DTMT: A Novel Deep Transition Architecture for Neural Machine Translation

Fandong Meng, Jinchao Zhang
WeChat AI - Pattern Recognition Center Tencent Inc.
{fandongmeng, dayerzhang}@tencent.com

Abstract

Past years have witnessed rapid developments in Neural Machine Translation (NMT). Most recently, with advanced modeling and training techniques, the RNN-based NMT (RNMT) has shown its potential strength, even compared with the well-known Transformer (self-attentional) model. Although the RNMT model can possess very deep architectures through stacking layers, the transition depth between consecutive hidden states along the sequential axis is still shallow. In this paper, we further enhance the RNN-based NMT through increasing the transition depth between consecutive hidden states and build a novel Deep Transition RNN-based Architecture for Neural Machine Translation, named DTMT. This model enhances the hidden-to-hidden transition with multiple non-linear transformations, as well as maintains a linear transformation path throughout this deep transition by the well-designed linear transformation mechanism to alleviate the gradient vanishing problem. Experiments show that with the specially designed deep transition modules, our DTMT can achieve remarkable improvements on translation quality. Experimental results on Chinese⇒English translation task show that DTMT can outperform the Transformer model by +2.09 BLEU points and achieve the best results ever reported in the same dataset. On WMT14 English⇒German and English⇒French translation tasks, DTMT shows superior quality to the state-of-the-art NMT systems, including the Transformer and the RNMT+.

Introduction

Neural Machine Translation (NMT) with an encoder-decoder (Cho et al. 2014; Sutskever, Vinyals, and Le 2014) framework has made promising progress in recent years. Generally, this kind of framework consists of two components: an encoder network that encodes the input sentence into a sequence of distributed representations, based on which a decoder network generates the translation with an attention mechanism (Bahdanau, Cho, and Bengio 2015; Luong, Pham, and Manning 2015). Driven by the breakthrough achieved in computer vision (He et al. 2016), research in NMT have turned towards studying deep architectures (Wu et al. 2016; Zhou et al. 2016; Kalchbrenner et al. 2017; Wang et al. 2017; Gehring et al. 2017; Vaswani et al. 2017). Among these studies, RNN-based NMT (RNMT) with deep stacked architectures (Wu et al. 2016; Zhou et al. 2016; Kalchbrenner et al. 2017; Wang et al. 2017) first outperforms the conventional shallow RNMT model, to be the de-facto standard for NMT. Most recently, after absorbing the advanced modeling and training techniques, the RNMT+ (Chen et al. 2018) has shown its greater potential strength, even surpasses the convolutional seq2seq (ConvS2S) model (Gehring et al. 2017) and achieves comparable results with the well-known Transformer model (Vaswani et al. 2017). These studies inspire researchers to make efforts for searching new architectures for RNMT.

Although the RNMT model can possess very deep architectures through stacking layers, for each recurrent level of the stacked RNN, the transition between the consecutive hidden states along the sequential axis is still shallow. Since the state transition between the consecutive hidden states effectively adds a new input to the summary of the previous inputs represented by the hidden state, this procedure of constructing a new summary from the combination of the previous one and the new input should be highly nonlinear, to allow the hidden state to rapidly adapt to quickly changing modes of the input while still preserving a useful summary of the past (Pascanu et al. 2014). From this perspective, some researchers (Pascanu et al. 2014; Barone et al. 2017) investigate deep transition recurrent architectures, which increase the depth of the hidden-to-hidden transition. This kind of structure extends the conventional shallow RNN in another aspect different from the stacked RNN, and has been proven to outperform the stacked one on language modeling task (Pascanu et al. 2014). Barone et al. (2017) apply this transition architecture to RNMT, while there is still a large margin between this transition model and the state-of-the-art model, e.g. the Transformer (Vaswani et al. 2017), in terms of BLEU, which is also confirmed by Tang et al. (2018).

In this paper, we further enhance the RNMT through increasing the transition depth of the consecutive hidden states along the sequential axis and build a novel and effective Deep Transition RNN-based Architecture for Neural Machine Translation, named DTMT. We design three deep transition modules, which correspondingly extend the RNN modules of shallow RNMT in the encoder and the decoder, to enhance the non-linear transformation between consecutive hidden states. Since the deep transition increases the number of nonlinear steps, this may lead to the problem of vanishing gradients. To alleviate this problem, we propose a Linear Transformation enhanced Gated Recurrent Unit (L-GRU) for DTMT, which provides a linear transformation path throughout the deep transition.

We test the effectiveness of our DTMT on Chinese⇒English, English⇒German and English⇒French translation tasks. Experimental results on NIST Chinese⇒English translation show that DTMT can outperform the Transformer model by +2.09 BLEU points and achieve the best results ever reported in the same dataset. On WMT14 English⇒German and English⇒French translation, it consistently leads to substantial improvements and shows superior quality to the state-of-the-art NMT systems (Vaswani et al. 2017; Cheng et al. 2018). The main contributions of this paper can be summarized as follows:

- We tap the potential strength of deep transition between consecutive hidden states and propose a novel deep transition RNN-based architecture for NMT, which achieves state-of-the-art results on multiple translation tasks.
- We propose a simple yet more effective linear transformation enhanced GRU for our deep transition RNMT, which provides a linear transformation path for deep transition of consecutive hidden states. Additionally, L-GRU can also be used to enhance other GRU-based architectures, such as the shallow RNMT and the stacked RNMT.
- We apply recent advanced techniques, including multi-head attention, layer normalization, label smoothing, and dropouts to enhance our DTMT. Additionally, we find the positional encoding (Vaswani et al. 2017) can assist the training of RNMT by modeling positions of the tokens in the sequence, although it is originally designed for the non-recurrent (self-attentional) architecture.

