A Appendix

A.1 Neural Machine Translation with ATR

We replace LSTM/GRU with our proposed ATR to build NMT models under the attention-based encoder-decoder framework (Bahdanau et al., 2015). The encoder that reads a source sentence is a bidirectional recurrent network. Formally, given a source sentence $\mathbf{x} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\}$, the encoder is formulated as follows:

$$\overrightarrow{\mathbf{h}}_i = \text{ATR}(\overrightarrow{\mathbf{h}}_{i-1}, \mathbf{x}_i), \overleftarrow{\mathbf{h}}_i = \text{ATR}(\overleftarrow{\mathbf{h}}_{i+1}, \mathbf{x}_i) \quad (11)$$

where $\text{ATR}(\cdot)$ is defined by Equation (6&9). The forward $\overrightarrow{\mathbf{h}}_i$ and backward $\overleftarrow{\mathbf{h}}_i$ hidden states are concatenated together to represent the i-th word: $\mathbf{h}_i = [\overrightarrow{\mathbf{h}}_i; \overleftarrow{\mathbf{h}}_i]$.

The decoder is a conditional language model that predicts the j-th target word via a multilayer perception:

$$p(y_j|\mathbf{x}, \mathbf{y}_{<j}) = \text{softmax}(g(\mathbf{y}_{j-1}, \tanh(\mathbf{s}_j), \mathbf{c}_j)) \quad (12)$$

where $\mathbf{y}_{<j}$ is a partial translation. $\mathbf{c}_j$ is the translation-sensitive semantic vector computed via the attention mechanism (Bahdanau et al., 2015) based on the source states $\{\tanh(\mathbf{h}_i)\}_{i=1}^n$ and internal target state $\widetilde{\mathbf{s}}_j$, and $\mathbf{s}_j$ is the j-th target-side hidden state calculated through a two-level hierarchy:

$$\widetilde{\mathbf{s}}_j = \text{ATR}(\mathbf{s}_{j-1}, \mathbf{y}_{j-1}), \mathbf{s}_j = \text{ATR}(\widetilde{\mathbf{s}}_j, \mathbf{c}_j) \quad (13)$$

A.2 Additional Experiments

A.2.1 Experiments on Chinese-English Translation

Our training data consists of 1.25M sentence pairs including 27.9M Chinese words and 34.5M English words respectively.[1] We used the NIST 2005 dataset as our dev set, and the NIST 2002, 2003, 2004, 2006 and 2008 datasets as our test sets. Unlike WMT14 translation tasks, we used word-based vocabulary for Chinese-English, preserving top-30K most frequent source and target words in the vocabulary. Case-insensitive BLEU-4 metric was used to evaluate the translation quality.

Translation Results

We compare our model against several advanced models on the same dataset, including:

- *Coverage (Wang et al., 2017)*: an attention-based NMT system enhanced with a coverage mechanism to handle the over-translation and under-translation problem.
- *MemDec (Wang et al., 2017)*: an attention-based NMT system that replaces the vanilla decoder with a memory-enhanced decoder to better capture important information for translation.
- *DeepLAU (Wang et al., 2017)*: a deep attention-based NMT system integrated with linear associative units that deals better with gradient propagation.
- *Distortion (Zhang et al., 2017c)*: an attention-based NMT system that incorporates word reordering knowledge to encourage more accurate attention.
- *CAEncoder (Zhang et al., 2017b)*: the same as our model but uses GRU unit.
- *FPNMT (Zheng et al., 2017)*: an attention-based NMT system that leverages past and future information to improve the attention model and the decoder states, also using addition and subtraction operations.
- *ASDBNMT (Zhang et al., 2018b)*: an attention-based NMT system that is equipped with a backward decoder to explore bidirectional decoding.

Table 5 summarizes the results. Although our model does not involve any sub-networks for modeling the coverage, distortion, memory and future context, our model clearly outperforms all these advanced models, achieving an average BLEU score of 39.82 on all test sets. This strongly suggests that 1) shallow models are also capable of generating extremely high-quality translations, and 2) our ATR model indeed has the ability in capturing translation correspondence in spite of its simplicity.

A.2.2 Experiments on Natural Language Inference

Given two sentences, namely a premise and a hypothesis, this task aims at recognizing whether the premise can entail the hypothesis. We used

[1] This corpora contain LDC2002E18, LDC2003E07, LDC2003E14, Hansards portion of LDC2004T07, LDC2004T08 and LDC2005T06.

System	MT05	MT02	MT03	MT04	MT06	MT08
Existing Systems						
Coverage (Wang et al., 2017)	34.91	-	34.49	38.34	34.25	-
MemDec (Wang et al., 2017)	35.91	-	36.16	39.81	35.98	-
DeepLAU (Wang et al., 2017)	38.07	-	39.35	41.15	37.29	-
Distortion (Zhang et al., 2017c)	36.71	-	38.33	40.11	35.29	-
CAEncoder (Zhang et al., 2017b)	36.44	40.12	37.63	39.83	35.44	27.34
FPNMT (Zheng et al., 2017)	36.75	39.65	37.90	40.37	34.55	-
ASDBNMT (Zhang et al., 2018b)	38.84	-	40.02	42.32	38.38	-
Our end-to-end NMT systems						
this work	**39.71**	**42.95**	**41.71**	**43.71**	**39.61**	**31.14**

Table 5: Case-insensitive BLEU scores of advanced systems on the Chinese-English translation tasks. "–" indicates that no result is provided in the original paper.

the Stanford Natural Language Inference Corpus (SNLI) (Bowman et al., 2015) for this experiment, which involves a collection of 570k human-written English sentence pairs manually labeled for balanced classification with the labels *entailment*, *contradiction*, and *neutral*. We formulated this problem as a three-way classification task.

