_i) as input, and then generates a fixed-dimension vector ($\mathbf{z}_i$) as the sentence representation; the decoder is applied to reconstruct the next sentence or the next few contiguous words (t_i). The difference of the generated sequence and the target sequence is measured by cross-entropy loss. An illustration is in Figure 1.

Encoder: The encoder is a bi-directional Gated Recurrent Unit (GRU) (Chung et al., 2014). We experimented with both Long-short Term Memory (LSTM, Hochreiter & Schmidhuber (1997)) and GRU. Since LSTM didn't give us significant performance boost, and generally GRU runs faster than LSTM, in our experiments, we stick to using GRU in the encoder. Suppose that a sentence s_i contains M words, which are $w_i^1, w_i^2, ..., w_i^M$. The bi-directional GRU will take the input sentence one word at a time, and run in both forward and backward direction; both sets of hidden states are concatenated to form the hidden state matrix $[\mathbf{h}_i^1; \mathbf{h}_i^2; ...; \mathbf{h}_i^M]$.

Representation: We aim to provide a model with faster training speed with better transferability than existing algorithms, thus we choose to apply a non-parametric composition function, which is a concatenation of the outputs from a mean pooling and a max pooling, on the computed sequence of hidden states. The composition function can be represented as $\mathbf{z}_i = \left[\frac{1}{M}\sum_{m=1}^{M}\mathbf{h}_i^m; \max_{m=1}^{M}\mathbf{h}_i^m\right]$.

Decoder: The decoder is a 3-layer CNN to reconstruct the paired target sequence t_i, which needs to expand $\mathbf{z}_i$ from length 1 to the length of t_i. Intuitively, the decoder could be a stack of deconvolution layers. For fast training speed, we optimized the architecture to make it plausible to use fully-connected layers and convolution layers in the decoder, since generally, convolution layers run faster than deconvolution layers in modern deep learning frameworks.

Suppose that the target sequence t_i has L words, the first layer of deconvolution will expand $\mathbf{z}_i$, which could be considered as a sequence with length 1, into a feature map with length L. It can be easily implemented as a concatenation of outputs from L fully-connected layers. Then the second and third layer are 1D-convolution layers with kernel size 3 and 1, respectively.

3 MOTIVATION

We walk through the motivation for the key components for improvement, and justify our decisions.

3.1 RNN AS ENCODER

We choose to use a bi-directional RNN as the encoder, since we believe that the learned representation will benefit from the explicit usage of word order information in RNN.

CNN implicitly learns to encode the word order information into vector representation of sentences, since it can be considered as a hierarchical n-gram model. Gan et al. (2017) first applied a CNN as the encoder in unsupervised sentence representation learning. The *future predictor* in their paper adopted the same idea as the Neighborhood Hypothesis, but the performance was relatively poor compared to other existing models. Instead of using an RNN as the decoder, Zhang et al. (2017) used CNNs for both encoding and decoding, however the unsupervised aspect of their CNN encoder decays over training time (and is overpowered by the supervised component), so it is difficult to evaluate the contribution of the CNN-CNN network for unsupervised sentence representation learning.

We agree that CNN with proper architecture design can be a great and efficient encoding model for language, as discussed in Zhang & LeCun (2017), but we are slightly conservative in our views about the transferability of CNN encoder for unsupervised language representation learning. Based on our experience and previous work, we stick to choosing a bi-directional RNN for encoding sentences.

3.2 CNN AS DECODER

Our model as well as Skip-thought (Kiros et al., 2015), CNN-LSTM (Gan et al., 2017), and FastSent (Hill et al., 2016) are all based on the sentence-level Distributional Hypothesis (Harris, 1954; Polajnar et al., 2015) and recent study on human language processing (Binder & Desai, 2011; Altmann & Mirkovic, 2009): the context in which a word or a sentence occurs and is understood contributes significantly to its semantics, and to human comprehension as well. However, how best to model contextual information is still an ongoing research topic, and how humans infer the meaning of a sentence based on the context (a conversation, a paragraph, etc.) is still to be unveiled.

According to the evidence we stated above, we listed 3 reasons for applying CNN as the decoder in our model.

Higher Efficiency: Although the decoder won't be used during test time, the model still benefits from the speed-up brought by a CNN decoder, since the computations in CNN run in parallel. While RNN requires to decode one word at a time, CNN is able to decode all words at a single run.

Less Constraints: During encoding, the explicit word order information used in RNN will help the vector representation capture more of the temporally-specific relationships among words, but this same constraint (if using RNN as the decoder) could be an inappropriate constraint in the decoding process.

At every time step in training the RNN decoder, the probability of the next word is computed based on the left-context embedding, ground-truth current word embedding, and sentence representation. The existence of the ground-truth current word embedding potentially decreases the tendency for the decoder to exploit other information from the sentence representation. In this way the information from the sentence representation, which we desire to extract, is underexploited.