Background

Attention-based RNMT

Given a source sentence $\mathbf{x} = \{x_1, x_2, \cdots, x_n\}$ and a target sentence $\mathbf{y} = \{y_1, y_2, \cdots, y_m\}$, RNN-based neural machine translation (RNMT) models the translation probability word by word:

$$\begin{aligned} p(\mathbf{y}|\mathbf{x}) &= \prod_{t=1}^{m} P(y_t|\mathbf{y_{<t}}, \mathbf{x}; \theta) \\ &= \prod_{t=1}^{m} softmax(f(\mathbf{c}_t, y_{t-1}, \mathbf{s}_t)) \end{aligned} \tag{1}$$

where $f(\cdot)$ is a non-linear function, and $\mathbf{s}_t$ is the hidden state of decoder RNN at time step t:

$$\mathbf{s}_t = g(\mathbf{s}_{t-1}, y_{t-1}, \mathbf{c}_t) \tag{2}$$

$\mathbf{c}_t$ is a distinct source representation for time t, calculated as a weighted sum of the source annotations:

$$\mathbf{c}_t = \sum_{j=1}^{n} a_{t,j}\mathbf{h}_j \tag{3}$$

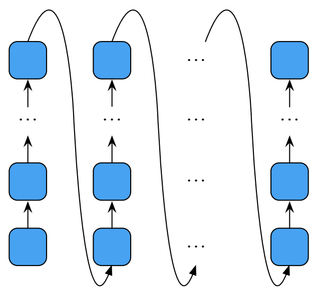

Figure 1: Deep transition RNN, in which the transition between consecutive hidden states is deep.

Formally, $\mathbf{h}_j = [\overrightarrow{\mathbf{h}}_j, \overleftarrow{\mathbf{h}}_j]$ is the annotation of x_j, which can be computed by a bi-directional RNN (Schuster and Paliwal 1997) with GRU and contains information about the whole source sentence with a strong focus on the parts surrounding x_j. Here,

$$\overrightarrow{\mathbf{h}}_j = \mathbf{GRU}(x_j, \overrightarrow{\mathbf{h}}_{j-1}); \quad \overleftarrow{\mathbf{h}}_j = \mathbf{GRU}(x_j, \overleftarrow{\mathbf{h}}_{j+1}) \tag{4}$$

The weight $a_{t,j}$ is computed as

$$a_{t,j} = \frac{exp(e_{t,j})}{\sum_{k=1}^{N} exp(e_{t,k})} \tag{5}$$

where $e_{t,j} = \mathbf{v}_a^T tanh(\mathbf{W}_a \tilde{s}_{t-1} + \mathbf{U}_a \mathbf{h}_j)$ scores how much $\tilde{\mathbf{s}}_{t-1}$ attends to $\mathbf{h}_j$, where $\tilde{\mathbf{s}}_{t-1} = g(\mathbf{s}_{t-1}, y_{t-1})$ is an intermediate state tailored for computing the attention score.

Deep Transition RNN

Barone et al. (2017) first apply the deep transition RNN to NMT. As shown in Figure 1, in a deep transition RNN, the next state is computed by the sequential application of multiple transition layers at each time step, effectively using a feed-forward network embedded inside the recurrent cell. Obviously, this kind of architecture increases the depth of transition between the consecutive hidden states along the sequential axis, unlike the deep stacked RNN, in which transition between the consecutive hidden states is still shallow.

Although the deep transition RNN has been proven to be superior to deep stacked RNN on language modeling task (Pascanu et al. 2014), there is still a large margin between this deep transition NMT model (Barone et al. 2017) and the state-of-the-art NMT model, e.g. the Transformer (Vaswani et al. 2017), in terms of BLEU, which is also confirmed by Tang et al. (2018).

Model Description

In this section, we describe our novel Deep Transition RNN-based Architecture for NMT (DTMT). As shown in Figure 2, the DTMT consists of a bidirectional deep transition encoder and a deep transition decoder, connected by the multi-head attention (Vaswani et al. 2017). There are three deep transition modules: 1) *encoder transition* for encoding the source sentence into a sequence of distributed representations; 2) *query transition* for forming a query state to attend to the source representations; and 3) *decoder transition*

