文章编号：1003-0077（2017）00-0000-00

基于简单循环单元的深层神经网络机器翻译模型

张文[1,2] 冯洋[1,2] 刘群[3,1]

（1.中国科学院计算技术研究所，智能信息处理重点实验室，北京 100190；
2.中国科学院大学，北京 100049；
3.都柏林城市大学，都柏林）

摘要：基于注意力机制的神经网络机器翻译模型已经成为目前主流的翻译模型，在许多翻译方向上均超过了统计机器翻译模型，尤其是在训练语料规模比较大的情况下，优势更加明显。该模型使用编码器-解码器框架，将翻译任务建模成序列到序列的问题。然而，在基于门控循环单元（Gated Recurrent Unit，GRU）的编码器-解码器模型中，随着模型层数的增加，梯度消失的问题使模型难以收敛并且严重退化，进而使翻译性能下降。在本文中，我们使用了一种简单循环单元（Simple Recurrent Unit，SRU）代替了 GRU 单元，通过堆叠网络层数加深编码器和解码器的结构，提高了神经网络机器翻译模型的性能。我们在德语-英语和维语-汉语翻译任务上进行了实验，实验结果表明，在神经网络机器翻译模型中使用 SRU 单元，可以有效地解决梯度消失带来的模型难以训练的问题；通过加深模型能够显著性地提升系统的翻译性能，同时保证训练速度基本不变。此外，我们还与基于残差连接（Residual Connections）的神经网络机器翻译模型进行了实验对比，实验结果表明，我们的模型有显著性优势。

关键词：门控循环单元；梯度消失；残差连接；简单循环单元

中图分类号：TP391 **文献标识码：**A

Deep Neural Machine Translation Model Based on Simple Recurrent Units

Wen Zhang[1,2], Yang Feng[1,2], Qun Liu[3,1]

(1.The Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China, 100190;
2.University of Chinese Academy of Sciences, Beijing, China, 100190;
3.ADAPT Centre, School of Computing, Dublin City University, Dublin 9, Ireland)

Abstract : Attention-based neural machine translation models become extremely popular, which have outperformed statistical machine translation on many directions of translation, especially in the case of the large training corpus. Based on an encoder-decoder framework, the machine translation is modelled as a sequence to sequence problem. However, in the encoder-decoder model with gated recurrent units, as the increasing of model layers, the problem of gradient vanishing happens and lets the model hard to converge and be deteriorating seriously, declining the translation quality further. In this paper, we replace the gated recurrent units in the naïve encoder and decoder with the simple recurrent units (SRUs), and deepen the structure of the encoder and decoder by stacking network layers to improve the performance of neural machine translation model. We conducted experiments on the German-English and Uyghur-Chinese translation tasks. Experiment results show that, introducing the simple recurrent units into the neural machine translation model could effectively solve the issue of being hard to train aroused by gradient vanishing; We improve the performance significantly by deepening the model, meanwhile, keeping the training speed almost unchanged. Besides, our model could get significant better results than the baseline model with residual connections.

Key words: gated recurrent unit; gradient vanishing; residual connection; simple recurrent unit

收稿收稿日期：2017-03-16；**定稿日期**：2017-04-26
基金项目：基金名（基金号）；基金名(基金号)

1 引言

近十年来，随着深度学习技术的发展，许多研究人员逐渐把机器翻译当作是序列到序列的任务，并使用编码器-解码器的神经网络框架对其进行建模，给机器翻译质量带来了很大的飞跃，大幅度超过了传统方法[1][2]。基于编码器-解码器的神经机器翻译模型凭借它简单有效的优势脱颖而出，已经成为当今主流的翻译模型之一。随后，注意力机制[3]的引入又给基于编码器-解码器的神经翻译模型带来了一次飞跃。近几年来，许多研究人员在基于注意力机制的神经机器翻译模型基础之上，从不同角度进行了后续的研究工作，使神经翻译模型得到了许多方面的优化[4][5][6][7]。

目前主流的基于注意力机制的神经翻译模型的编码器和解码器多数是使用门控循环单元或长短期记忆网络（Long Short-Term Memory，LSTM[8]）。GRU 和 LSTM 类似，均是顺序地以词为单位对序列进行循环展开，然后通过内部精心设计的门控机制来控制每一个循环步骤中输入和输出的信息量。这一机制促使模型可以存取序列中更远的上下文信息，同时也缓解了普通循环神经网络在过长序列上存在的梯度消失问题。GRU 单元或 LSTM 解决了普通循环神经网络的诸多问题，但即便如此，在基于 GRU 单元的神经网络机器翻译模型中仍然存在着一些显而易见和令人棘手的问题。对于卷积神经网络来说，增加神经网络的层数会使模型的能力增强[9],[10],[11]，这也是符合人们直观想法的。然而，当我们试图增加基于 GRU 单元的神经网络翻译模型中编码器和解码器的深度来优化翻译模型的性能时发现，翻译的质量并没有随之增加，有时反而有所下降。我们猜想这是由于增加模型深度所导致梯度消失问题使得模型难以收敛，进而影响最终译文的质量。另外，层数的叠加会严重地影响训练速度。