While an RNN decoder, is designed to produce the next word, a CNN decoder is free to find any relevant local patterns within the target sequences. The generation process is only conditioned on the sentence representation. Although the word order information is implicitly encoded in the CNN decoder, it is not emphasized as it is explicitly in the RNN decoder. The CNN decoder cares about the quality of generated sequences globally instead of the quality of the next generated word. Relaxing the emphasis on the next word, may help the CNN decoder model to explore the contribution of context in a larger space.

Application: A CNN decoder matches the application of the learned representation after training. In evaluation, the generated representations will be used as a fixed-length input to a linear classifier, or a multi-layer perceptron customized for a downstream task. The CNN architecture matches the final usage of the representation, thus we choose to apply a CNN as the decoder.

3.3 MEAN+MAX POOLING

Since the encoder is a bi-directional RNN in our model, we have multiple ways to select/compute on the generated hidden states to produce a sentence representation. In Skip-thought (Kiros et al., 2015) and SDAE (Hill et al., 2016), only the hidden state at the last time step computed by the RNN encoder is regarded as the vector representation for a given sentence, which may not be the most expressive vector for representing the input sentence.

We adopted the idea proposed in Chen et al. (2016). They aim to build a model for supervised SNLI task (Bowman et al., 2015), and the model concatenates the outputs from a global mean-pooling function and a global max-pooling function to serve as a sentence representation, and shows a performance boost on the SNLI dataset. Besides, Conneau et al. (2017) found that the model with global max-pooling function has stronger transferability than the model with a global mean-pooling function after supervised training on SNLI.

The concatenation of a mean-pooling and a max-pooling function is actually a non-parametric composition function, and the computation load is negligible compared to heavy matrix multiplications. Also, the non-linearity of the max-pooling function augments the mean pooling function for building a representation that captures more complex composition of the syntactic information.

3.4 TYING WORD EMBEDDINGS AND WORD PREDICTION LAYER

We choose to share the parameters in the word embedding layer in RNN encoder and the word prediction layer in CNN decoder. The tying was proposed in both Press & Wolf (2017) and Inan et al. (2016), and it generally helps to learn a better language model.

In our case, tying the weights in the word prediction layer with the word embeddings constrains the input space and output space to be the same, and the model learns to explore the non-linear compositionality of the input words and the uncertain contribution of the target words in the same space. The tying also drastically reduces the number of parameters in the model, which could prevent overfitting.

Furthermore, we initialize the word embeddings with pretrained word vectors, such as word2vec (Mikolov et al., 2013b) and GloVe (Pennington et al., 2014), since it has been shown that these pretrained word vectors can serve as good initialization for deep learning models, and more likely lead to better results than random samples from a uniform distribution.

4 EXPERIMENT SETTINGS

The large corpus that we used for unsupervised training is the BookCorpus dataset Zhu et al. (2015), which contains 74 million sentences from 7000 books in total. For stable training, we use ADAM (Kingma & Ba, 2014) algorithm for optimization, and gradient clipping (Pascanu et al., 2013) when

the norm of gradient exceeds a certain value. Since we didn't find any significant difference between word2vec and GloVe as initialization in terms of the performance, we stick to using word vectors from word2vec to initialize the word embedding layer in our models.

The vocabulary for unsupervised training is set to contain the top 20k most frequent words in BookCorpus. In order to generalize the model trained with a relatively small, fixed vocabulary to the much larger set of all possible English words, Kiros et al. (2015) proposed a word expansion method that learns a linear projection from the pretrained word embeddings word2vec to the learned RNN word embeddings. Thus, the model benefits from the generalization ability of the pretrained word embeddings.

The downstream tasks for evaluation include semantic relatedness (SICK) (Marelli et al., 2014), paraphrase detection (MSRP) (Dolan et al., 2004), question-type classification (TREC) (Li & Roth, 2002), and 5 benchmark sentiment and subjective datasets, which includes movie review sentiment (MR, SST) (Pang & Lee, 2005; Socher et al., 2013), customer product reviews (CR) (Hu & Liu, 2004), subjectivity/objectivity classification (SUBJ) (Pang & Lee, 2004), opinion polarity (MPQA) (Wiebe et al., 2005), and microsoft COCO image-caption retrieval (COCO) (Lin et al., 2014). After unsupervised training on the BookCorpus dataset, we fix the parameters in the encoder, and apply it as a sentence representation extractor on the 10 tasks.

In order to compare the effect of different corpora, we also trained 2 models on Amazon Book Review dataset without scores. The Amazon Book Review is the largest subset of Amazon Review (McAuley et al., 2015), and it has 142 million sentences after tokenization, which is around twice as large as BookCorpus.

