Deep Neural Machine Translation with Linear Associative Unit

Mingxuan Wang[1] **Zhengdong Lu**[2] **Jie Zhou**[2] **Qun Liu**[4,5]
[1]Mobile Internet Group, Tencent Technology Co., Ltd
wangmingxuan@ict.ac.cn
[2]DeeplyCurious.ai
[3] Insititute of Deep Learning Research, Baidu Co., Ltd
[4] Institute of Computing Technology, Chinese Academy of Sciences
[5]ADAPT Centre, School of Computing, Dublin City University

Abstract

Deep Neural Networks (DNNs) have provably enhanced the state-of-the-art Neural Machine Translation (NMT) with their capability in modeling complex functions and capturing complex linguistic structures. However NMT systems with deep architecture in their encoder or decoder RNNs often suffer from severe gradient diffusion due to the non-linear recurrent activations, which often make the optimization much more difficult. To address this problem we propose novel linear associative units (LAU) to reduce the gradient propagation length inside the recurrent unit. Different from conventional approaches (LSTM unit and GRU), LAUs utilizes linear associative connections between input and output of the recurrent unit, which allows unimpeded information flow through both space and time direction. The model is quite simple, but it is surprisingly effective. Our empirical study on Chinese-English translation shows that our model with proper configuration can improve by 11.7 BLEU upon Groundhog and the best reported results in the same setting. On WMT14 English-German task and a larger WMT14 English-French task, our model achieves comparable results with the state-of-the-art.

1 Introduction

Neural Machine Translation (NMT) is an end-to-end learning approach to machine translation which has recently shown promising results on multiple language pairs (Luong et al., 2015; Shen et al., 2015; Wu et al., 2016; Zhang et al., 2016; Tu et al., 2016; Zhang and Zong, 2016; Jean et al., 2015; Meng et al., 2015). Unlike conventional Statistical Machine Translation (SMT) systems (Koehn et al., 2003; Chiang, 2005; Liu et al., 2006; Xiong et al., 2006; Mi et al., 2008) which consist of multiple separately tuned components, NMT aims at building upon a single and large neural network to directly map input text to associated output text. Typical NMT models consists of two recurrent neural networks (RNNs), an encoder to read and encode the input text into a distributed representation and a decoder to generate translated text conditioned on the input representation (Sutskever et al., 2014; Bahdanau et al., 2014).

Driven by the breakthrough achieved in computer vision (He et al., 2015; Srivastava et al., 2015), research in NMT has recently turned towards studying Deep Neural Networks (DNNs). Wu et al. (2016) and Zhou et al. (2016) found that deep architectures in both the encoder and decoder are essential for capturing subtle irregularities in the source and target languages. However, training a deep neural network is not as simple as stacking layers. Optimization often becomes increasingly difficult with more layers. One reasonable explanation is the notorious problem of vanishing/exploding gradients which was first studied in the context of vanilla RNNs (Pascanu et al., 2013b). Most prevalent approaches to solve this problem rely on short-cut connections between adjacent layers such as residual or fast-forward connections (He et al., 2015; Srivastava et al., 2015; Zhou et al., 2016). Differ-

Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, pages 136–145
Vancouver, Canada, July 30 - August 4, 2017. ©2017 Association for Computational Linguistics
https://doi.org/10.18653/v1/P17-1013

ent from previous work, we choose to reduce the gradient path inside the recurrent units and propose a novel Linear Associative Unit (LAU) which creates a fusion of both linear and non-linear transformations of the input. Through this design, information can flow across several steps both in time and in space with little attenuation. The mechanism makes it easy to train deep stack RNNs which can efficiently capture the complex inherent structures of sentences for NMT. Based on LAUs, we also propose a NMT model , called DEEPLAU, with deep architecture in both the encoder and decoder.

Although DEEPLAU is fairly simple, it gives remarkable empirical results. On the NIST Chinese-English task, DEEPLAU with proper settings yields the best reported result and also a 4.9 BLEU improvement over a strong NMT baseline with most known techniques (e.g, dropout) incorporated. On WMT English-German and English-French tasks, it also achieves performance superior or comparable to the state-of-the-art.

2 Neural machine translation

A typical neural machine translation system is a single and large neural network which directly models the conditional probability $p(\mathbf{y}|\mathbf{x})$ of translating a source sentence $\mathbf{x} = \{x_1, x_2, \cdots, x_{T_x}\}$ to a target sentence $\mathbf{y} = \{y_1, y_2, \cdots, y_{T_y}\}$.