本文中，我们在基于注意力机制的神经网络翻译模型中引入简单循环单元（Simple Recurrent Unit，SRU[12]）并替换编码器和解码器中门控循环单元。在小规模德语到英语和大规模维语到汉语翻译任务上的实验表明，通过堆叠编码器和解码器的层数，我们获得了比基线系统有显著性提升的翻译性能，甚至优于融合残差连接的基线系统。而且在 1-10 层之内，翻译质量与层数是成正比的。同时不像 GRU 单元那样，编码器和解码器中 SRU 层数的增加对训练速度的影响并不大。

在这篇文章接下来的几节内容里，我们首先在第二节中对相关工作做一个简要的介绍；然后，在第三节中对基于注意力机制的神经网络翻译模型以及简单循环单元进行数学形式化；在第四节中，重点描述网络层堆叠方式以及如何替换基于注意力机制的神经网络翻译模型中的 GRU 单元；之后，在第五节中，我们在两个数据集上设计了四组实验验证所提出方法的有效性；最后把所得出的结论以及对未来工作方向的阐述放到第六节里。

2 相关工作

国内外有许多学者试图探索一些有效的方法来增加模型的深度，以改进模型的性能。由于比较深层的神经网络模型往往难以训练，不容易收敛，Kaiming He 等人[10]提出残差连接，把前一层的输入信息通过直连边与当前层的输出相连，疏通梯度传播，减轻网络的训练难度，他们在图像识别的任务上证明了残差网络使模型更加容易优化，通过加深模型获得非常好的效果；Alexis Conneau 等人[11]提出了一种新的网络架构，该架构仅使用卷积和池操作，使得模型的能力会随着模型深度的增加而增强，他们将模型的卷积层增加到 29，在一些公共文本分类任务上取得最好的效果。

另外，也有一些研究人员尝试改进卷积神经网络中的残差连接。赵朋成等人[13]对传统残差网络进行改进，通过在池化层使用重叠池化方案有效地保留上一层有用信息，从而有效地提高了手体识别率,而且训练时间大幅度降低。王一宁等人[14]使用一种多阶段级联残差卷积神经网络模型，能更好地重建出图像的细节和纹理，避免了经过迭代之后造成的图像过度平滑，获得更高的峰值信噪比和平均结构相似度。

学者们也在尝试使用神经翻译模型改进少数民族语言的翻译。哈里旦木等人[15]从词语切分粒度的角度，分析并对比了多个翻译模型在维汉翻译任务上的性能，证明维语采用恰当的子词切分且中文端基于字符能够使模型获得最佳性能。Lei 等人[12]提出了简单循环单元，并将其引入到多个任务上的神经网络模型中，通过增加模型深度提升了模型性能，同时保证了训练速度。

我们借鉴前人的相关工作，使用 SRU 单元替换基于注意力机制的神经网络翻译模型中的 GRU 单元，解决加深模型所导致的梯度消失问题。然后通过加深模型中编码器和解码器的层数，稳定地提升模型的性能，产生比基线模型更加准确的译文。基于简单循环单元的深度模型性能还要优于同层数的引入残差网络的基线模型。

3 背景

3.1 基于注意力机制的神经翻译模型

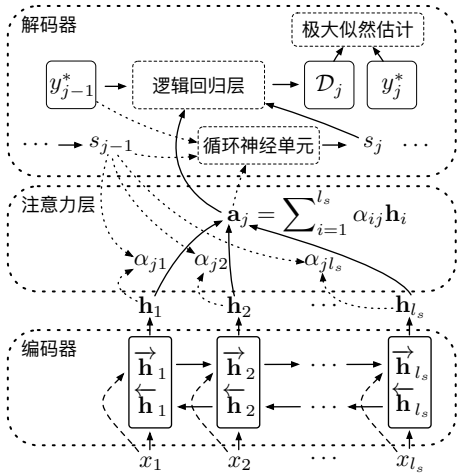

图 1 基于注意力机制的神经翻译模型

给定平行句对 $\mathbf{x}=\{x_1,x_2,\cdots,x_{l_s}\}$ 和 $\mathbf{y}=\{y_1^*,y_2^*,\cdots,y_n^*\}$，我们使用双向循环神经网络对 $\mathbf{x}$ 进行编码：

$$\mathbf{h}_i=\begin{bmatrix}\overrightarrow{\mathbf{h}}_i\\ \overleftarrow{\mathbf{h}}_i\end{bmatrix}=\begin{bmatrix}\overrightarrow{\mathrm{GRU}}(x_i,\overrightarrow{\mathbf{h}}_{i-1})\\ \overleftarrow{\mathrm{GRU}}(x_i,\overrightarrow{\mathbf{h}}_{i+1})\end{bmatrix} \tag{1}$$