Both training and evaluation of our models were conducted in PyTorch [1], and we used SentEval [2] provided by Conneau et al. (2017) to evaluate the transferability of models with different settings. All the models were trained for the same number of iterations with the same batch size, and the performance was measured at the end of training for each of the models.

5 Architectural Design

In this section, we will discuss how we made key architectural decisions, and their influence on the transferability of the associated models. The comparison is based on supervised evaluation tasks, including SICK, MSRP, SST and TREC, and an unsupervised evaluation task, which is STS14. The training time of same iterations with same batch size is also recorded during training to serve as an evidence for model comparison. Table 1 presents results. Since our model was directly inspired by Skip-thought model, we also list the results of Skip-thought model (Kiros et al., 2015), and Skip-thought with Layer Normalization (Ba et al., 2016).

5.1 Decoding Sentences vs. Decoding Sequences

Given that the encoder takes a sentence as input, decoding the next sentence versus decoding the next fixed length window of contiguous words is conceptually different. This is because decoding the next fixed-length sequence might not reach or might go beyond the boundary of the next sentence. Since the CNN decoder in our model takes a fixed-length sequence as the target, when it comes to decoding sentences, we would need to zero-pad or chop the sentences into a fixed length. As the transferability of the models trained in both cases perform similarly on the evaluation tasks (see rows 1 and 2 in Table 1), we focus on the simpler CNN decoder that learns to reconstruct the next window of contiguous words.

5.2 Length of Target Sequences

We varied the length of target sequences in 3 cases, which are 10, 30 and 50, and measured the performance of 3 models on all tasks. As stated in rows 1, 3, and 4 in Table 1, short target sequences result in a slightly lower Pearson score on SICK, and long target sequences lead to a longer training time. We think that, longer target sequences result in a harder optimization task, and shorter ones lead

[1] http://pytorch.org/

[2] https://github.com/facebookresearch/SentEval

to a problem that not enough context information is included for every input sentence. A proper length of target sequences is able to balance these 2 issues. The following experiments set 30 contiguous words as the target sequence.

5.3 RNN DECODER VS. CNN DECODER

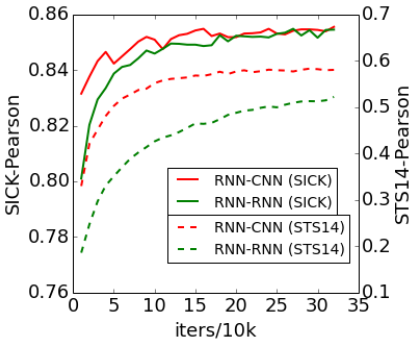

Figure 2: The comparison of our RNN-CNN model and the RNN-RNN model on supervised (SICK-Pearson) and unsupervised (STS14-Pearson) recorded at same training iterations. The left y-axis stands for SICK, and and right one stands for STS14. We can clearly see that our proposed RNN-CNN model outperforms the RNN-RNN model consistently through training. (Better view in color.)

With the same RNN encoder, the model with RNN decoder and the model with CNN decoder perform slightly better than Skip-thought model on STS14 and MSRP, and reach similar performance on other evaluation tasks as the Skip-thought model does. Since CNN decoder is able to parallelize computation across time steps, the model with it trains faster than that with RNN decoder.

The RNN-CNN model also demonstrates fast performance increase during training. We trained both models for a certain number of iterations, and recorded the performance on both supervised and unsupervised evaluation. The Figure 2 present the results. As shown in the figure, our RNN-CNN model consistently gets higher performance on SICK-Pearson (Supervised) and STS14-Pearson (Unsupervised) than RNN-RNN does.

To further understand the CNN decoder, we conducted experiments on the architecture. To explore the effect of number of layers, we added another convolution layer into the decoder; to explore the effect of number of channels, we doubled the channel size in each convolution layer while keeping the number of layers constant. Both models didn't give us significant performance increase over the basic RNN-CNN model. Therefore, in the following experiments, we fix the hyperparameter settings for the CNN decoder.

5.4 RNN ENCODER VS. CNN ENCODER

The CNN encoder we built followed the idea of AdaSent (Zhao et al., 2015), and we adopted the architecture proposed in (Conneau et al., 2017). The CNN encoder has 4 layers of convolution, each followed by a non-linear activation function. At every layer, a vector is calculated by a global max-pooling function over the feature map, and 4 vectors from 4 layers are concatenated to serve as a sentence representation. We tweaked the CNN encoder, including different kernel size and activation function, and we report the best results of CNN-CNN model at row 6 in Table 1.

Even searching over many hyperparameters and selecting the best performance on the evaluation tasks (overfitting), the CNN-CNN model performs poorly on the evaluation tasks, although the model trains much faster than any other models with RNNs (which were not similarly searched). The RNN and CNN are both non-linear systems, and they both are capable of learning complex composition functions on words in a sentence. We stated above that, in our belief, explicit usage of the word order information will augment the transferability of the encoder, and constrain the search space of the parameters in the encoder. The results match our belief.