Attention-based NMT, with RNNsearch as its most popular representative, generalizes the conventional notion of encoder-decoder in using an array of vectors to represent the source sentence and dynamically addressing the relevant segments of them during decoding. The process can be explicitly split into an encoding part, a decoding part and an attention mechanism. The model first encodes the source sentence $\mathbf{x}$ into a sequence of vectors $\mathbf{c} = \{h_1, h_2, \cdots, h_{T_x}\}$. In general, h_i is the annotation of x_i from a bi-directional RNN which contains information about the whole sentence with a strong focus on the parts of x_i. Then, the RNNsearch model decodes and generates the target translation $\mathbf{y}$ based on the context $\mathbf{c}$ and the partial traslated sequence $\mathbf{y}_{<t}$ by maximizing the probability of $p(y_i|y_{<i}, \mathbf{c})$. In the attention model, $\mathbf{c}$ is dynamically obtained according to the contribution of the source annotation made to the word prediction. This is called automatic alignment (Bahdanau et al., 2014) or attention mechanism (Luong et al., 2015), but it is essentially reading with content-based addressing defined in (Graves et al., 2014). With this addressing strategy the decoder can attend to the source representation that is most relevant to the stage of decoding.

Deep neural models have recently achieved a great success in a wide range of problems. In computer vision, models with more than 100 convolutional layers have outperformed shallow ones by a big margin on a series of image tasks (He et al., 2015; Srivastava et al., 2015). Following similar ideas of building deep CNNs, some promising improvements have also been achieved on building deep NMT systems. Zhou et al. (2016) proposed a new type of linear connections between adjacent layers to simplify the training of deeply stacked RNNs. Similarly, Wu et al. (2016) introduced residual connections to their deep neural machine translation system and achieve great improvements. However the optimization of deep RNNs is still an open problem due to the massive recurrent computation which makes the gradient propagation path extremely tortuous.

3 Model Description

In this section, we discuss Linear Associative Unit (LAU) to ease the training of deep stack of RNNs. Based on this idea, we further propose DEEPLAU, a neural machine translation model with a deep encoder and decoder.

3.1 Recurrent Layers

A recurrent neural network (Williams and Zipser, 1989) is a class of neural network that has recurrent connections and a state (or its more sophisticated memory-like extension). The past information is built up through the recurrent connections. This makes RNN applicable for sequential prediction tasks of arbitrary length. Given a sequence of vectors $\mathbf{x} = \{\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_T\}$ as input, a standard RNN computes the sequence hidden states $\mathbf{h} = \{\mathbf{h}_1, \mathbf{h}_2, \cdots, \mathbf{h}_T\}$ by iterating the following equation from $t = 1$ to $t = T$:

$$\mathbf{h}_t = \phi(\mathbf{x}_t, \mathbf{h}_{t-1}) \tag{1}$$

ϕ is usually a nonlinear function such as composition of a logistic sigmoid with an affine transformation.

3.2 Gated Recurrent Unit

It is difficult to train RNNs to capture long-term dependencies because the gradients tend to either vanish (most of the time) or explode. The effect of long-term dependencies is dropped exponentially with respect to the gradient propagation length. The problem was explored in depth by (Hochreiter and Schmidhuber, 1997; Pascanu et al., 2013b). A successful approach is to design a more sophisticated activation function than a usual activation function consisting of gating functions to control the information flow and reduce the propagation path. There is a long thread of work aiming to solve this problem, with the long short-term memory units (LSTM) being the most salient examples and gated recurrent unit (GRU) being the most recent one (Hochreiter and Schmidhuber, 1997; Cho et al., 2014). RNNs employing either of these recurrent units have been shown to perform well in tasks that require capturing long-term dependencies.

GRU can be viewed as a slightly more dramatic variation on LSTM with fewer parameters. The activation function is armed with two specifically designed gates called update and reset gates to control the flow of information inside each hidden unit. Each hidden state at time-step t is computed as follows

$$\mathbf{h}_t = (1 - \mathbf{z}_t) \odot \mathbf{h}_{t-1} + \mathbf{z}_t \odot \tilde{\mathbf{h}}_t \tag{2}$$

where $\odot$ is an element-wise product, $\mathbf{z}_t$ is the update gate, and $\tilde{\mathbf{h}}_t$ is the candidate activation.

$$\tilde{\mathbf{h}}_t = \tanh(\mathbf{W}_{xh}\mathbf{x}_t + \mathbf{W}_{hh}(\mathbf{r}_t \odot \mathbf{h}_{t-1})) \tag{3}$$

where $\mathbf{r}_t$ is the reset gate. Both reset and update gates are computed as :

$$\mathbf{r}_t = \sigma(\mathbf{W}_{xr}\mathbf{x}_t + \mathbf{W}_{hr}\mathbf{h}_{t-1}) \tag{4}$$

$$\mathbf{z}_t = \sigma(\mathbf{W}_{xz}\mathbf{x}_t + \mathbf{W}_{hz}\mathbf{h}_{t-1}) \tag{5}$$

This procedure of taking a linear sum between the existing state and the newly computed state is similar to the LSTM unit.

