A Supplemental Material

Table 1 presents the effect of hyperparameters.

A.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 subsequent 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 predict-all-words CNN decoder that learns to reconstruct the next window of contiguous words.

A.2 Length of the Target Sequence T

We varied the length of target sequences in three cases, which are 10, 30 and 50, and measured the performance of three models on all tasks. As stated in rows 1, 3, and 4 in Table 1, decoding short target sequences results in a slightly lower Pearson score on SICK, and decoding longer target sequences lead to a longer training time. In our understanding, decoding longer target sequences leads to a harder optimisation task, and decoding shorter ones leads 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 two issues. The following experiments set subsequent 30 contiguous words as the target sequence.

A.3 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 four 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 time, and four vectors from four layers are concatenated to serve as the 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 hypothesised that the 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 support our hypothesis.

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.

A.4 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.

B Experimental Details

Our **small** RNN-CNN model has a bi-directional GRU as the encoder, with 300 dimension each di-

Encoder		Decoder		Hrs	SICK-R	SICK-E	STS14	MSRP	SST	TREC
type	dim	type	dim					(Acc/F1)		
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-LSTM, 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. For STS14, the performance measures are Pearson's and Spearman's score. For MSRP, the performance measures are accuracy and F1 score.

rection, and the **large** one has 1200 dimension GRU in each direction. The batch size we used for training our model is 512, and the sequence length for both encoding and decoding are 30. The initial learning rate is 0.0005, and the Adam optimiser (Kingma and Ba, 2014) is applied to tune the parameters in our model.

C Results including supervised task-dependent models

Table 2 contains all supervised task-dependent models for comparison.

References

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

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

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

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

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

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

Model	Hrs	SICK-R	SICK-E	STS14	MSRP	TREC	MR	CR	SUBJ	MPQA	SST
Measurement		r	Acc.	r/ρ	Acc./F1	Accuracy					
Unsupervised training with unordered sentences											
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
SIF (GloVe+WR)	-	**0.8603**	**84.6**	**0.69**/ -	- / -	-	-	-	-	-	**82.2**
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											
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	720	0.7920	-	-	75.0/**82.8**	-	**86.9**	**91.4**	94.6	88.5	-
Supervised training - Transfer learning											
DiscSent	8	-	-	-	75.0/ -	87.2	-	-	93.0	-	-
DisSent Books 8	-	0.8170	81.5	-	-/ -	87.2	**82.9**	81.4	**93.2**	90.0	80.2
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	-
InferSent(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
InferSent(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**

Table 2: **Related Work and Comparison.** As presented, our designed asymmetric RNN-CNN model has strong transferability, and is overall better than existing unsupervised models in terms of fast training speed and good performance on evaluation tasks. "†"s refer to our models, and "**small/large**" refers to the dimension of representation as 1200/4800. **Bold** numbers are the best ones among the models with same training and transferring setting, and underlined numbers are best results among all transfer learning models. The training time of each model was collected from the paper that proposed it.