The *future predictor* in Gan et al. (2017) also applies a CNN as the encoder, but the decoder is still an RNN, listed at row 11 in Table 1 . Compared to our designed CNN-CNN model, their CNN-LSTM model contains more parameters than our model does, but they have similar performance on the evaluation tasks, which is also worse than our RNN-CNN model.

5.5 DIMENSIONALITY

Clearly, we can tell from the comparison between rows 1, 9 and 12 in Table 1, increasing the dimensionality of the RNN encoder leads to better transferability of the model.

Compared with RNN-RNN model, even with double-sized encoder, the model with CNN decoder still runs faster than that with RNN decoder, and it slightly outperforms the model with RNN decoder on the evaluation tasks.

At the same dimensionality of representation with Skip-thought and Skip-thought+LN, our proposed RNN-CNN model performs better on all tasks but TREC, on which our model gets similar results as other models do.

Compared with the model with larger-size CNN decoder, apparently, we can see that larger encoder size helps more than larger decoder size does (rows 7,8, and 9 in Table 1).

In other words, an encoder with larger size will result in a representation with higher dimensionality, and generally, it will augment the expressiveness of the vector representation, and the transferability of the model.

5.6 BOOKCORPUS VS. AMAZON BOOK REVIEW

In terms of single sentence classification tasks, including MR, CR, SUBJ, MPQA, and SST, our models trained on the Amazon Book Review corpus largely outperform our models trained on BookCorpus. Our thought is that since the target domain of these tasks matches the source domain, which is Amazon Book Review data, the results are generally better. However, the results on unsupervised evaluation, such as STS14, are relatively worse than our models trained on BookCorpus. The results are presented in Table 2. The results show that our model is able to transfer knowledge from corpora of different domains, and they also point out that the Amazon Book Review could potentially be a suitable source for learning sentence representation.

6 RELATED WORK AND COMPARISON

Table 2 presented the results on 10 evaluation tasks of our proposed RNN-CNN models, and related work. "small RNN-CNN" refers to the model with the dimension of representation as 1200, and "large RNN-CNN" refers to that as 4800. The results of our model on SNLI can be found in Table 3.

6.1 UNSUPERVISED LEARNING WITH ORDERED SENTENCES

Skip-thought : Kiros et al. (2015) was the first one that applied learning from the context information into unsupervised representation learning for sentences, and later on, Ba et al. (2016) augmented the LSTM with proposed layer-normalization, which improved the skip-thought model generally on all downstream tasks. Both models demonstrated strong ability to transfer knowledge from unlabeled text data, with ordered sentences, to other NLP supervised evaluation tasks. The only issue that exists in both models is that training took so long since the dimensionality is huge, and it is almost impossible to reproduce the results from scratch, although they kindly released their trained models. Compared to Skip-thought, we cut off a branch for decoding, and combined a suite of simple techniques to enhance the transferability of our proposed RNN-CNN model, and the model reaches overall better results than both Skip-thought models.

FastSent : Motivated by the idea of Skip-thought, Hill et al. (2016) proposed FastSent, which is a simple yet effective algorithm for learning sentence representations, and it is also asymmetric. The model only learns source and target word embeddings, which is a generalization of CBOW (Mikolov et al., 2013b) to sentence-level learning, and the composition function over word embeddings is a summation operation. Since the model doesn't have any complex architecture, it could be trained within 2 hours, and outperforms Skip-thought on STS14, which shows a strong ability of transferring knowledge. However, FastSent performs poorly over all supervised evaluation tasks. Apparently, our RNN-CNN model works better on supervised evaluation tasks than FastSent does, but it still gets worse numbers on unsupervised evaluation (STS14). We think that since our RNN-CNN model has a similar training scheme and learning strategy to Skip-thought, the behavior of transferring knowledge