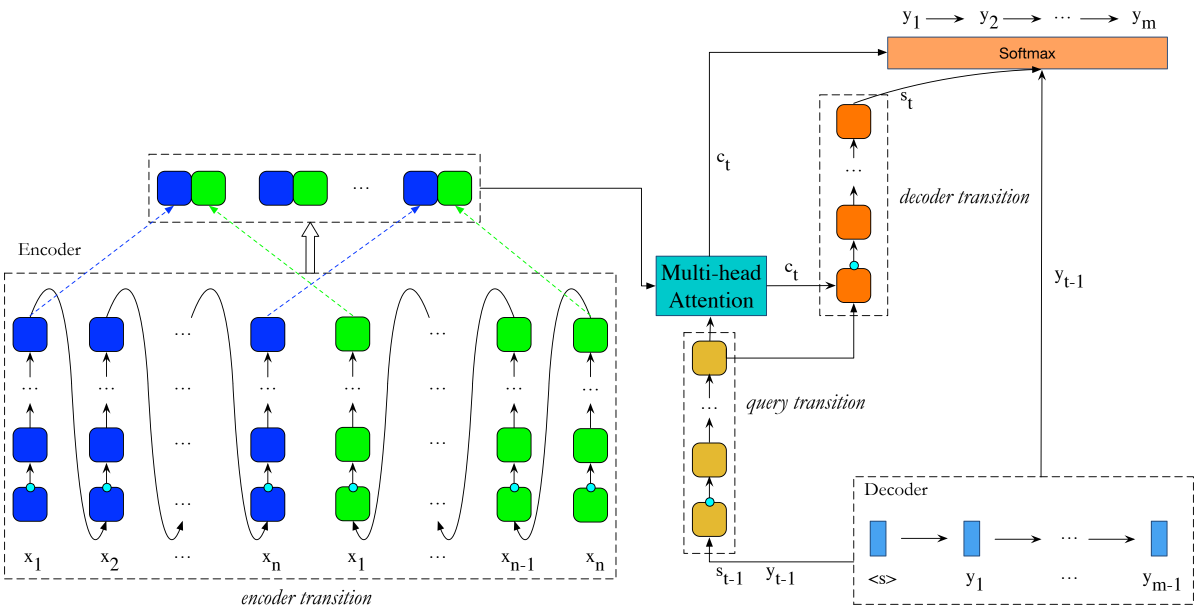

Figure 2: The architecture of DTMT. The bidirectional deep transition encoder (on the left) and the deep transition decoder (on the right) are connected by multi-head attention. There are three deep transition modules, namely the *encoder transition*, the *query transition* and the *decoder transition*, each of which consists of a L-GRU (the square frames fused with a small circle) at the bottom followed by several T-GRUs (the square frames) from bottom to up.

for generating the final decoder state of current time step. In each transition module, the transition block consists of a Linear Transformation enhanced GRU (L-GRU) at the bottom followed by several Transition GRUs (T-GRUs) from bottom to up. Before proceeding to the details of DTMT, we first describe the key components (i.e. GRU and its variants) of our deep transition modules.

Gated Recurrent Unit and its Variants

GRU: Gated Recurrent Unit (GRU) (Cho et al. 2014) is a variation of LSTM with fewer parameters. The activation function is armed with two specifically designed gates, named update gate and reset gate, to control the flow of information inside each unit. Each hidden state at time-step t is computed as follows:

$$\mathbf{h}_t = (1 - \mathbf{z}_t) \odot \mathbf{h}_{t-1} + \mathbf{z}_t \odot \widetilde{\mathbf{h}}_t \tag{6}$$

where $\odot$ is an element-wise product, $\mathbf{z}_t$ is the update gate, and $\widetilde{\mathbf{h}}_t$ is the candidate activation, computed as:

$$\widetilde{\mathbf{h}}_t = tanh(\mathbf{W}_{xh}\mathbf{x}_t + \mathbf{r}_t \odot (\mathbf{W}_{hh}\mathbf{h}_{t-1})) \tag{7}$$

where $\mathbf{x}_t$ is the input embedding, and $\mathbf{r}_t$ is the reset gate. Reset and update gates are computed as:

$$\mathbf{r}_t = \sigma(\mathbf{W}_{xr}\mathbf{x}_t + \mathbf{W}_{hr}\mathbf{h}_{t-1}) \tag{8}$$
$$\mathbf{z}_t = \sigma(\mathbf{W}_{xz}\mathbf{x}_t + \mathbf{W}_{hz}\mathbf{h}_{t-1}) \tag{9}$$

Actually, GRU can be viewed as a non-linear activation function with a specially designed gating mechanism, since the updated $\mathbf{h}_t$ has two sources controlled by the update gate and the reset gate: 1) the direct transfer from previous state $\mathbf{h}_{t-1}$; and 2) the candidate update $\widetilde{\mathbf{h}}_t$, which is a nonlinear transformation of the previous state $\mathbf{h}_{t-1}$ and the input embedding.

T-GRU: Transition GRU (T-GRU) is a key component of deep transition block. A basic deep transition block can be composed of a GRU followed by several T-GRUs from bottom to up at each time step, just as Figure 1. In the whole recurrent procedure, for the current time step, the "state" output of one GRU/T-GRU is used as the "state" input of the next T-GRU. And the "state" output of the last T-GRU for the current time step is carried over as the "state" input of the first GRU for the next time step. For a T-GRU, each hidden state at time-step t is computed as follows:

$$\mathbf{h}_t = (1 - \mathbf{z}_t) \odot \mathbf{h}_{t-1} + \mathbf{z}_t \odot \widetilde{\mathbf{h}}_t \tag{10}$$
$$\widetilde{\mathbf{h}}_t = tanh(\mathbf{r}_t \odot (\mathbf{W}_{hh}\mathbf{h}_{t-1})) \tag{11}$$

where reset gate $\mathbf{r}_t$ and update gate $\mathbf{z}_t$ are computed as:

$$\mathbf{r}_t = \sigma(\mathbf{W}_{hr}\mathbf{h}_{t-1}) \tag{12}$$
$$\mathbf{z}_t = \sigma(\mathbf{W}_{hz}\mathbf{h}_{t-1}) \tag{13}$$

As we can see, T-GRU is a special case of GRU with only "state" as input. It is also like the convolutional GRU (Kaiser and Sutskever 2015). Here the updated $\mathbf{h}_t$ has two sources controlled by the update gate and the reset gate: 1) the direct transfer from previous hidden state $\mathbf{h}_{t-1}$; and 2) the candidate update $\widetilde{\mathbf{h}}_t$, which is a nonlinear transformation of the previous hidden state $\mathbf{h}_{t-1}$. That is to say, T-GRU conducts both non-linear transformation and direct transfer of the input. This architecture will make training deep models easier.