3.3 Linear Associative Unit

GRU can actually be viewed as a non-linear activation function with gating mechanism. Here we propose LAU which extends GRU by having an additional linear transformation of the input in its dynamics. More formally, the state update function becomes

$$\begin{aligned}\mathbf{h}_t =&((1 - \mathbf{z}_t) \odot \mathbf{h}_{t-1} + \mathbf{z}_t \odot \tilde{\mathbf{h}}_t) \odot (1 - \mathbf{g}_t)\\ &+ \mathbf{g}_t \odot \mathbf{H}(\mathbf{x}_t).\end{aligned} \tag{6}$$

Here the updated $\mathbf{h}_t$ has three sources: 1) the direct transfer from previous state $\mathbf{h}_{t-1}$, 2) the candidate update $\tilde{\mathbf{h}}_t$, and 3) a direct contribution from the input $\mathbf{H}(\mathbf{x}_t)$. More specifically, $\tilde{\mathbf{h}}_t$ contains the nonlinear information of the input and the previous hidden state.

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

where $\mathbf{f}_t$ and $\mathbf{r}_t$ express how much of the nonlinear abstraction are produced by the input $\mathbf{x}_t$ and previous hidden state $\mathbf{h}_t$, respectively. For simplicity, we set $\mathbf{f}_t = 1 - \mathbf{r}_t$ in this paper and find that this works well in our experiments. The term $\mathbf{H}(\mathbf{x}_t)$ is usually an affine linear transformation of the input $\mathbf{x}_t$ to mach the dimensions of $\mathbf{h}_t$, where $\mathbf{H}(\mathbf{x}_t) = \mathbf{W}_x x_t$. The associated term $\mathbf{g}_t$ (the input gate) decides how much of the linear transformation of the input is carried to the hidden state and then the output. The gating function $\mathbf{r}_t$ (reset gate) and $\mathbf{z}_t$ (update gate) are computed following Equation (4) and (5) while $\mathbf{g}_t$ is computed as

$$\mathbf{g}_t = \sigma(\mathbf{W}_{xg}\mathbf{x}_t + \mathbf{W}_{hg}\mathbf{h}_{t-1}). \tag{8}$$

The term $\mathbf{g}_t \odot \mathbf{H}(\mathbf{x}_t)$ therefore offers a direct way for input $\mathbf{x}_t$ to go to later hidden layers, which can eventually lead to a path to the output layer when applied recursively. This mechanism is potentially very useful for translation where the input, no matter whether it is the source word or the attentive reading (context), should sometimes be directly carried to the next stage of processing without any substantial composition or nonlinear transformation. To understand this, imagine we want to translate a English sentence containing a relative rare entity name such as “Bahrain” to Chinese: LAU is potentially able to retain the embedding of this word in its hidden state, which will otherwise be prone to serious distortion due to the scarcity of training instances for it.

3.4 DEEPLAU

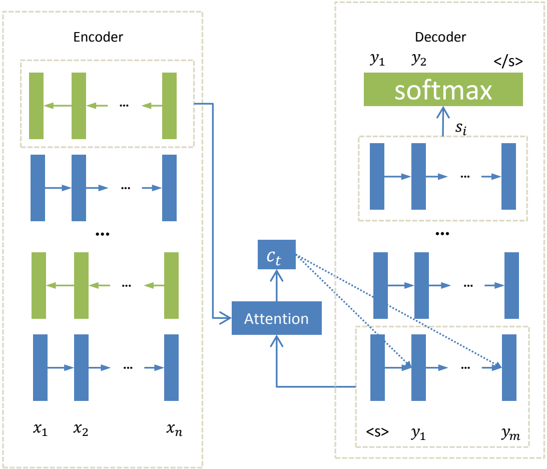

Figure 1: DEEPLAU: a neural machine translation model with deep encoder and decoder.