Encoder type	Encoder dim	Decoder type	Decoder dim	Hrs	SICK-r	SICK-E	STS14	MSRP (Acc/F1)	SST	TREC
Dimension of Sentence Representation: 1200										
RNN	2x300	CNN	600-1200-300	20	0.8530	82.6	0.58/0.56	**75.6/82.9**	82.8	**89.2**
		CNN†	600-1200-300	21	0.8515	82.7	0.58/0.56	75.3/82.5	**82.9**	85.2
		CNN(10)	600-1200-300	11	0.8474	82.9	0.57/0.55	74.2/81.6	82.8	88.0
		CNN(50)	600-1200-300	27	0.8533	82.5	0.57/0.55	74.7/82.2	81.5	86.2
RNN	2x300	RNN	600	26	0.8530	82.6	0.51/0.50	74.1/81.7	81.0	89.0
CNN	4x300§	CNN	600-1200-300	8	0.8117	80.5	0.44/0.42	72.7/80.7	78.4	85.0
RNN	2x300	CNN	600-1200-2400-300	28	**0.8570**	**84.0**	0.58/0.56	74.3/81.5	82.8	88.2
		CNN	1200-2400-300	27	0.8541	83.0	**0.59/0.57**	74.3/82.2	**82.9**	89.0
Dimension of Sentence Representation: 2400										
RNN	2x600	CNN	600-1200-300	25	0.8631	83.9	**0.58/0.55**	**74.7/83.1**	83.4	**90.2**
RNN	2x600	RNN	600	32	**0.8647**	**84.2**	0.52/0.51	74.0/81.2	**84.2**	87.6
CNN	3x800‡	RNN	600	8	0.8132	-	-	71.9/81.9	-	86.6
Dimension of Sentence Representation: 4800										
RNN	2x1200	CNN	600-1200-300	34	**0.8698**	**85.2**	**0.59/0.57**	**75.1/83.2**	**84.1**	**92.2**
Skip-thought (Kiros et al., 2015)				336	0.8584	82.3	0.29/0.35	73.0/82.0	82.0	**92.2**
Skip-thought+LN (Ba et al., 2016)				720	0.8580	79.5	0.44/0.45	-	82.9	88.4

Table 1: **Architecture Comparison**. As shown in the table, our designed asymmetric RNN-CNN model (row 1,9, and 12) works better than other asymmetric models (CNN-RNN, row 11), and models with symmetric structure (RNN-RNN, row 5 and 10). In addition, with larger encoder size, our model demonstrates stronger transferability. The default setting for our CNN decoder is that it learns to reconstruct 30 words right next to every input sentence. "CNN(10)" represents a CNN decoder with the length of outputs as 10, and "CNN(50)" represents it with the length of outputs as 50. "†" indicates that the CNN decoder learns to reconstruct next sentence. "‡" indicates the results reported in Gan et al. as *future predictor*. The CNN encoder in our experiment, noted as "§", was based on AdaSent in Zhao et al. and Conneau et al.. Bold numbers are best results among models at same dimension, and underlined numbers are best results among all models.

on to other evaluation tasks should be similar to the Skip-thought model. However, compared to Skip-thought, our model improves a lot on the unsupervised evaluations.

Model	**SNLI** (Acc %)
Unsupervised Transfer Learning	
Skip-thought (Vendrov et al.)	81.5
large RNN-CNN *BookCorpus*	**81.7**
large RNN-CNN *Amazon*	81.5
Supervised Training	
ESIM (Chen et al.)	86.7
DIIN (Gong et al.)	**88.9**

Table 3: We implemented the same classifier as mentioned in Vendrov et al. (2015) on top of the features computed by our model. Our proposed RNN-CNN model gets similar result on SNLI as Skip-thought, but with much less training time.

CNN-LSTM : Later on, Gan et al. (2017) first applied a CNN as the encoder in an unsupervised fashion for learning sentence representation, which is an asymmetric CNN-RNN structure with context-based training scheme. The proposed composition model follows the idea of encoding the current sentence and predicting the surrounding 2 sentences (the previous one and the next one); the proposed hierarchical model leverages the context information from both sentence-level and paragraph-level, while learning to encode the current sentence and predict the next one, the model has another RNN to process the sentence representation one at a time at paragraph-level. Each of the models slightly outperforms Skip-thought on supervised evaluation tasks, and the combination of 2 models, which is listed in Table 2 as "combine CNN-LSTM", shows a significant improvement over each single one. Our proposed model has similar performance to the combined model, and it has better results than each of the 2 models alone. Sadly, Gan et al. (2017) didn't mention the training speed for any of the 2 models, but we expect