其中 $\overrightarrow{\mathrm{GRU}}$ 和 $\overleftarrow{\mathrm{GRU}}$ 是两个门控循环单元，从两个方向循环地对 x 编码，然后拼接每个词的输出状态，$\mathbf{h}=\{\mathbf{h}_1,\mathbf{h}_2,\cdots,\mathbf{h}_{l_s}\}$。

解码器通过最大化目标待预测词汇的似然来优化整个翻译模型

$$p\left(y_j\middle|y_{j-1}^*,s_j,\mathbf{a}_j\right)\propto\exp\left(o\left(y_{j-1}^*,s_j,\mathbf{a}_j\right)\right) \tag{2}$$

$o(\cdot)$ 是一个线性转换函数，y_{j-1}^* 是第 $j-1$ 步的参考词语，s_j 是解码器循环神经单元第 j 步时的隐状态

$$s_j=\mathrm{GRU}\left(y_{j-1}^*,s_{j-1},\mathbf{a}_j\right) \tag{3}$$

其中 $\mathbf{a}_j$ 是每个解码步都动态更新的源端上下文表示，即每个源端词语状态的加权和

$$\mathbf{a}_j=\sum_{i=1}^{l_s}\alpha_{i,j}\mathbf{h}_i \tag{4}$$

$\alpha_{i,j}$ 可被看作是源端第 i 个词语与目标端第 j 个词语之间的对齐概率

$$\alpha_{i,j}=\exp\left(e_{i,j}\right)/\sum_{i=1}^{l_s}\exp\left(e_{i,j}\right) \tag{5}$$

$$e_{i,j}=\mathrm{f}\left(s_{j-1},\mathbf{h}_i\right) \tag{6}$$

f(·) 是一个前馈神经网络，图 1 展示了基于注意力机制的神经网络翻译模型。其中 $\mathcal{D}_j$ 表示模型在第 j 步所预测词语的概率分布，极大似然估计表示计算损失的方法。

3.2 简单循环单元

简单循环单元的主要思想是避免 GRU 中状态计算和门控机制的复杂程度，消除门状态对前一步隐状态的依赖性，从而提高门计算单元的可并行性以加快训练速度。同时引入高速网络（Highway Network）使得模型能自主控制从前一层的输入中选择多少信息量不经过任何非线性变换，直接输入到下一层。这样做确保在加深模型深度时，尽量避免梯度消失的问题。

主流循环神经单元（比如 GRU 或 LSTM）利用门机制来控制同一层不同时序状态中信息流的输入和输出量，从而有效地避免了序列过长时所产生的梯度消失问题。第 t 步循环计算单元的内部存储单元 c_t 是通过下列公式计算得到

$$c_t=f_t*c_{t-1}+i_t*g_t \tag{7}$$

$*$ 表示按元素相乘。在循环计算单元的原生定义中，忘记门 f_t，输入门 i_t 以及候选隐状态 g_t 的计算均需要依赖于当前步的输入 x_t 和前一步的输出隐状态 h_{t-1}，分别是 x_t 和 h_{t-1} 加权和的非线性变换。这里对原生计算进行简化操作，g_t 是对当前输入 x_t 的一个线性变换：$g_t=\mathrm{W}x_t$。输入门 i_t 使用一个耦合版本：

$$i_t=1-f_t \tag{8}$$

下一步内部存储单元 c_t 的计算改变为：

$$c_t=f_t*c_{t-1}+(1-f_t)*(\mathrm{W}x_t) \tag{9}$$

最后，当前步的中间输出状态 $\tilde{h}_t$ 是对内部状态 c_t 的直接非线性变换 $\tilde{h}_t=\tanh(c_t)$。特别地，为了更加高效地训练深层 SRU 模型，高速连接操作被用于生成最终的输出状态：

$$\begin{aligned}h_t&=(1-z_t)*\tilde{h}_t+z_t*x_t\\&=(1-z_t)*\tanh(c_t)+z_t*x_t\end{aligned} \tag{10}$$