L-GRU: L-GRU is a Linear Transformation enhanced GRU by incorporating an additional linear transformation of the input in its dynamics. Each hidden state at time-step t is computed as follows:

$$\mathbf{h}_t = (1 - \mathbf{z}_t) \odot \mathbf{h}_{t-1} + \mathbf{z}_t \odot \widetilde{\mathbf{h}}_t \tag{14}$$

where the candidate activation $\widetilde{\mathbf{h}}_t$ is computed as:

$$\widetilde{\mathbf{h}}_t = tanh(\mathbf{W}_{xh}\mathbf{x}_t + \mathbf{r}_t \odot (\mathbf{W}_{hh}\mathbf{h}_{t-1})) + \mathbf{l}_t \odot \mathbf{H}(\mathbf{x}_t) \tag{15}$$

where reset gate $\mathbf{r}_t$ and update gate $\mathbf{z}_t$ are computed as the formula (8) and (9), and $\mathbf{H}(\mathbf{x}_t) = \mathbf{W}_x\mathbf{x}_t$ is the linear transformation of the input $\mathbf{x}_t$, controlled by the linear transformation gate $\mathbf{l}_t$, which is computed as:

$$\mathbf{l}_t = \sigma(\mathbf{W}_{xl}\mathbf{x}_t + \mathbf{W}_{hl}\mathbf{h}_{t-1}) \tag{16}$$

In L-GRU, the updated $\mathbf{h}_t$ has three sources controlled by the update gate, the reset gate and the linear transformation gate: 1) the direct transfer from previous state $\mathbf{h}_{t-1}$; 2) the candidate update $\widetilde{\mathbf{h}}_t$; and 3) a direct contribution from the linear transformation of input $\mathbf{H}(\mathbf{x}_t)$. Compared with GRU, L-GRU conducts both non-linear transformation and linear transformation for the inputs, including the embedding input and the state input. Clearly, with L-GRU and T-GRUs, deep transition model can alleviate the problem of vanishing gradients since this structure provides a linear transformation path as a supplement between consecutive hidden states, which are originally connected by only non-linear transformations with multi-steps (e.g. GRU+T-GRUs).

Our L-GRU is inspired by the Linear Associative Unit (LAU) (Wang et al. 2017), while we exploit more concise operations with the same parameter quantity to the LAU to incorporate the linear transformation of the input x_t as well as preserving the original non-linear abstraction produced by the input and previous hidden state. Different from the LAU, 1) the linear transformation of input is controlled by both the update gate $\mathbf{z}_t$ and the linear transformation gate $\mathbf{l}_t$; and 2) the linear transformation gate $\mathbf{l}_t$ only focus on the linear transformation of the embedding input. These may be the main reasons why L-GRU is more effective than the LAU, as verified in our experiments described later.

DTMT

The formal description of the encoder and the decoder of DTMT is as follows:

Encoder: The encoder is a bidirectional deep transition encoder based on recurrent neural networks. Let L_s be the depth of *encoder transition*, then for the jth source word in the forward direction the forward source state $\overrightarrow{\mathbf{h}}_j \equiv \overrightarrow{\mathbf{h}}_{j,L_s}$ is computed as:

$$\begin{aligned}\overrightarrow{\mathbf{h}}_{j,0} &= \textbf{L-GRU}(\mathbf{x}_j, \overrightarrow{\mathbf{h}}_{j-1,L_s}) \\ \overrightarrow{\mathbf{h}}_{j,k} &= \textbf{T-GRU}_\mathbf{k}(\overrightarrow{\mathbf{h}}_{j,k-1}) \quad \text{for } 1 \le k \le L_s\end{aligned}$$

where the input to the first L-GRU is the word embedding $\mathbf{x}_j$, while the T-GRUs have only "state" as the input. Recurrence occurs as the previous state $\overrightarrow{\mathbf{h}}_{j-1,L_s}$ enters the computation in the first L-GRU transition for the current step. The reverse source word states are computed similarly and concatenated to the forward ones to form the bidirectional source annotations $C \equiv \{[\overrightarrow{\mathbf{h}}_{j,L_s}, \overleftarrow{\mathbf{h}}_{j,L_s}]\}$.

Decoder: As shown in Figure 2, the deep transition decoder consists of two transition modules, named *query transition* and *decoder transition*, of which *query transition* is conducted before the multi-head attention and *decoder transition* is conducted after the multi-head attention. These transition modules can be conducted to an arbitrary transition depth. Suppose the depth of *query transition* is L_q and the depth of *decoder transition* is L_d, then

$$\begin{aligned}\mathbf{s}_{t,0} &= \textbf{L-GRU}(\mathbf{y}_{t-1}, \mathbf{s}_{t-1,L_q+L_d+1}) \\ \mathbf{s}_{t,k} &= \textbf{T-GRU}(\mathbf{s}_{t,k-1}) \quad \text{for } 1 \le k \le L_q\end{aligned}$$

where $\mathbf{y}_{t-1}$ is the embedding of the previous target word. And then the context representation $\mathbf{c}_t$ of source sentence is computed by multi-head additive attention:

$$\mathbf{c}_t = Multihead\text{-}Attention(C, \mathbf{s}_{t,L_q})$$

after that, the *decoder transition* is computed as

$$\begin{aligned}\mathbf{s}_{t,L_q+1} &= \textbf{L-GRU}(\mathbf{c}_t, \mathbf{s}_{t,L_q}) \\ \mathbf{s}_{t,L_q+p} &= \textbf{T-GRU}(\mathbf{s}_{t,L_q+p-1}) \quad \text{for } 2 \le p \le L_d + 1\end{aligned}$$

The current state vector $\mathbf{s}_t \equiv \mathbf{s}_{t,L_q+L_d+1}$ is then used by the feed-forward output network as Formula (1) to predict the current target word.

Advanced Techniques

Except for the multi-head attention, we apply most recently advanced techniques during training to enhance our model:

- **Dropout:** We apply dropout on embedding layers, the output layer before prediction, and the candidate activation output (Semeniuta, Severyn, and Barth 2016) of RNN.
- **Label Smoothing:** We use uniform label smoothing with an uncertainty=0.1 (Szegedy et al. 2015), which has been proved to have a positive impact for the performance.
- **Layer Normalization:** Inspired by the Transformer, per-gate layer normalization (Ba, Kiros, and Hinton 2016) is applied within each gate (i.e. reset gate, update gate and linear transformation gate) of L-GRU/T-GRU. It is critical to stabilize the training process of deep transition model.
- **Positional Encoding:** We also add the positional encoding (Vaswani et al. 2017) to the input embeddings at the bottoms of the encoder and decoder to assist modeling positions of the tokens in the sequence. Although the positional encoding is originally designed for the non-recurrent (self-attentional) architecture, we find RNMT can also benefit from it. To stabilize the training process of our deep transition model, we add a scaling factor $1/\sqrt{d_k}$ (d_k is the dimension of embedding) to the original positional encoding function.