Model	Hrs	SICK-r	SICK-E	STS14	MSRP	TREC	MR	CR	SUBJ	MPQA	SST
Unsupervised training with unordered sentences											
Unigram-TFIDF	-	-	-	-	**73.6/81.7**	**85.0**	73.7	79.2	90.3	82.4	-
ParagraphVec	4	-	-	0.42/0.43	72.9/81.1	59.4	60.2	66.9	76.3	70.7	-
word2vec BOW	2	**0.8030**	**78.7**	**0.65/0.64**	72.5/81.4	83.6	77.7	**79.8**	90.9	**88.3**	79.7
fastText BOW	-	0.8000	77.9	0.63/0.62	72.4/81.2	81.8	76.5	78.9	**91.6**	87.4	78.8
GloVe BOW	-	0.8000	78.6	0.54/0.56	72.1/80.9	83.6	**78.7**	78.5	**91.6**	87.6	**79.8**
SDAE	72	-	-	0.37/0.38	73.7/80.7	78.4	74.6	78.0	90.8	86.9	-
Unsupervised training with ordered sentences - BookCorpus											
DiscSent‡	8	-	-	-	75.0/ -	87.2	-	-	93.0	-	-
FastSent	2	-	-	**0.63/0.64**	72.2/80.3	76.8	70.8	78.4	88.7	80.6	-
FastSent+AE	2	-	-	0.62/0.62	71.2/79.1	80.4	71.8	76.5	88.8	81.5	-
Skip-thought	336	0.8580	82.3	0.29/0.35	73.0/82.0	92.2	76.5	80.1	93.6	87.1	82.0
Skip-thought+LN	720	0.8580	79.5	0.44/0.45	-	88.4	79.4	**83.1**	93.7	89.3	82.9
combine CNN-LSTM	-	0.8618	-	-	**76.5/83.8**	**92.6**	77.8	82.1	93.6	**89.4**	-
***small RNN-CNN*†**	20	0.8530	82.6	0.58/0.56	75.6/82.9	89.2	77.6	80.3	92.3	87.8	82.8
***large RNN-CNN*†**	34	**0.8698**	**85.2**	0.59/0.57	75.1/83.2	92.2	**79.7**	81.9	**94.0**	88.7	**84.1**
Unsupervised training with ordered sentences - Amazon Book Review											
***small RNN-CNN*†**	21	0.8476	82.7	**0.53/0.53**	73.8/81.5	84.8	83.3	83.0	94.7	88.2	87.8
***large RNN-CNN*†**	33	**0.8616**	**84.3**	0.51/0.51	**75.7/82.8**	**90.8**	85.3	86.8	**95.3**	**89.0**	**88.3**
Unsupervised training with ordered sentences - Amazon Review											
BYTE m-LSTM	-	0.7920	-	-	75.0/**82.8**	-	**86.9**	**91.4**	94.6	88.5	-
Supervised training - Transfer learning											
NMT En-to-Fr	72	-	-	0.43/0.42	-	82.8	64.7	70.1	84.9	81.5	-
CaptionRep BOW	24	-	-	0.46/0.42	-	72.2	61.9	69.3	77.4	70.8	-
DictRep BOW	24	-	-	0.67/**0.70**	68.4/76.8	81.0	76.7	78.7	90.7	87.2	-
BiLSTM-Max(SNLI)	<24	**0.8850**	84.6	0.68/0.65	75.1/82.3	**88.7**	79.9	84.6	92.1	89.8	83.3
BiLSTM-Max(AllNLI)	<24	0.8840	**86.3**	**0.70**/0.67	**76.2/83.1**	88.2	**81.1**	**86.3**	**92.4**	**90.2**	**84.6**
Supervised task-dependent training - No transfer learning											
NB-SVM	-	-	-	-	-	-	79.4	81.8	93.2	86.3	83.1
AdaSent	-	-	-	-	-	92.4	83.1	86.3	95.5	93.3	-
Tree-LSTM	-	0.8680	-	-	-	-	-	-	-	-	-
TF-KLD	-	-	-	-	80.4/85.9	-	-	-	-	-	-

Table 2: **Related Work and Comparison.** As presented in the table, our designed asymmetric RNN-CNN model has a strong transferability, and overall better than existing models in terms of fast training speed and good performance on evaluation tasks. The table presents the model comparison. "†"s refer to our models, and "*small*" refers to the dimension of representation as 1200, and "*large*" refers to that as 4800. "‡" indicates that DiscSent model was trained with additional data from Wikipedia and the Gutenberg project. Bold numbers are the best ones among the models with same training and transferring setting, and underlined numbers are best results among all unsupervised representation learning models.

that our model should run faster than theirs during training, since both their models use LSTM for decoding, and the hierarchical model has another LSTM running at paragraph-level.

DiscSent : A recent paper pushes the idea of learning from the context information in another direction. In Jernite et al. (2017), The model encodes two sentences in two representations, respectively, and the classifier on top of the representations judges 1) whether the two sentences are adjacent to each other, 2) whether the two sentences are in the correct order, and 3) whether the second sentence starts with a conjunction phrase. The paper only reported the results on MSRP, TREC and SUBJ, which is listed as "DiscSent" in Table 2, and they are worse than other methods. The interesting point in the paper is that they first trained their model only on BookCorpus dataset, and it didn't work. Then they chose to add more data from the Gutenberg project (Stroube, 2003) and Wikipedia, and the model started to show relatively strong transferability. The fact that the model worked with more data along with BookCorpus could be evidence showing that the contextual supervision from the

contiguous sentences in the BookCorpus dataset might be too vague, in other words, it might be hard for models to learn from the context available in the BookCorpus dataset. Further investigation needs to be done to determine the suitability of BookCorpus and similar corpora for unsupervised learning.