其中，z_t 是当前步的重置门。另外，所有门状态的计算仅仅依赖于当前步骤的输入 x_t，取消对前一个输出状态 h_{t-1} 的依赖，所以对一个第 t 步输出，SRU 的整个计算过程如下

$$f_t=\sigma(\mathrm{W}_f x_t+\mathrm{b}_f) \tag{11}$$

$$z_t=\sigma(\mathrm{W}_z x_t+\mathrm{b}_z) \tag{12}$$

$$c_t=f_t*c_{t-1}+(1-f_t)*(\mathrm{W}x_t) \tag{13}$$

$$h_t=(1-z_t)*\tanh(c_t)+z_t*x_t \tag{14}$$

其中 W，W_f，W_z 为 SRU 中的参数矩阵，b_f 和 b_z 为偏置单元向量。从上述公式可以看出，对于一个输入序列 $\mathbf{x}=\{x_1,x_2,\cdots,x_{l_s}\}$，忘记门和重置门单元的计算仅依赖于输入序列，所以它们可以在序列长度的维度上实现并行计算。但是，GRU 中的门控状态的计算必须依赖于前一步的隐状态，所以在前一步隐状态没有计算完成的情况下，当前步的门控状态向量是无法获得的。这一局限性使 GRU 的中间门控状态计算的可并行性受到约束。

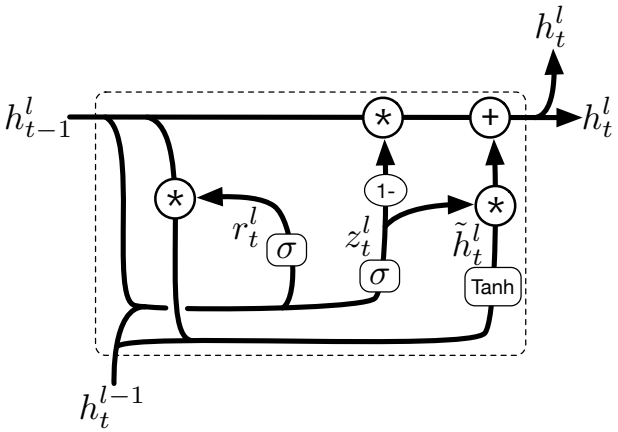

图 2 基于 GRU 的深层循环单元示意图

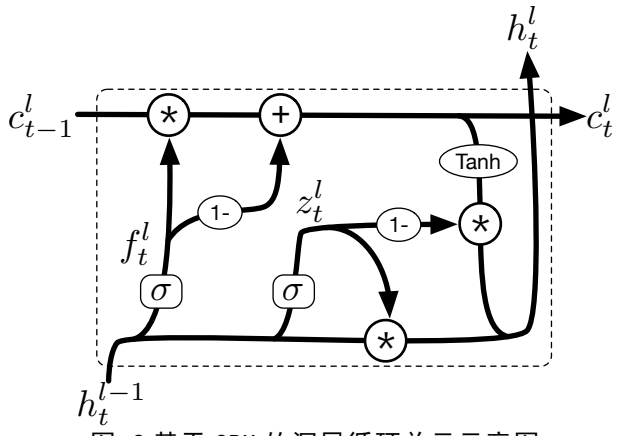

图 3 基于 SRU 的深层循环单元示意图

4 融合简单循环单元的深层神经翻译模型

4.1 基于 GRU 的深层循环单元

我们在基于注意力机制的神经翻译模型的基础上进行改进，该模型的编码器和解码器中使用 GRU 作为循环单元，且每个 GRU 只有一层，我们将 GRU 扩展到多层，每一层的输入是前一层的输出状态。每一层拥有自身的隐状态，是在序列长度上进行展开的。如图 2 所示，l 表示堆叠层数的索引，t 表示循环序列的步数索引。对于第 l 层的第 t 步，第 $l-1$ 层第 t 步的输出 h_t^{l-1} 作为输入，输出状态是 h_t^l 。第 l 层前一步的隐状态 h_{t-1}^l 转换成第 t 步的隐状态 h_t^l 。值得注意的是，对于 GRU 而言，第一层的输入是 $\mathbf{x}=\{x_1,x_2,\cdots,x_n\}$ ，每一层的输出状态和该层下一步的隐状态相同。

4.2 基于 SRU 的深层循环单元

把循环单元当作是一个黑匣子，与 GRU 作对比，SRU 每一步的输出状态和中间隐状态是不同的，而 GRU 的输出状态与隐状态相同。根据 3.2 节 SRU 的计算公式我们画出第 l 层第 t 步计算单元的示意图（如图 3 所示），其中 l 表示堆叠层数的索引，t 表示循环序列的步数索引， c_t^l 为第 l 层第 t 步的隐状态。前一层的输出状态 h_t^{l-1} 作为输入，经过当前计算单元的一系列运算，前一步的隐状态 c_t^{l-1} 被转换为 c_t^l ，输出状态是 h_t^l 。对于第一层而言， $h_t^{l-1}=x_t$ 。GRU 和 SRU 的对比如图 2 和图 3 所示。