Graves et al. (2013) explored the advantages of deep RNNs for handwriting recognition and text generation. There are multiple ways of combining one layer of RNN with another. Pascanu et al. (2013a) introduced Deep Transition RNNs with Skip connections (DT(S)-RNNs). Kalchbrenner et al. (2015) proposed to make a full connection of all the RNN hidden layers. In this work we employ vertical stacking where only the output of the previous layer of RNN is fed to the current layer as input. The input at recurrent layer ℓ (denoted as $\mathbf{x}_t^{\ell}$) is exactly the output of the same time step at layer $\ell - 1$ (denoted as $\mathbf{h}_t^{\ell-1}$). Additionally, in order to learn more temporal dependencies, the sequences can be processed in different directions. More formally, given an input sequence $\mathbf{x} = (\mathbf{x}_1, ..., \mathbf{x}_T)$, the output on layer ℓ is

$$\mathbf{h}_t^{(\ell)} = \begin{cases} \mathbf{x}_t, & \ell = 1 \\ \phi^{\ell}(\mathbf{h}_{t+d}^{(\ell)}, \mathbf{h}_t^{(\ell-1)}), & \ell > 1 \end{cases} \quad (9)$$

where

- $\mathbf{h}_t^{(\ell)}$ gives the output of layer ℓ at location t.
- ϕ is a recurrent function and we choose LAUs in this work.
- The directions are marked by a direction term $d \in \{-1, 1\}$. If we fixed d to -1, the input will be processed in forward direction, otherwise backward direction.

The deep architecture of DEEPLAU, as shown in Figure 1, consists of three parts: a stacked LAU-based encoder, a stacked LAU-based decoder and an improved attention model.

Encoder One shortcoming of conventional RNNs is that they are only able to make use of previous context. In machine translation, where whole source utterances are transcribed at once, there is no reason not to exploit future context as well. Thus bi-directional RNNs are proposed to integrate information from the past and the future. The typical bidirectional approach processes the raw input in backward and forward direction with two separate layers, and then concatenates them together. Following Zhou et al. (2016), we choose another bidirectional approach to process the sequence in order to learn more temporal dependencies in this work. Specifically, an RNN layer processes the input sequence in forward direction. The output of this layer is taken by an upper RNN layer as input, processed in reverse direction. Formally, following Equation (9), we set $d = (-1)^{\ell}$. This approach can easily build a deeper network with the same number of parameters compared to the classical approach. The final encoder consists of L_{enc} layers and produces the output $\mathbf{h}^{L_{\text{enc}}}$ to compute the conditional input $\mathbf{c}$ to the decoder.

Attention Model The alignment model $\alpha_{t,j}$ scores how well the output at position t matches the inputs around position j based on $\mathbf{s}_{t-1}^1$ and $\mathbf{h}_j^{L_{\text{enc}}}$ where $\mathbf{h}_j^{L_{\text{enc}}}$ is the top-most layer of the encoder at step j and $\mathbf{s}_{t-1}^1$ is the first layer of decoder at step $t - 1$. It is intuitively beneficial to exploit the information of y_{t-1} when reading from the source sentence representation, which is missing from the implementation of attention-based NMT in (Bahdanau et al., 2014). In this work, we build a more effective alignment path by feeding both the previous hidden state $\mathbf{s}_{t-1}^1$ and the context word y_{t-1} to the attention model, inspired by the recent implementation of attention-based

NMT[1]. The conditional input $\mathbf{c}_j$ is a weighted sum of attention score $\alpha_{t,j}$ and encoder output $\mathbf{h}^{L_{\text{enc}}}$. Formally, the calculation of $\mathbf{c}_j$ is

$$\mathbf{c}_j = \sum_{t=1}^{t=L_x} \alpha_{t,j} \mathbf{h}_t^{L_{\text{enc}}} \tag{10}$$

where

$$\begin{aligned} e_{t,j} &= \mathbf{v}_a^T \sigma(\mathbf{W}_a \mathbf{s}_{t-1}^1 + \mathbf{U}_a \mathbf{h}_j^{L_{\text{enc}}} + \mathbf{W}_y \mathbf{y}_{t-1}) \\ \alpha_{t,j} &= \text{softmax}(e_{t,j}). \end{aligned} \tag{11}$$

σ is a nonlinear function with the information of y_{t-1} (its word embedding being $\mathbf{y}_{t-1}$) added. In our preliminary experiments, we found that GRU works slightly better than tanh function, but we chose the latter for simplicity.

Decoder The decoder follows Equation (9) with fixed direction term $d = -1$. At the first layer, we use the following input:

$$\mathbf{x}_t = [\mathbf{c}_t, \mathbf{y}_{t-1}]$$

where $\mathbf{y}_{t-1}$ is the target word embedding at time step t, $\mathbf{c}_t$ is dynamically obtained follows Equation (10). There are L_{dec} layers of RNNs armed with LAUs in the decoder. At inference stage, we only utilize the top-most hidden states $\mathbf{s}^{L_{\text{dec}}}$ to make the final prediction with a softmax layer:

$$p(y_i|y_{<i}, \mathbf{x}) = \text{softmax}(\mathbf{W}_o \mathbf{s}_i^{L_{\text{dec}}}) \tag{12}$$

.