BYTE m-LSTM : Other potential corpora that researchers have started investigating include Wikipedia, Amazon Review data (McAuley et al., 2015), etc. Radford et al. (2017) applied the multiplicative-LSTM (Krause et al., 2016) on language modeling on Amazon Review data. We argue that their language modeling still follows the idea of exploiting the context information, since the hidden state at the end of the current sentence is used as the initial hidden state to predict the next sentences. A performance gain relative to previous existing models can be found on MR and CR, but the model performs poorly on SICK dataset. We think that the performance boost was brought by domain matching among movie review (MR), customer review (CR), and the source domain (Amazon Review data). However, the hidden state could bring information far before the current sentence which would likely decrease the chance of discovering the semantic similarity between the adjacent sentences. Our large RNN-CNN model trained on Amazon Book Review (the largest subset of Amazon Review) performs on par with BYTE m-LSTM model, and ours works better than theirs on semantic relatedness and entailment tasks.

6.2 Unsupervised Learning with unordered Sentences

ParagraphVec : Previously, Mikolov et al. (2013a) proposed the continuous bag-of-words (CBOW) model and the skip-gram model for distributed word representation learning. The main idea is to learn a word representation by discovering the context information from the surrounding words. Mikolov et al. (2013b) improved the skip-gram model, and empirically showed that additive composition of the learned word representations successfully captures contextual information of phrases and sentences, which is a strong baseline model for NLP tasks. Similarly, Le & Mikolov (2014) proposed a method to learn a fixed-dimension vector for each sentence by predicting the words within the given sentence. However, after training, the representation for a new sentence is hard to derive, since it requires optimizing the sentence representation towards an objective.

SDAE : In Hill et al. (2016), the other model proposed is Sequential Denoising Auto-encoder (SDAE). The noise was added in the encoder by replacing words with a fixed token, and swapping two words, both with a specific probability. Compared with those models trained with ordered sentences, SDAE has a focus on the sentence itself during training, and shows competitive results on supervised evaluation tasks, while it has poor performance on STS14. Our proposed RNN-CNN model trains faster than SDAE does, since the CNN decoder runs faster than the RNN decoder, and since we utilized the sentential continuity as a supervision which SDAE doesn't, our model largely performs better than SDAE.

6.3 Supervised Transfer Learning

BiLSTM-Max : Supervised transfer learning is also promising when we are able to get large enough labeled data. Conneau et al. (2017) applied a simple bi-directional LSTM as the sentence encoder with multiple fully-connected layer to deal with both SNLI (Bowman et al., 2015), and MultiNLI (Williams et al., 2017). The trained model demonstrates a very impressive transferability on all downstream tasks, including both supervised and unsupervised. The direct and discriminative training signal pushes the RNN encoder to focus on the semantics of a given sentence, which learns to a boost in performance, and beats all other methods. Our RNN-CNN model trained on Amazon Book Review data has better results on supervised classification tasks than BiLSTM-Max does, while the performance of ours on semantic relatedness tasks is inferior to BiLSTM-Max. We argue that labeling a large amount of training data is time-consuming and costly, and working on unsupervised learning could potentially provide a great initial point for human labeling, which can make it less costly and more efficient.

7 Conclusion

Inspired by learning to exploit the contextual information present in adjacent sentences, we proposed an asymmetric encoder-decoder model with a suite of techniques for improving context-based unsupervised sentence representation learning. Since we believe that a simple model will be faster in

training and easier to analyze, we opt to use simple techniques in our proposed model, including 1) an RNN as the encoder, and a CNN as the decoder, 2) learning by inferring next contiguous words, 3) mean+max pooling, and 4) tying word vectors with word prediction. With thorough discussion and extensive evaluation, we justify our decision making for each component in our RNN-CNN model. In terms of the performance and the efficiency of training, we justify that our model is a fast and simple algorithm for learning generic sentence representations from unlabeled corpora. Further research will focus on how to maximize the utility of the context information, and how to design simple architectures to best make use of it.

REFERENCES

Gerry Altmann and Jelena Mirkovic. Incrementality and prediction in human sentence processing. *Cognitive science*, 33 4:583–609, 2009.

Jimmy Ba, Ryan Kiros, and Geoffrey E. Hinton. Layer normalization. *CoRR*, abs/1607.06450, 2016.

Jeffrey R Binder and Rutvik H Desai. The neurobiology of semantic memory. *Trends in cognitive sciences*, 15 11:527–36, 2011.

Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. A large annotated corpus for learning natural language inference. In *EMNLP*, 2015.

Qian Chen, Xiaodan Zhu, Zhenhua Ling, Si Wei, and Hui Jiang. Enhancing and combining sequential and tree lstm for natural language inference. *arXiv preprint arXiv:1609.06038*, 2016.

Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*, 2014.

Alexis Conneau, Douwe Kiela, Holger Schwenk, Loïc Barrault, and Antoine Bordes. Supervised learning of universal sentence representations from natural language inference data. In *EMNLP*, 2017.