4.3 堆叠深层循环单元

不论是对于 GRU 还是 SRU，我们都采用同样的层堆叠方式。对于每一个循环步骤，前一步的所有隐状态作为输入，逐层地输出当前步的输出状态与所有隐状态。图 4 展示了双向循环神经网络在序列长度维度上的展开以及在层次维度上的堆叠过程。其中方框表示循环单元，箭头表示信息流的方向， $\{x_1,x_2,\cdots,x_{l_s}\}$ 是第一层输入序列，两个方向上每一层的初始化隐状态分别表示为 $\{\vec{c}_0^1,\cdots,\vec{c}_0^L\}$ 和 $\{\overleftarrow{c}_0^1,\cdots,\overleftarrow{c}_0^L\}$， $\{\vec{c}_{l_s}^1,\cdots,\vec{c}_{l_s}^L\}$ 和 $\{\overleftarrow{c}_{l_s}^1,\cdots,\overleftarrow{c}_{l_s}^L\}$ 分别表示在两个方向上初始化隐状态经过 L 层循环计算单元展开之后输出的所有隐状态。 $\{h_1^L,h_2^L,\cdots,h_{l_s}^L\}$ 表示输入序列经过双向 L 层堆叠之后的输出状态序列。对于第 t 个循环步，每一层的输入为前一层的输出状态（第一层的输入为 x_t ）。对于每一个网络层，前一个步骤的隐状态通过一次循环计算单元转换成该层下一个步骤的隐状态（我们使用零向量初始化每一层的隐状态 $\{\vec{c}_0^1,\cdots,\vec{c}_0^L\}$ 和 $\{\overleftarrow{c}_0^1,\cdots,\overleftarrow{c}_0^L\}$）。对于编码器中的多层双向循环网络，我们使用图 4 的堆叠方式；解码器中使用多层单向循环网络，即图 4 的下半部分。

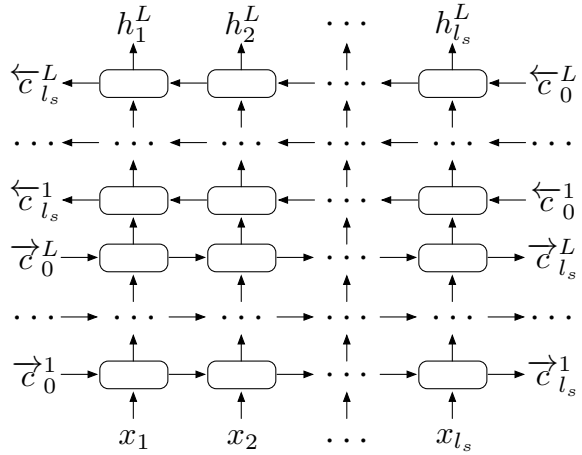

图 4 多层双向循环神经网络计算单元堆叠示意图

5 实验和分析

我们在两个翻译任务上进行实验，分别是小规模的德语到英语翻译任务和大规模的维语到汉语翻译任务。

5.1 训练数据和前处理

德英数据来自于 2014 年 IWSLT 评测竞赛中德英翻译任务，该语料主要是由 TED 演讲中双语对齐的字幕所组成。我们使用摩西的工具包[16]对德语

表 1 语料数量和词典大小统计

	德-英	维-汉
训练集	153326	2887973
源端词典	32013	32376
目标端词典	22826	38377
开发集	6969	900
测试集	6750	861

源端和英语目标端进行标记化，标记化之后源端或目标端句子长度大于 50 的平行句对被过滤掉。开发集是从训练集中抽取的 6969 个平行句对，训练集除去开发集部分一共是 153326 个平行句对，测试集是 dev2010，dev2012，tst2010，tst2011 和 tst2012 的合并，一共包含 6750 句[1]。

对于维汉翻译方向，我们使用新疆大学收集的新闻领域语料，训练语料包含 289 万左右的平行句对。新闻领域的开发集和测试集采用 2017 年 CWMT 上新疆大学所提供的，其中开发集包含 900 个维语句子，每句对应 4 个汉语参考译文，测试集包含 861 句，每句同样也有四个汉语参考译文。维语端做 32000 次操作的字节对编码（Byte Pair Encoding，BPE[17]），中文端以字为单位解码。语料前处理后，源端或目标端句子长度大于 80 的平行句对被过滤掉。两个翻译任务中语料规模的统计结果和词表大小见表 1。

5.2 评价指标

对于德英翻译方向，我们使用大小写敏感的 4 元 BLEU[18]值作为评价指标（摩西开源的 *multi-bleu.perl* 脚本）。而对于维汉翻译任务，我们使用基于字的大小写不敏感的 4 元 BLEU 值作为评价指标（摩西开源的 *mteval-v11b.pl* 脚本）。