We employed the attentional architecture (Rocktäschel et al., 2016) as our basic model, and replaced its recurrent unit with our ATR model. We fixed word embedding initialized with the pre-trained 300-D Glove vector (Pennington et al., 2014). The hidden size of ATR was also set to 300. We optimized model parameters using the Adam method (Kingma and Ba, 2015) with hyperparameters $\beta_1 = 0.9$ and $\beta_2 = 0.999$. The learning rate was fixed at 0.0005. The mini-batch size was set to 128. Dropout was applied on both word embedding layer and pre-classification layer to avoid overfitting, with a rate of 0.15. The maximum training epoch was set to 20.

Classification Results

Table 6 shows the results. The GRU equipped model in our implementation achieves a test accuracy of 84.6% with about 3.2m trainable model parameters, outperforming the LSTM-enhanced counterpart (Rocktäschel et al., 2016) by a margin of 1.1%. By contrast, the same architecture with ATR model yields a test accuracy of 85.6%, with merely 1.5m model parameters. In other words, using fewer parameters, our ATR model gains a significant improvement of 1.0%, reaching a comparable performance against some deep architectures (Cheng et al., 2016).

A.2.3 Experiments on Chinese Word Segmentation

Chinese word segmentation (CWS) is a fundamental preprocessing step for Chinese-related NLP tasks. Unlike other languages, Chinese sentences are recorded without explicit delimiters. Therefore, before performing in-depth modeling, researchers need to segment the whole sentence into a sequence of tokens, which is exactly the goal of CWS.

Following previous work (Chen et al., 2015), we formulate CWS as a sequence labeling task. Each character in a sentence is assigned with a unique label from the set $\{B, M, E, S\}$, where $\{B, M, E\}$ indicate *Begin, Middle, End* of a multi-character word respectively, and S denotes a *Single* character word. Given a sequence of characters, we first embed them individually through a character embedding layer, followed by a bidirectional RNN layer to generate context-sensitive representation for each character. The output representations are then passed through a CRF inference layer to capture dependencies among character labels. The whole model is optimized using a max-margin objective towards minimizing the differences between predicted sequences and gold label sequences.

We used the MSRA and CTB6 datasets to evaluate our model. The former is provided by the second International Chinese Word Segmentation Bakeoff (Sproat and Emerson, 2003), and the latter is from Chinese TreeBank6.0 (LDC2007T36) (Xue et al., 2005). For MSRA dataset, we split the first 90% sentences of the training data as the training set and the rest as the development set. For CTB6 dataset, we divided the training, development and test sets in the same way as in (Chen et al., 2015). Precision, recall, F1-score and out-of-vocabulary (OOV) word recall calculated by the standard back-off scoring program were used for evaluation.

We set the dimensionality of both character

Model	d	#Params	Train	Test
LSTM encoders (Bowman et al., 2016)	300	3.0m	83.9	80.6
GRU encoders w/ pretraining (Vendrov et al., 2015)	1024	15m	98.8	81.4
BiLSTM encoders with intra-attention (Liu et al., 2016)	600	2.8m	84.5	84.2
LSTMs w/ word-by-word attention (Rocktäschel et al., 2016)	100	250k	85.3	83.5
mLSTM word-by-word attention model (Wang and Jiang, 2016)	300	1.9m	92.0	86.1
LSTMN with deep attention fusion (Cheng et al., 2016)	450	3.4m	88.5	86.3
BiMPM (Wang et al., 2017b)	100	1.6m	90.9	87.5
this work with GRU	300	3.2m	91.0	84.6
this work with ATR	300	1.5m	90.9	85.6

Table 6: Classification results on the SNLI task. For comparison, we provide the model dimension (d), the parameter amount (# Params), the training accuracy (Train) and the test accuracy (Test). m: million. We also provide results of several existing RNN models from the SNLI official website.

Model	MSRA			CTB6		
	P	R	F	P	R	F
(Zheng et al., 2013)	92.9	93.6	93.3	94.0	93.1	93.6
(Pei et al., 2014)	94.6	94.2	94.4	94.4	93.4	93.9
(Chen et al., 2015)	96.7	96.2	96.4	95.0	94.8	94.9
this work + LSTM	95.5	94.9	95.2	93.3	93.1	93.2
this work + GRU	95.2	95.1	95.1	93.3	93.0	93.2
this work + ATR	95.3	95.1	95.2	94.0	93.9	94.0

Table 7: Model performance on MSRA and CTB6 datasets. We report precision (P), recall (R) and F1-score (F).

embedding and RNN hidden state to be 300. Model parameters were tuned by Adam algorithm (Kingma and Ba, 2015) with default hyper-parameters ($\beta_1 = 0.9, \beta_2 = 0.999$) and mini-batch size 128. Gradient was clipped when its norm exceeds 1.0 to avoid gradient explosion. We applied dropout on both character embedding layer and pre-CRF layer with a rate of 0.2. The discount parameter in max-margin objective was set to 0.2. The maximum training epoch was set to 50. Learning rate was initially set to 0.0005, and halved after each epoch.

Model Performance

Table 7 shows the overall performance. We observe that our ATR model performs as efficient as both GRU and LSTM on this task. ATR yields a F1-score of 95.2% and 94.0% on MSRA and CTB6 dataset respectively, almost the same as that of GRU (95.1%/93.2%) and LSTM (95.2%/93.2%). Particularly, ATR achieves better results on CTB6, with a gain of 0.8% F1 points over GRU and LSTM. This further demonstrates the effectiveness of the proposed ATR model.