4 Experiments

4.1 Setup

We mainly evaluated our approaches on the widely used NIST Chinese-English translation task. In order to show the usefulness of our approaches, we also provide results on other two translation tasks: English-French, English-German. The evaluation metric is BLEU[2] (Papineni et al., 2002).

For Chinese-English, our training data consists of 1.25M sentence pairs extracted from LDC corpora[3], with 27.9M Chinese words and 34.5M English words respectively. We choose NIST 2002 (MT02) dataset as our development set, and the NIST 2003 (MT03), 2004 (MT04) 2005 (MT05) and 2006 (MT06) datasets as our test sets.

For English-German, to compare with the results reported by previous work (Luong et al., 2015; Zhou et al., 2016; Jean et al., 2015), we used the same subset of the WMT 2014 training corpus that contains 4.5M sentence pairs with 91M English words and 87M German words. The concatenation of news-test 2012 and news-test 2013 is used as the validation set and news-test 2014 as the test set.

To evaluate at scale, we also report the results of English-French. To compare with the results reported by previous work on end-to-end NMT (Sutskever et al., 2014; Bahdanau et al., 2014; Jean et al., 2015; Luong et al., 2014; Zhou et al., 2016), we used the same subset of the WMT 2014 training corpus that contains 12M sentence pairs with 304M English words and 348M French words. The concatenation of news-test 2012 and news-test 2013 serves as the validation set and news-test 2014 as the test set.

4.2 Training details

Our training procedure and hyper parameter choices are similar to those used by (Bahdanau et al., 2014). In more details, we limit the source and target vocabularies to the most frequent $30K$ words in both Chinese-English and English-French. For English-German, we set the source and target vocabularies size to $120K$ and $80K$, respectively.

For all experiments, the dimensions of word embeddings and recurrent hidden states are both set to 512. The dimension of c_t is also of size 512. Note that our network is more narrow than most previous work where hidden states of dimmention 1024 is used. we initialize parameters by sampling each element from the Gaussian distribution with mean 0 and variance 0.04^2.

Parameter optimization is performed using stochastic gradient descent. Adadelta (Zeiler,

[1] github.com/nyu-dl/dl4mt-tutorial/tree/master/session2

[2] For Chinese-English task, we apply case-insensitive NIST BLEU. For other tasks, we tokenized the reference and evaluated the performance with *multi-bleu.pl*. The metrics are exactly the same as in previous work.

[3] The corpora include LDC2002E18, LDC2003E07, LDC2003E14, Hansards portion of LDC2004T07, LDC2004T08 and LDC2005T06.

SYSTEM	MT03	MT04	MT05	MT06	AVE.
Existing systems					
Moses	31.61	33.48	30.75	30.85	31.67
Groundhog	31.92	34.09	31.56	31.12	32.17
COVERAGE	34.49	38.34	34.91	34.25	35.49
MEMDEC	36.16	39.81	35.91	35.98	36.95
Our deep NMT systems					
DEEPGRU	33.21	36.76	33.05	33.30	34.08
DEEPLAU	**39.35**	**41.15**	**38.07**	**37.29**	**38.97**
DEEPLAU +Ensemble + PosUnk	42.21	43.85	44.75	42.58	43.35

Table 1: Case-insensitive BLEU scores on Chinese-English translation.

2012) is used to automatically adapt the learning rate of each parameter ($\epsilon = 10^{-6}$ and $\rho = 0.95$). To avoid gradient explosion, the gradients of the cost function which had ℓ_2 norm larger than a predefined threshold τ were normalized to the threshold (Pascanu et al., 2013a). We set τ to 1.0 at the beginning and halve the threshold until the BLEU score does not change much on the development set. Each SGD is a mini-batch of 128 examples. We train our NMT model with the sentences of length up to 80 words in the training data, while for the Moses system we use the full training data. Translations are generated by a beam search and log-likelihood scores are normalized by sentence length. We use a beam width of 10 in all the experiments. Dropout was also applied on the output layer to avoid over-fitting. The dropout rate is set to 0.5. Except when otherwise mentioned, NMT systems are have 4 layers encoders and 4 layers decoders.

4.3 Results on Chinese-English Translation

Table 1 shows BLEU scores on Chinese-English datasets. Clearly DEEPLAU leads to a remarkable improvement over their competitors. Compared to DEEPGRU, DEEPLAU is +4.89 BLEU score higher on average four test sets, showing the modeling power gained from the liner associative connections. We suggest it is because LAUs apply adaptive gate function conditioned on the input which make it able to automatically decide how much linear information should be transferred to the next step.

To show the power of DEEPLAU, we also make a comparison with previous work. Our best single model outperforms both a phrased-based MT system (Moses) as well as an open source attention-based NMT system (Groundhog) by +7.3 and +6.8 BLEU points respectively on average. The result is also better than some other state-of-the-art variants of attention-based NMT mode with big margins. After PosUnk and ensemble, DEEPLAU seizes another notable gain of +4.38 BLEU and outperform Moses by +11.68 BLEU.