5.3 系统和训练细节

我们使用基于 PyTorch[2]深度学习框架所实现的 RNNsearch* 模型作为基线系统。RNNsearch* 是 3.1 节中描述的 RNNsearch 模型的增强版本，在解码器中使用两个 GRUs 并引入中间隐状态 $\tilde{s}_j$，将公式（3）分解为以下两个公式

$$\tilde{s}_j = \mathrm{GRU}_1\left(y^*_{j-1}, s_{j-1}\right) \quad (15)$$

$$s_j = \mathrm{GRU}_2\left(\mathbf{a}_j, \tilde{s}_j\right) \quad (16)$$

在计算注意力模型 $\mathbf{a}_j$ 时，将 s_{j-1} 替换成所引入的中间隐状态 $\tilde{s}_j$，模型中其他的组件与 RNNsearch 保持一致。

RNNsearch* 模型的训练参数和细节如下所述。所有的参数矩阵和向量都使用 $[-0.1,0.1]$ 上的均匀分布进行初始化，所有的模型参数都使用批随机梯度下降算法进行优化，批大小设定为 80，学习率使用 Adadelta[19]算法进行自动调节，算法的衰减常数 $\rho = 0.95$，分母常数 $\epsilon = 1\mathrm{e}-6$。源端和目标端的词向量维度都设为 512，所有隐状态和输出状态神经元的大小也均设定为 512。为了避免梯度爆炸，如果所有梯度的 L2 范数大于预先设定的阈值 1.0，则每个梯度都除以 L2 范数以限定在阈值内。我们仅在模型最终输出层使用 dropout 策略，dropout 率为 0.5。测试时我们使用柱搜索算法进行解码，柱大小设置为 10，在进行柱搜索时，句子级别的负对数似然除以句长进行归一化。

表 2 深度 GRU 模型在德英翻译任务上的实验结果

层数	RNNsearch*		
	开发集	测试集	训练速度
GRU-1	29.65	27.62	2.23
GRU-2	29.10	27.34	2.76
GRU-4	27.56	25.90	3.51
GRU-6	25.16	23.05	4.05
GRU-8	20.63	19.29	4.50
GRU-10	2.86	2.83	4.17

另外，为了保证对比的公平性，对于 GRU 和 SRU 的隐状态，我们使用相同大小的神经元。所有实验在 GeForce GTX TITAN X 单块 GPU 上运行。我们使用 Collins 等人[20]提出的方法作显著性差异的检验。

5.4 实验结果

5.4.1 GRU 深层翻译模型

在基线系统中，RNNsearch* 模型的编码器中使用两个方向的 GRUs，而解码器中使用了两层单向的 GRUs。我们尝试增加基线系统的编码器和解码器中 GRUs 的层数，然后观察随着 GRUs 层数的增加，翻译性能和训练速度的变化情况。表 2 给出了德英任务上的实验结果，其中“训练速度”的单位是小时/轮。

我们将编码器和解码器层数从 1 增加至 10，开发集和测试集上的翻译性能并没有提升。相反，层数从 1 增加到 4 的过程中，RNNsearch* 的翻译质量迅速地下降，从第 6 层开始，其性能开始急剧退化。经过分析，原因是 RNNsearch* 的解码器中使用两个门控循环单元，层数的增多使模型在训练过程中反向求导时更加容易产生梯度消失而无法更新参数的现象。此外，实验表明，模型的深度越大，越难以收敛。如表 2 所示，当模型深度到达 10 层时，RNNsearch* 几乎已经无法收敛。另外，层数堆叠也使模型的参数随之增多，系统的训练速度变得越来越慢。

[1] IWSLT2014 德英翻译方向数据集的下载链接：http://wit3.fbk.eu/archive/2014-01/texts/de/en/de-en.tgz

[2] http://pytorch.org/

表 3 德英任务翻译结果

系统	开发集	测试集	速度	参数量
RNNsearch*	29.65	27.62	0.90	22.1
SRU-1	28.58	26.42	1.33	21.9
SRU-2	29.53	27.55	1.32	23.1
SRU-4	30.21	28.05	1.50	25.4
SRU-6	30.31	28.30	1.37	27.8
SRU-8	30.37	28.46	1.49	30.1
SRU-10	30.37	28.25	1.38	32.5
SRU-12	30.46	**28.56†**	1.45	34.9