SYSTEM	ARCHITECTURE	# Para.	MT06	MT02	MT03	MT04	MT05	MT08	AVE.
Existing end-to-end NMT systems									
Shen et al. (2016)	GRU with MRT	–	37.34	40.36	40.93	41.37	38.81	29.23	38.14
Wang et al. (2017)	DeepLAU (4 layers)	–	37.29	–	39.35	41.15	38.07	–	–
Zhang et al. (2018)	Bi-directional decoding	–	38.38	–	40.02	42.32	38.84	–	–
Meng et al. (2018)	GRU with KV-Memory	–	39.08	40.67	38.40	41.10	38.73	30.87	37.95
Cheng et al. (2018)	AST (2 layers)	–	44.44	46.10	44.07	45.61	44.06	34.94	42.96
Vaswani et al. (2017)	Transformer (BIG)	277.6M	44.78	45.32	44.13	45.92	44.06	35.33	42.95
Our end-to-end NMT systems									
	SHALLOWRNMT	143.2M	42.99	44.24	42.96	44.97	42.69	33.00	41.57
this work	DTMT#1	170.5M	45.99	46.90	45.85	46.78	45.96	36.58	44.41
	DTMT#4	208.4M	**46.74**	**47.03**	**46.34**	**47.52**	**46.70**	**37.61**	**45.04**

Table 1: Case-insensitive BLEU scores (%) on NIST Chinese⇒English translation. Our deep transition model outperforms the state-of-the-art models including the Transformer (Vaswani et al. 2017) and the deep stacked RNMT (Cheng et al. 2018).

Experiments

Setup

We carry out experiments on Chinese⇒English (Zh⇒En), English⇒German (En⇒De) and English⇒French (En⇒Fr) translation tasks. For these tasks, we tokenize the references and evaluated the translation quality with BLEU scores (Papineni et al. 2002) as calculated by the *multi-bleu.pl* script.

For Zh⇒En, the training data consists of 1.25M sentence pairs extracted from the LDC corpora. We choose NIST 2006 (MT06) dataset as our valid set, and NIST 2002 (MT02), 2003 (MT03), 2004 (MT04), 2005 (MT05) and 2008 (MT08) datasets as our test sets. For En⇒De and En⇒Fr, we perform our experiments on the corpora provided by WMT14 that comprise 4.5M and 36M sentence pairs, respectively. We use newstest2013 as the valid set, and newstest2014 as the test set.

Training Details

In training the neural networks, we follow Sennrich, Haddow, and Birch (2016) to split words into sub-word units. For Zh⇒En, the number of merge operations in byte pair encoding (BPE) is set to 30K for both source and target languages. For En⇒De and En⇒Fr, we use a shared vocabulary generated by 32K BPEs following Chen et al. (2018).

The parameters are initialized uniformly between [-0.08, 0.08] and updated by SGD with the learning rate controlled by the Adam optimizer (Kingma and Ba 2014) ($\beta_1 = 0.9$, $\beta_2 = 0.999$, and $\epsilon = 1e^{-6}$). And we follow Chen et al. (2018) to vary the learning rate as follows:

$$lr = lr_0 \cdot min(1 + t \cdot (n-1)/np,\ n,\ n \cdot (2n)^{\frac{s-nt}{e-s}}) \quad (17)$$

Here, t is the current step, n is the number of concurrent model replicas in training, p is the number of warmup steps, s is the start step of the exponential decay, and e is the end step of the decay. For Zh⇒En, we use 2 M40 GPUs for synchronous training and set lr_0, p, s and e to 10^{-3}, 500, 8000, and 64000 respectively. For En⇒De, we use 8 M40 GPUs and set lr_0, p, s and e to 10^{-4}, 50, 200000, and 1200000 respectively. For En⇒Fr, we use 8 M40 GPUs and set lr_0, p, s and e to 10^{-4}, 50, 400000, and 3000000 respectively.

We limit the length of sentences to 128 sub-words for Zh⇒En and 256 sub-words for En⇒De and En⇒Fr in the training stage. We batch sentence pairs according to the approximate length, and limit input and output tokens to 4096 per GPU. We apply dropout strategy to avoid overfitting (Hinton et al. 2012). In particular, for Zh⇒En, we set dropout rates of the embedding layers, the layer before prediction and the RNN output layer to 0.5, 0.5 and 0.3 respectively. For En⇒De, we set these dropout rates to 0.3, 0.3 and 0.1 respectively. For En⇒Fr, we set these dropout rates to 0.2, 0.2 and 0.1 respectively. For each model of the translation tasks, the dimension of word embeddings and hidden layer is 1024. Translations are generated by beam search and log-likelihood scores are normalized by the sentence length. We set $beam\_size = 4$ and length penalty $alpha = 0.6$. We monitor the training process every 2K iterations and decide the early stop condition by validation BLEU.

System Description

- SHALLOWRNMT: a shallow yet strong RNMT baseline system, which is our in-house implementation of the attention-based RNMT (Bahdanau, Cho, and Bengio 2015) augmented by combining advanced techniques, including multi-head attention, layer normalization, label smoothing, dropouts on multi-layers (the embedding layers, the output layer before prediction and the candidate activation output of each GRU) and the positional encoding.
- DTMT#NUM: our deep transition systems, and the #NUM stands for the transition depth (i.e. the number of T-GRUs) above the bottom L-GRU in each transition module (i.e. *encoder transition*, *query transition*, and *decoder transition*). For example, DTMT#2 means each transition module contains 1 L-GRU layer and 2 T-GRU layers, namely 3 layers in the encoder and 6 layers in the decoder.