4.4 Results on English-German Translation

The results on English-German translation are presented in Table 2. We compare our NMT systems with various other systems including the winning system in WMT14 (Buck et al., 2014), a phrase-based system whose language models were trained on a huge monolingual text, the Common Crawl corpus. For end-to-end NMT systems, to the best of our knowledge, Wu et al. (2016) is currently the SOTA system and about 4 BLEU points on top of previously best reported results even though Zhou et al. (2016) used a much deeper neural network[4].

Following Wu et al. (2016), the BLEU score represents the averaged score of 8 models we trained. Our approach achieves comparable results with SOTA system. As can be seen from the Table 2, DeepLAU performs better than the word based model and even not much worse than the best wordpiece models achieved by Wu et al. (2016). Note that DEEPLAU are sim-

[4] It is also worth mentioning that the result reported by Zhou et al. (2016) does not include PosUnk, and this comparison is not fair enough.

SYSTEM	Architecture	Voc.	BLEU
Existing systems			
Buck et al. (2014)	Winning WMT14 system phrase-based + large LM	-	20.7
Jean et al. (2015)	gated RNN with search + LV + PosUnk	$500K$	19.4
Luong et al. (2015)	LSTM with 4 layers + dropout + local att. + PosUnk	$80K$	20.9
Shen et al. (2015)	gated RNN with search + PosUnk + MRT	$80K$	20.5
Zhou et al. (2016)	LSTM with 16 layers + F-F connections	$80K$	20.6
Wu et al. (2016)	LSTM with 8 laysrs + RL-refined Word	$80K$	23.1
Wu et al. (2016)	LSTM with 8 laysrs + RL-refined WPM-32K	-	24.6
Wu et al. (2016)	LSTM with 8 laysrs + RL-refined WPM-32K + Ensemble	-	26.3
Our deep NMT systems			
this work	DEEPLAU	$80K$	22.1(±0.3)
this work	DEEPLAU + PosUnk	$80K$	23.8(±0.3)
this work	DEEPLAU + PosUnk + Ensemble 8 models	$80K$	26.1

Table 2: Case-sensitive BLEU scores on German-English translation.

ple and easy to implement, as opposed to previous models reported in Wu et al. (2016), which dependends on some external techniques to achieve their best performance, such as their introduction of length normalization, coverage penalty, fine-tuning and the RL-refined model.

4.5 Results on English-French Translation

SYSTEM	BLEU
Enc-Dec (Luong et al., 2014)	30.4
RNNsearch (Bahdanau et al., 2014)	28.5
RNNsearch-LV (Jean et al., 2015)	32.7
Deep-Att (Zhou et al., 2016)	35.9
DEEPLAU	35.1

Table 3: English-to-French task: BLEU scores of single neural models.

To evaluate at scale, we also show the results on an English-French task with $12M$ sentence pairs and $30K$ vocabulary in Table 3. Luong et al. (2014) achieves BLEU score of 30.4 with a six layers deep Encoder-Decoder model. The two attention models, RNNSearch and RNNsearch-LV achieve BLEU scores of 28.5 and 32.7 respectively. The previous best single NMT Deep-Att model with an 18 layers encoder and 7 layers decoder achieves BLEU score of 35.9. For DEEPLAU, we obtain the BLEU score of 35.1 with a 4 layers encoder and 4 layers decoder, which is on par with the SOTA system in terms of BLEU. Note that Zhou et al. (2016) utilize a much larger depth as well as external alignment model and extensive regularization to achieve their best results.

4.6 Analysis

Then we will study the main factors that influence our results on NIST Chinese-English translation task. We also compare our approach with two SOTA topologies which were used in building deep NMT systems.

- Residual Networks (ResNet) are among the pioneering works (Szegedy et al., 2016; He et al., 2016) that utilize extra identity connections to enhance information flow such that very deep neural networks can be effectively optimized. Share the similar idea, Wu et al. (2016) introduced to leverage residual connections to train deep RNNs.
- Fast Forward (F-F) connections were proposed to reduce the propagation path length which is the pioneer work to simplify the training of deep NMT model (Zhou et al., 2016). The work can be viewed as a parametric ResNet with short cut connections between adjacent layers. The procedure takes a linear sum between the input and the newly computed state.