William B. Dolan, Chris Quirk, and Chris Brockett. Unsupervised construction of large paraphrase corpora: Exploiting massively parallel news sources. In *COLING*, 2004.

Zhe Gan, Yunchen Pu, Ricardo Henao, Chunyuan Li, Xiaodong He, and Lawrence Carin. Learning generic sentence representations using convolutional neural networks. In *EMNLP*, 2017.

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

Zellig S Harris. Distributional structure. *Word*, 10(2-3):146–162, 1954.

Felix Hill, Kyunghyun Cho, and Anna Korhonen. Learning distributed representations of sentences from unlabelled data. In *HLT-NAACL*, 2016.

Sepp Hochreiter and Juergen Schmidhuber. Long short-term memory. *Neural Computation*, 9: 1735–1780, 1997.

Minqing Hu and Bing Liu. Mining and summarizing customer reviews. In *KDD*, 2004.

Hakan Inan, Khashayar Khosravi, and Richard Socher. Tying word vectors and word classifiers: A loss framework for language modeling. *CoRR*, abs/1611.01462, 2016.

Yacine Jernite, Samuel R. Bowman, and David Sontag. Discourse-based objectives for fast unsupervised sentence representation learning. *CoRR*, abs/1705.00557, 2017.

Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Jamie Ryan Kiros, Yukun Zhu, Ruslan Salakhutdinov, Richard S. Zemel, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. Skip-thought vectors. In *NIPS*, 2015.

Ben Krause, Liang Lu, Iain Murray, and Steve Renals. Multiplicative lstm for sequence modelling. *CoRR*, abs/1609.07959, 2016.

Quoc V. Le and Tomas Mikolov. Distributed representations of sentences and documents. In *ICML*, 2014.

Xin Li and Dan Roth. Learning question classifiers. In *COLING*, 2002.

Tsung-Yi Lin, Michael Maire, Serge J. Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft coco: Common objects in context. In *ECCV*, 2014.

Marco Marelli, Stefano Menini, Marco Baroni, Luisa Bentivogli, Raffaella Bernardi, and Roberto Zamparelli. A sick cure for the evaluation of compositional distributional semantic models. In *LREC*, 2014.

Julian J. McAuley, Christopher Targett, Qinfeng Shi, and Anton van den Hengel. Image-based recommendations on styles and substitutes. In *SIGIR*, 2015.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013a.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Gregory S. Corrado, and Jeffrey Dean. Distributed representations of words and phrases and their compositionality. In *NIPS*, 2013b.

Bo Pang and Lillian Lee. A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. In *ACL*, 2004.

Bo Pang and Lillian Lee. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In *ACL*, 2005.

Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. On the difficulty of training recurrent neural networks. In *ICML*, 2013.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. Glove: Global vectors for word representation. In *EMNLP*, 2014.

Tamara Polajnar, Laura Rimell, and Stephen Clark. An exploration of discourse-based sentence spaces for compositional distributional semantics. In *Workshop on LSDSem*, pp. 1, 2015.

Ofir Press and Lior Wolf. Using the output embedding to improve language models. In *EACL*, 2017.

Alec Radford, Rafal Józefowicz, and Ilya Sutskever. Learning to generate reviews and discovering sentiment. *CoRR*, abs/1704.01444, 2017.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. Recursive deep models for semantic compositionality over a sentiment treebank. 2013.

Bryan Stroube. Literary freedom: project gutenberg. *ACM Crossroads*, 10:3, 2003.

Shuai Tang, Hailin Jin, Chen Fang, Zhaowen Wang, and Virginia R. de Sa. Rethinking skip-thought: A neighborhood based approach. In *RepL4NLP, ACL Workshop*, 2017.

Ivan Vendrov, Jamie Ryan Kiros, Sanja Fidler, and Raquel Urtasun. Order-embeddings of images and language. *CoRR*, abs/1511.06361, 2015.

Janyce Wiebe, Theresa Wilson, and Claire Cardie. Annotating expressions of opinions and emotions in language. *Language Resources and Evaluation*, 39:165–210, 2005.

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

Xiang Zhang and Yann LeCun. Which encoding is the best for text classification in chinese, english, japanese and korean? *CoRR*, abs/1708.02657, 2017.

Yizhe Zhang, Dinghan Shen, Guoyin Wang, Zhe Gan, Ricardo Henao, and Lawrence Carin. Deconvolutional paragraph representation learning. *CoRR*, abs/1708.04729, 2017.

Han Zhao, Zhengdong Lu, and Pascal Poupart. Self-adaptive hierarchical sentence model. In *IJCAI*, 2015.

Yukun Zhu, Ryan Kiros, Richard S. Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. *ICCV*, pp. 19–27, 2015.