Results on NIST Chinese⇒English

Table 1 shows the results on NIST Zh⇒En translation task. Our baseline system SHALLOWRNMT significantly outperforms previous RNN-based NMT systems on the same datasets. Shen et al. (2016) propose minimum risk training (MRT) to optimize the model with respect to BLEU scores. Wang et al. (2017) propose the linear associative units

SYSTEM	ARCHITECTURE	EN-DE	EN-FR
Zhou et al. (2016)	LSTM (8 layers)	20.60	37.70
Luong, Pham, and Manning (2015)	LSTM (4 layers)	20.90	31.50
Wang et al. (2017)	DeepLAU (4 layers)	23.80	35.10
Wu et al. (2016)	GNMT (8 layers)	24.60	38.95
Gehring et al. (2017)	ConvS2S (15 layers)	25.16	40.46
Cheng et al. (2018)	AST (2 layers)	25.26	–
Vaswani et al. (2017)	Transformer (BIG)	28.40	41.00
Chen et al. (2018)	RNMT+ (8 layers)	28.49	41.00
this work	SHALLOWRNMT	25.66	39.28
	DTMT#1	27.92	40.75
	DTMT#4	**28.70**	**42.02**

Table 2: Case-sensitive BLEU scores (%) on WMT 14 English⇒German and English⇒French translation. DTMT#4 outperforms the state-of-the-art models including the Transformer (Vaswani et al. 2017) and the RNMT+ (Chen et al. 2018).

(LAU) to address the issue of gradient diffusion, and their system is a deep model with 4 layers. Zhang et al. (2018) propose to exploit both left-to-right and right-to-left decoding strategies to capture bidirectional dependencies. Meng et al. (2018) propose key-value memory augmented attention to improve the adequacy of translation. Compared with them, our baseline system SHALLOWRNMT outperforms their best models by more than 3 BLEU points. SHALLOWRNMT is only 1.4 BLEU points lower than the state-of-the-art deep models, i.e. the Transformer (with 6 attention layers) (Vaswani et al. 2017) and the deep stacked RNMT augmented with adversarial stability training (AST) (Cheng et al. 2018). We build this strong baseline system to show that the shallow RNMT model with advanced techniques is indeed powerful. And we hope that the strong baseline system used in this work makes the evaluation convincing.

Our deep transition model DTMT#1 with only one transition layer can further bring in up to +3.58 BLEU points (+2.84 BLEU on average) improvements over the strong baseline SHALLOWRNMT. DTMT#1 outperforms all the previous systems list in Table 1, and achieves about +1.45 BLEU points improvements over the best model. With a deeper transition architecture, DTMT#4 achieves the best results, which is up to +4.61 BLEU points (+3.47 BLEU on average) higher than the SHALLOWRNMT. Compared with the Transformer and deep RNMT augmented with AST, DTMT#4 yields a gain of +2.08 BLEU on average and achieves the best performance ever reported on this dataset.

Results on WMT14 En⇒De and En⇒Fr

To demonstrate that our models work well across different language pairs, we also evaluate our models on the WMT14 benchmarks on En⇒De and En⇒Fr translation tasks, as listed in Table 2. For comparison, we list existing NMT systems which are trained on the same WMT 14 corpora. Among these systems, the Transformer (Vaswani et al. 2017) represents the best non-recurrent model, and the RNMT+ (Chen et al. 2018) represents the best RNMT model. Our baseline system SHALLOWRNMT can achieve better performance than most RNMT systems except for the RNMT+, which demonstrates that SHALLOWRNMT is also a strong baseline system for both En⇒De and En⇒Fr.

Our deep transition model DTMT#1 with only one transition layer can bring in +2.26 BLEU for En⇒De and +1.47 BLEU for En⇒Fr over the strong baseline SHALLOWRNMT. With a deeper transition architecture, our DTMT#4 achieves +3.04 BLEU for En⇒De and +2.74 BLEU for En⇒Fr over the baseline, and outperforms state-of-the-art systems, i.e. the Transformer and the RNMT+.

ARCHITECTURE	RNN	# Para.	BLEU
SHALLOWRNMT	GRU	143.2M	41.57
	LAU	157.9M	43.06
	L-GRU	157.9M	43.83
DTMT#1	*GRU+T-GRU*	155.8M	43.63
	LAU+T-GRU	170.5M	43.79
	L-GRU+T-GRU	170.5M	44.41
DTMT#4	*GRU+T-GRUs*	193.7M	44.16
	LAU+T-GRUs	208.4M	44.54
	L-GRU+T-GRUS	208.4M	**45.04**

Table 3: Comparisons of GRU, LAU and L-GRU with different architectures on NIST Chinese⇒English translation (average BLEU scores (%) on test sets). The italics in the "RNN" column indicate the potential variants of the corresponding architecture. For example, the RNN units in DTMT#1 can be replaced with the *LAU+T-GRU*.

Analysis

L-GRU vs. GRU & LAU: We investigate the effectiveness of the proposed L-GRU on different architectures, including the SHALLOWRNMT and the DTMTs. From Table 3 we can see that the L-GRU is effective since it can consistently bring in substantial improvements over different architectures. In particular, it brings in +2.26 BLEU points improvements over SHALLOWRNMT averagely on five test sets, and it also leads to +0.78 $\sim$ +0.88 BLEU points improvements over our deep transition architectures (i.e. DTMT#1 and DTMT#4). Additionally, with more concise operations and the same parameter quantity, L-GRU can further outperform the LAU (Wang et al. 2017) on different strong systems by +0.5 $\sim$ +0.77 BLEU points. These results demonstrate that L-GRU is a more effective unit for both deep transition models and the shallow RNMT model.