LAU vs. GRU Table 4 shows the effect of the novel LAU. By comparing row 3 to row 7, we see that when L_{Enc} and L_{Dec} are set to 2,

SYSTEM	(L_{enc},L_{Dec})	width	AVE.
1 DeepGRU	(2,1)	512	33.59
2 DeepGRU	(2,2)	1024	34.68
3 DeepGRU	(2,2)	512	34.91
4 DeepGRU	(4,4)	512	34.08
5 4+ResNet	(4,4)	512	36.40
6 4+F-F	(4,4)	512	37.62
7 DeepLAU	(2,2)	512	37.65
8 DeepLAU	(4,4)	512	38.97
9 DeepLAU	(8,6)	512	39.01
10 DeepLAU	(8,6)	256	38.91

Table 4: BLEU scores of DeepLAU and DeepGRU with different model sizes.

the average BLEU scores achieved by DeepGRU and DeepLAU are 34.68 and 37.65, respectively. LAU can bring an improvement of 2.97 in terms of BLEU. After increasing the model depth to 4 (row 4 and row 6), the improvement is enlarged to 4.91. When DeepGRU is trained with larger depth (say, 4), the training becomes more difficult and the performance falls behind its shallow partner. While for DeepLAU, as can be see in row 9, with increasing the depth even to $L_{\text{Enc}} = 8$ and $L_{\text{Dec}} = 6$ we can still obtain growth by 0.04 BLEU score. Compared to previous shortcut connection methods (row 5 and row 6), The LAU still achieve meaningful improvements over F-F connections and Residual connections by +1.35 and +2.57 BLEU points respectively.

DeepLAU introduces more parameters than DeepGRU. In order to figure out the effect of DeepLAU comparing models with the same parameter size, we increase the hidden size of DeepGRU model. Row 3 shows that, after using a twice larger GRU layer, the BLEU score is 34.68, which is still worse than the corresponding DeepLAU model with fewer parameters.

Depth vs. Width Next we will study the model size. In Table 4, starting from $L_{\text{Enc}} = 2$ and $L_{\text{Dec}} = 2$ and gradually increasing the model depth, we can achieve substantial improvements in terms of BLEU. With $L_{\text{Enc}} = 8$ and $L_{\text{Dec}} = 6$, our DeepLAU model yields the best BLEU score. We tried to increase the model depth with the same hidden size but failed to see further improvements.

We then tried to increase the hidden size. By comparing row 2 and row 3, we find the improvements is relative small with a wider hidden size. It is also worth mentioning that a deep and thin network with fewer parameters can still achieve comparable results with its shallow partner. This suggests that depth plays a more important role in increasing the complexity of neural networks than width and our deliberately designed LAU benefit from the optimizing of such a deep model.

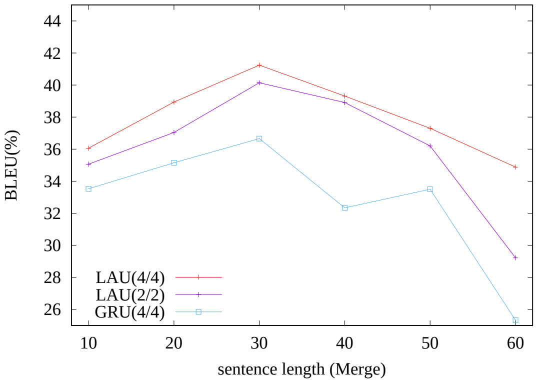

Figure 2: The BLEU scores of generated translations on the merged four test sets with respect to the lengths of source sentences.

About Length A more detailed comparison between DeepLAU (4 layers encoder and 4 layers decoder), DeepLAU(2 layer encoder and 2 layer decoder) and DeepGRU (4 layers encoder and 4 layers decoder), suggest that with deep architectures are essential to the superior performance of our system. In particular, we test the BLEU scores on sentences longer than $\{10, 20, 30, 40, 50, 60\}$ on the merged test set. Clearly, in all curves, performance degrades with increased sentence length. However, DeepLAU models yield consistently higher BLEU scores than the DeepGRU model on longer sentences. These observations are consistent with our intuition that very deep RNN model is especially good at modeling the nested latent structures on relatively complicated sentences and LAU plays an important role on optimizing such a complex deep model.

5 Conclusion

We propose a Linear Associative Unit (LAU) which makes a fusion of both linear and non-linear transformation inside the recurrent unit. On this way, gradients decay much slower compared to the standard deep networks which enable us to build a deep neural network for machine translation. Our empirical study shows that it can significantly improve the performance of NMT.

6 acknowledge

We sincerely thank the anonymous reviewers for their thorough reviewing and valuable suggestions. Wang's work is partially supported by National Science Foundation for Deep Semantics Based Uighur to Chinese Machine Translation (ID 61662077). Qun Liu's work is partially supported by Science Foundation Ireland in the ADAPT Centre for Digital Content Technology (www.adaptcentre.ie) at Dublin City University funded under the SFI Research Centres Programme (Grant 13/RC/2106) co-funded under the European Regional Development Fund.