5.4.2 SRU 深层翻译模型

从 5.4.1 节的实验结果可以看出，对于基线系统，堆叠编码器和解码器中的 GRU 计算单元，会使模型的性能退化，特别是层数增加到 6 层之后。我们使用 SRU 替换第二个基线系统 RNNsearch* 中的 GRU，然后以同样的堆叠方式尝试增加模型的深度，观察模型性能和训练速度的变化情况，表 3 显示了基于 SRU 的深层模型在德-英翻译任务上的 BLEU 结果。其中“训练速度”的单位是小时/轮，“参数量”的单位是兆，粗体表示最好模型，“†”表示在 99%的置信区间显著优于基线模型。为了保证实验对比的公平性，深层 GRU 模型和深层 SRU 模型的堆叠均不使用残差连接。

对比表 2 右边部分 RNNsearch* 与表 3 的实验结果可以发现，我们基于 SRU 的模型在增加层数之后表现出了明显的优势。只有在层数为 1 的情况下，基于 SRU 的系统性能不如基于 GRU 的系统；随着层数的增加，不管是在开发集还是在测试集上，SRU 系统的翻译质量均不断提升，而且在对应层数上都要优于基于 GRU 的系统。基于 GRU 的系统在层数超过 6 之后就会由于梯度消失导致翻译质量急剧退化，而我们的系统依然保持很好的性能并不断提升。在训练速度方面，随着层数的增加，SRU 模型的参数量也会有所增加，但是对训练速度的影响并不大，不像 GRU 系统的训练速度受到层数的很大影响。在 IWSLT 德英翻译任务上，基于 SRU 的深度模型比基线系统提升 0.94 个 BLEU 值，表现出显著性优势。

我们尝试继续加深基于 SRU 的深度模型，翻译性能的变化基本保持一个平稳的趋势，但是，由于参数量的增加，训练速度会有所下降。

我们又尝试在维汉翻译任务上验证 SRU 模型的能力。表 4 给出了基于 SRU 的深层模型在维汉翻译任务上与基线模型的 BLEU 对比结果。其中粗体表示最好模型，“*”表示在 95%的置信区间内比基线模型有显著性提高。从表 4 可以看出，深度模型随着层数的增加，翻译的质量在整体上呈现上升趋势，最好的单模型比基线模型要高 3.31 个 BLEU 值。由此可见，基于 SRU 的深度模型不仅可以有效地解决梯度消失的问题，而且增加编码器和解码器的层数稳定地提升了翻译性能，比基线系统有非常显著的提升。

表 4 维汉任务翻译结果

系统	测试集
RNNsearch*	58.02
SRU-1	57.22
SRU-2	60.62
SRU-4	60.70
SRU-6	60.71
SRU-8	60.57
SRU-10	**61.33***

5.4.3 对比深层 SRU 模型与融合残差网络的深层 GRU 模型

从表 2 的实验结果可以看出，随着模型深度的增加，基于 GRU 的模型 RNNsearch* 会出现梯度消失而退化的问题。为了解决这一问题，残差网络连接被提出，通过在一个浅层网络基础上叠加恒等映射层，将前一层的输出信息直接与当前层的输出信息相加作为下一层的输入，使得低层的信息有效地与高层建立直连，可以让模型性能随深度的增加而不退化。而基于 SRU 的深度模型是从另外一个角度去解决梯度消失的问题。我们在德英翻译任务上分别训练 4 层和 8 层的不引入和引入残差连接的深层 GRU 模型以及深度 SRU 模型，对比它们之间的性能。如表 5 所示，其中粗体表示 SRU 模型优于 GRU+Res，“†”表示在 99%的置信区间上比 GRU+Res 有显著性提升。

表 5 深度模型在德英翻译任务上的对比实验

层数	GRU	GRU+Res	SRU
4	25.90	27.42	**28.05†**
8	18.07	27.42	**28.46†**

表 5 中的实验结果表明，加入残差接连后，基于 GRU 的模型 RNNsearch* 的翻译性能相比于不加入残差连接有明显的优势，而且在使用超过 6 层的编码器和解码器时并没有出现梯度消失而导致模型退化；在使用 8 层的编码器和解码器时，性能大幅度超过不引入残差连接的模型。然而，在使用相同层数的编码器和解码器的情况下，基于 SRU 的深度模型的性能要明显优于融合残差连接的 GRU 深度模型，并有非常显著的提升。

5.5 对实验结果的分析

实验部分的结果表明，对于使用 GRU 单元的翻译模型来讲，当编码器和解码器的深度增加得少于 6 层时，基线系统 RNNsearch* 的性能会有所下降，使翻译质量微弱地变差。但是，当编码器和解码器超过 6 层之后，就会导致模型无法收敛。原因