Ablation Study: Our deep transition model consists of three deep transition modules, including the *encoder transition*, the *query transition* and the *decoder transition*. We perform an ablation study on Zh⇒En translation to investigate the effectiveness of these transition modules by choosing DTMT#4 as an example. As shown in Table 4, replacing any one with its corresponding part of the shallow RNMT (Bahdanau, Cho, and Bengio 2015) leads to the translation performance decrease (-0.68 $\sim$ -1.74 BLEU). Among these transition modules, the *encoder transition* is the most important, since deleting it leads to the most obvious decline (-1.74 BLEU). We also conduct an ablation study of the L-GRU and the "Advanced Techniques" on En⇒De task. As shown in Table 5, deleting the L-GRU and/or the "Advanced Techniques" leads to sharp declines on translation quality. Therefore, we can conclude that both

enc-transion	*query-transion*	*dec-transion*	BLEU
√	√	√	**45.04**
×	√	√	43.30
√	×	√	44.35
√	√	×	44.36

Table 4: Ablation study of deep transition modules on NIST Chinese⇒English translation (average BLEU scores (%) on test sets). Here "×" stands for replacing the transition module with the corresponding part of the conventional shallow RNMT (Bahdanau, Cho, and Bengio 2015).

ARCHITECTURE	BLEU
DTMT#4	**28.70**
−L-GRU	27.81
−L-GRU & Advanced Techniques	26.27

Table 5: Ablation study of the L-GRU and the "Advanced Techniques" on WMT14 English⇒German translation.

the L-GRU and the "Advanced Techniques" are key components for DTMT#4 to achieve the state-of-the-art performance.

Transition Depth & Positional Encoding: Table 6 shows the impact of the transition depth and the positional encoding on Zh⇒En translation. From these results, we can draw the following conclusions: 1) with the increasing of transition depth (rows 3-6), our model can consistently achieve better performance; 2) T-GRUs do bring in significant improvements even over the strong baseline (rows 2-3); and 3) on different architectures, we can see that the positional encoding can further bring in consistent improvements (+0.1 ∼ +0.3 BLEU) over its counterpart without the positional encoding. This demonstrates that, although the positional encoding is originally designed for non-recurrent network (i.e. Transformer), it also can assist the training of RNN-based models by modeling positions of tokens in the sequence.

About Length: A more detailed comparison between DTMT#4, DTMT#1, SHALLOWRNMT and the Transformer suggest that our deep transition architectures are essential to achieve the superior performance. Figure 3 shows the BLEU scores of generated translations on the test sets with respect to the lengths of the source sentences. In particular, we test the BLEU scores on sentences longer than $\{0, 10, 20, 30, 40, 50, 60\}$ in the merged test set of MT02, MT03, MT04, MT05 and MT08. Clearly, on sentences with different lengths, DTMT#4 and DTMT#1 always yield higher BLEU scores than SHALLOWRNMT and the Transformer consistently. And DTMT#4 yields the best BLEU scores on sentences with different lengths.

Related Work

Our work is inspired by the deep transition RNN (Pascanu et al. 2014), which is applied on language modeling task. Barone et al. (2017) fist apply this kind of architecture on NMT, while there is still a large margin between this transition model and the state-of-the-art NMT models. Different from these works, we extremely enhance the deep transition architecture and build the state-of-the-art deep transition NMT model from three aspects: 1) fusing L-GRU and T-GRUs, to provide a linear transformation path between consecutive hidden states, as well as preserving the non-linear transformation path; 2) exploiting three deep transition modules, including the *encoder transition*, the *query transition* and the *decoder transition*; and 3) investigating and combing recent advanced techniques, including multi-head attention, labeling smoothing, layer normalization, dropout on multi-layers and positional encoding.

Our work is also inspired by deep stacked RNN models for NMT (Zhou et al. 2016; Wang et al. 2017; Chen et al. 2018). Zhou et al. (2016) propose fast-forward connections to address the notorious problem of vanishing/exploding gradients for deep stacked RNMT. Wang et al. (2017) propose the Linear Associative Unit (LAU) to reduce the gradient path inside the recurrent units. Different from these studies, we focus on the deep transition architecture and propose a novel linear transformation enhanced GRU (L-GRU) for our deep transition RNMT. L-GRU is verified more effective than the LAU, although L-GRU exploits more concise operations with the same parameter quantity to incorporate the linear transformation of the embedding input. Inspired

#	ARCHITECTURE	NON-PE	PE
1	SHALLOWRNMT	41.22	41.57
2	+ L-GRU	43.54	43.83
3	DTMT#1	44.28	44.41
4	DTMT#2	44.50	44.66
5	DTMT#3	44.61	44.70
6	DTMT#4	**44.72**	**45.04**

Table 6: Impact of transition depth and the positional encoding (PE) on NIST Chinese⇒English translation (average BLEU scores (%) on test sets).

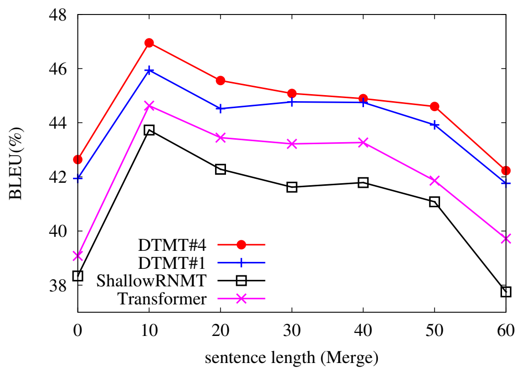

Figure 3: The BLEU scores (%) of generated translations on the merged four test sets with respect to the lengths of source sentences. The numbers on X-axis of the figure stand for sentences *longer than* the corresponding length, e.g., 40 for source sentences with > 40 words.

by RNMT+ (Chen et al. 2018), we investigate and combine generally applicable training and optimization techniques, and finally enable our DTMT to achieve superior quality to state-of-the-art NMT systems.