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473* .

Christian Buck, Kenneth Heafield, and Bas Van Ooyen. 2014. N-gram counts and language models from the common crawl. In *LREC*. Citeseer, volume 2, page 4.

David Chiang. 2005. A hierarchical phrase-based model for statistical machine translation. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*. Association for Computational Linguistics, pages 263–270.

Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078* .

Alex Graves. 2013. Generating sequences with recurrent neural networks. *arXiv preprint arXiv:1308.0850* .

Alex Graves, Greg Wayne, and Ivo Danihelka. 2014. Neural turing machines. *arXiv preprint arXiv:1410.5401* .

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2015. Deep residual learning for image recognition. *arXiv preprint arXiv:1512.03385* .

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. pages 770–778.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* 9(8):1735–1780.

Sébastien Jean, Kyunghyun Cho, Roland Memisevic, and Yoshua Bengio. 2015. On using very large target vocabulary for neural machine translation. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. Association for Computational Linguistics, Beijing, China, pages 1–10. http://www.aclweb.org/anthology/P15-1001.

Nal Kalchbrenner, Ivo Danihelka, and Alex Graves. 2015. Grid long short-term memory. *arXiv preprint arXiv:1507.01526* .

Philipp Koehn, Franz Josef Och, and Daniel Marcu. 2003. Statistical phrase-based translation. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1*. Association for Computational Linguistics, pages 48–54.

Yang Liu, Qun Liu, and Shouxun Lin. 2006. Tree-to-string alignment template for statistical machine translation. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, pages 609–616.

Minh-Thang Luong, Hieu Pham, and Christopher D Manning. 2015. Effective approaches to attention-based neural machine translation. *arXiv preprint arXiv:1508.04025* .

Minh-Thang Luong, Ilya Sutskever, Quoc V Le, Oriol Vinyals, and Wojciech Zaremba. 2014. Addressing the rare word problem in neural machine translation. *arXiv preprint arXiv:1410.8206* .

Fandong Meng, Zhengdong Lu, Zhaopeng Tu, Hang Li, and Qun Liu. 2015. Neural transformation machine: A new architecture for sequence-to-sequence learning. *CoRR* abs/1506.06442. http://arxiv.org/abs/1506.06442.

Haitao Mi, Liang Huang, and Qun Liu. 2008. Forest-based translation. In *ACL*. pages 192–199.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting on association for computational linguistics*. Association for Computational Linguistics, pages 311–318.

Razvan Pascanu, Caglar Gulcehre, Kyunghyun Cho, and Yoshua Bengio. 2013a. How to construct deep recurrent neural networks. *arXiv preprint arXiv:1312.6026* .

Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. 2013b. On the difficulty of training recurrent neural networks. *ICML (3)* 28:1310–1318.

Shiqi Shen, Yong Cheng, Zhongjun He, Wei He, Hua Wu, Maosong Sun, and Yang Liu. 2015. Minimum risk training for neural machine translation. *arXiv preprint arXiv:1512.02433* .

Rupesh K Srivastava, Klaus Greff, and Jürgen Schmidhuber. 2015. Training very deep networks. In *Advances in neural information processing systems*. pages 2377–2385.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*. pages 3104–3112.

Christian Szegedy, Sergey Ioffe, Vincent Vanhoucke, and Alex Alemi. 2016. Inception-v4, inception-resnet and the impact of residual connections on learning. *arXiv preprint arXiv:1602.07261* .

Zhaopeng Tu, Zhengdong Lu, Yang Liu, Xiaohua Liu, and Hang Li. 2016. Modeling coverage for neural machine translation. *ArXiv eprints, January* .

Ronald J Williams and David Zipser. 1989. A learning algorithm for continually running fully recurrent neural networks. *Neural computation* 1(2):270–280.

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

Deyi Xiong, Qun Liu, and Shouxun Lin. 2006. Maximum entropy based phrase reordering model for statistical machine translation. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, pages 521–528.

Matthew D Zeiler. 2012. Adadelta: an adaptive learning rate method. *arXiv preprint arXiv:1212.5701* .

Biao Zhang, Deyi Xiong, and Jinsong Su. 2016. Variational neural machine translation. *arXiv preprint arXiv:1605.07869* .

Jiajun Zhang and Chengqing Zong. 2016. Exploiting source-side monolingual data in neural machine translation. In *Proceedings of EMNLP*.

Jie Zhou, Ying Cao, Xuguang Wang, Peng Li, and Wei Xu. 2016. Deep recurrent models with fast-forward connections for neural machine translation. *arXiv preprint arXiv:1606.04199* .