在于 RNNsearch* 模型的解码器使用了两层的 GRUs，深度过大导致反向求导过程失败而无法更新参数。两个翻译方向上的实验结果均表明，基于 SRU 的深度模型有效地解决了这一问题，并且随着深度增加，模型性能也稳定地得以提升，显著地超过了基线模型。与 GRU 相比，SRU 保留门控机制特性的同时，计算上避免了对前一步隐状态的依赖，能够在一定程度上实现并行，所以深度的增加并没有对训练速度产生很大影响。而基于 GRU 的基线模型随着层数增加，训练速度变得非常慢。在不使用残差连接的情况下，我们模型的性能显著性地超过了引入残差连接的基线模型。

6 结论

对于基于门控循环单元的神经网络机器翻译模型而言，编码器和解码器的堆叠达到一定深度时，会出现梯度消失和模型无法收敛的问题，使得模型性能急剧恶化，译文质量大幅下降。为解决该问题，在本文中，我们将简单循环单元引入到基于注意力机制的神经网络翻译模型当中，通过堆叠的方式加深了编码器和解码器的结构。我们在 2014 年 IWSLT 德英和 2017 年 CWMT 维汉两个翻译任务上进行了实验。实验结果表明，基于 SRU 的深度模型有效地解决了层数增加到一定程度所导致的梯度消失问题，其性能不仅随着深度的增加而得到逐步地强化，而且比基线模型有非常显著性地提升。另外，当基于 GRU 的深度模型使用残差连接时，梯度消失的问题也可以在一定程度上得以缓解；在层数相同的情况下，基于 SRU 的深度模型比使用残差连接的 GRU 深度模型还要有非常显著的优势。

我们将在下一步的工作中，探索如何更好地堆叠简单循环单元，以进一步改进模型性能。另外维语端词语的切分粒度，对翻译性能的影响比较大，我们考虑在维语端基于字符进行操作。

参考文献

[1] Sutskever I, Vinyals O, Le Q V. Sequence to Sequence Learning with Neural Networks[J]. 2014, 4:3104-3112.

[2] Cho K, Merrienboer B V, Gulcehre C, et al. Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation[J]. Computer Science, 2014.

[3] Bahdanau D, Cho K, Bengio Y. Neural Machine Translation by Jointly Learning to Align and Translate[J]. Computer Science, 2014.

[4] Luong M T, Pham H, Manning C D. Effective Approaches to Attention-based Neural Machine Translation[J]. Computer Science, 2015.

[5] Shen S, Cheng Y, He Z, et al. Minimum Risk Training for Neural Machine Translation[J]. Computer Science, 2015.

[6] Tu Z, Lu Z, Liu Y, et al. Modeling Coverage for Neural Machine Translation[J]. 2016:76-85.

[7] Wang M, Lu Z, Li H, et al. Memory-enhanced Decoder for Neural Machine Translation[J]. 2016.

[8] Hochreiter S, Schmidhuber J. Long Short-Term Memory[J]. Neural Computation, 1997, 9(8):1735-1780.

[9] Gehring J, Auli M, Grangier D, et al. Convolutional Sequence to Sequence Learning[J]. 2017.

[10] He K, Zhang X, Ren S, et al. Deep Residual Learning for Image Recognition[J]. 2015:770-778.

[11] Conneau A, Schwenk H, Barrault L, et al. Very Deep Convolutional Networks for Text Classification[J]. 2016:1107-1116.

[12] Lei T, Zhang Y, Artzi Y. Training RNNs as Fast as CNNs[J]. arXiv preprint arXiv:1709.02755, 2017.

[13] 赵朋成, 冯玉田, 罗涛,等. 基于深度残差网络的手写体数字识别[J]. 工业控制计算机, 2017, 30(10):82-83.

[14] 王一宁, 秦品乐, 李传朋,等. 基于残差神经网络的图像超分辨率改进算法[J]. 计算机应用, 2017.

[15] 哈里旦木·阿布都克里木，刘洋，孙茂松. 神经机器翻译系统在维吾尔语-汉语翻译中的性能对比[J]. 清华大学学报(自然科学版), 2017(8):878-883.

[16] Koehn, Philipp, Hoang, et al. Moses: open source toolkit for statistical machine translation[J]. in Proceedings of the Association for Computational Linguistics (ACL’07, 2007, 9(1):177--180.

[17] Sennrich R, Haddow B, Birch A. Neural Machine Translation of Rare Words with Subword Units[J]. Computer Science, 2015.

[18] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. BLEU: a method for automatic evaluation of machine translation[C]// Meeting on Association for Computational Linguistics. Association for Computational Linguistics, 2002:311-318.

[19] Zeiler M D. ADADELTA: An Adaptive Learning Rate Method[J]. Computer Science, 2012.

[20] Collins M, Koehn P. Clause restructuring for statistical machine translation[C]// Meeting on Association for Computational Linguistics. Association for Computational Linguistics, 2005:531-540.