Conclusion

We propose a novel and effective deep transition architecture for NMT. Our empirical study on Chinese⇒English, English⇒German and English⇒French translation tasks shows that our DTMT can achieve remarkable improvements on translation quality. Experimental results on NIST Chinese⇒English translation show that DTMT can outperform the Transformer by +2.09 BLEU points even with fewer parameters and achieve the best results ever reported on the same dataset. On WMT14 English⇒German and English⇒French tasks, it shows superior quality to the state-of-the-art NMT systems (Vaswani et al. 2017; Chen et al. 2018).

References

Ba, J. L.; Kiros, J. R.; and Hinton, G. E. 2016. Layer normalization. *arXiv preprint arXiv:1607.06450*.

Bahdanau, D.; Cho, K.; and Bengio, Y. 2015. Neural machine translation by jointly learning to align and translate. In *ICLR*.

Barone, A. V. M.; Helcl, J.; Sennrich, R.; Haddow, B.; and Birch, A. 2017. Deep architectures for neural machine translation. *arXiv preprint arXiv:1707.07631*.

Chen, M. X.; Firat, O.; Bapna, A.; Johnson, M.; Macherey, W.; Foster, G.; Jones, L.; Schuster, M.; Shazeer, N.; Parmar, N.; Vaswani, A.; Uszkoreit, J.; Kaiser, L.; Chen, Z.; Wu, Y.; and Hughes, M. 2018. The best of both worlds: Combining recent advances in neural machine translation. In *ACL*.

Cheng, Y.; Tu, Z.; Meng, F.; Zhai, J.; and Liu, Y. 2018. Towards robust neural machine translation. In *ACL*.

Cho, K.; van Merrienboer, B.; Gulcehre, C.; Bougares, F.; Schwenk, H.; and Bengio, Y. 2014. Learning phrase representations using rnn encoder-decoder for statistical machine translation. In *EMNLP*.

Gehring, J.; Auli, M.; Grangier, D.; Yarats, D.; and Dauphin, Y. N. 2017. Convolutional sequence to sequence learning. In *ICML*.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep residual learning for image recognition. In *CVPR*.

Hinton, G. E.; Srivastava, N.; Krizhevsky, A.; Sutskever, I.; and Salakhutdinov, R. R. 2012. Improving neural networks by preventing co-adaptation of feature detectors. *arXiv*.

Kaiser, L., and Sutskever, I. 2015. Neural gpus learn algorithms. *CoRR* abs/1511.08228.

Kalchbrenner, N.; Espeholt, L.; Simonyan, K.; Oord, A. v. d.; Graves, A.; and Kavukcuoglu, K. 2017. Neural machine translation in linear time. In *ICML*.

Kingma, D. P., and Ba, J. 2014. Adam: A method for stochastic optimization. *CoRR* abs/1412.6980.

Luong, M.-T.; Pham, H.; and Manning, C. D. 2015. Effective approaches to attention-based neural machine translation. In *EMNLP*.

Meng, F.; Tu, Z.; Cheng, Y.; Wu, H.; Zhai, J.; Yang, Y.; and Wang, D. 2018. Neural machine translation with key-value memory-augmented attention. In *IJCAI*.

Papineni, K.; Roukos, S.; Ward, T.; and Zhu, W.-J. 2002. Bleu: a method for automatic evaluation of machine translation. In *ACL*.

Pascanu, R.; Gulcehre, C.; Cho, K.; and Bengio, Y. 2014. How to construct deep recurrent neural networks. In *ICLR*.

Schuster, M., and Paliwal, K. K. 1997. Bidirectional recurrent neural networks. *TSP*.

Semeniuta, S.; Severyn, A.; and Barth, E. 2016. Recurrent dropout without memory loss. *CoRR* abs/1603.05118.

Sennrich, R.; Haddow, B.; and Birch, A. 2016. Neural machine translation of rare words with subword units. In *ACL*.

Shen, S.; Cheng, Y.; He, Z.; He, W.; Wu, H.; Sun, M.; and Liu, Y. 2016. Minimum risk training for neural machine translation. In *ACL*.

Sutskever, I.; Vinyals, O.; and Le, Q. V. 2014. Sequence to sequence learning with neural networks. In *NIPS*.

Szegedy, C.; Vanhoucke, V.; Ioffe, S.; Shlens, J.; and Wojna, Z. 2015. Rethinking the inception architecture for computer vision. *CoRR* abs/1512.00567.

Tang, G.; Müller, M.; Rios, A.; and Sennrich, R. 2018. Why Self-Attention? A Targeted Evaluation of Neural Machine Translation Architectures. In *EMNLP*.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, L.; and Polosukhin, I. 2017. Attention is all you need. In *NIPS*.

Wang, M.; Lu, Z.; Zhou, J.; and Liu, Q. 2017. Deep neural machine translation with linear associative unit. In *ACL*.

Wu, Y.; Schuster, M.; Chen, Z.; Le, Q. V.; Norouzi, M.; Macherey, W.; Krikun, M.; et al. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*.

Zhang, X.; Su, J.; Qin, Y.; Liu, Y.; Ji, R.; and Wang, H. 2018. Asynchronous bidirectional decoding for neural machine translation. In *AAAI*.

Zhou, J.; Cao, Y.; Wang, X.; Li, P.; and Xu, W. 2016. Deep recurrent models with fast-forward connections for neural machine translation. *TACL*.