- 作者
  - Lijun Wu 中山大学计算机科学与数据中心
  - 微软亚洲研究院
- 代码仓库
  - https://github.com/apeterswu/Depth_Growing_NMT
- **1.摘要**
  - 解决问题：机器翻译任务中网络深度增加而又不会使翻译质量下降
  - 方法：两阶段的训练方式，使用三个特别设计的组件
  - baselines：strong Transformer
  - 数据集: WMT14 English->German and English->French
- **2.导论**
  - 近年来使用深度神经网络进行机器翻译的文献
  - 机器翻译中三类不同的网络架构文献
  - 通过更好的初始化参数、多阶段训练和设计新颖的模型架构解决部分训练深度网络的难题
  - 在识别、问答、文本生成任务中深度模型可以达到几十甚至超过一百的深度，而机器翻译任务中通常是6层。
  - 直接堆叠更多的层不会提升性能甚至导致优化失败
  - 历史上在构建深度NMT模型上的尝试，然而有缺点
  - 本文提出：使用两阶段训练策略，直接在训练好的NMT模型上使用三个设计的组件克服优化难题，并且利用了深浅两个架构的能力
  - 本文的方法是通用的
  - 实证研究显示：在En->Ge/En->Fr数据集上相比于strong transformer，BLEU分数分别提高了1.0/0.6分

- 方法

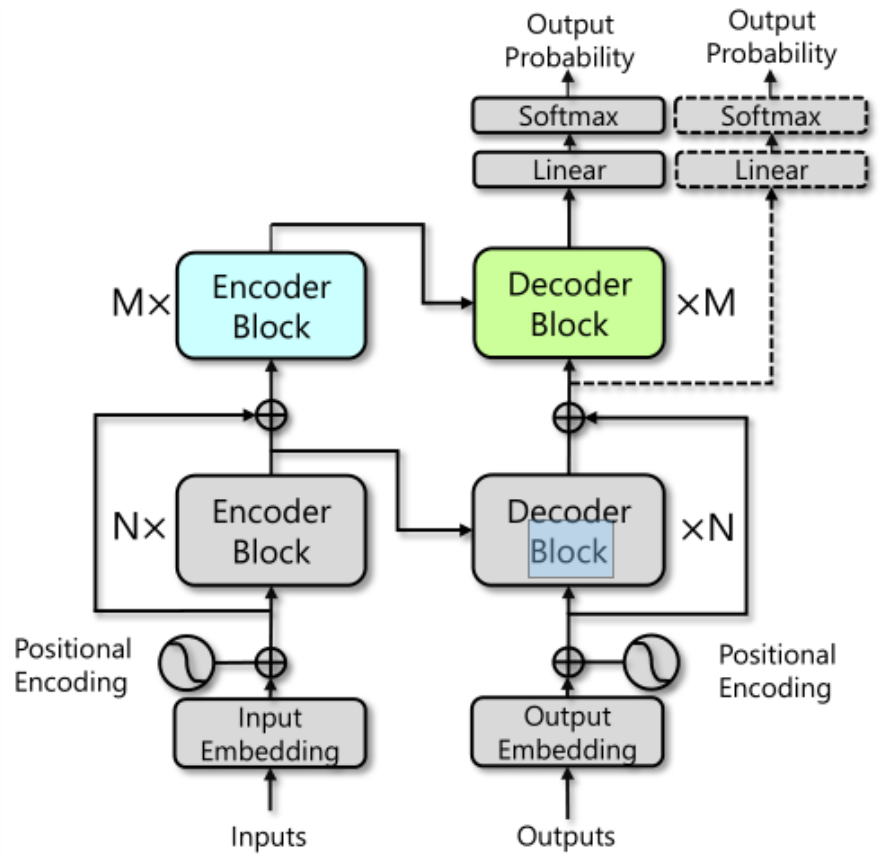

Figure 2: The overall framework of our proposed deep model architecture. N and M are the numbers of blocks in the bottom module (i.e., grey parts) and top module (i.e., blue and green parts). Parameters of the bottom module are fixed during the top module training. The dashed parts denote the original training/decoding of the bottom module. The weights of the two linear operators before softmax are shared.

模型由N层的底层模块和M层的上层模块组成，每层都是encoder layer和decoder layer 。使用 enc_1 和 dec_1 表示底层模块的encoder和decoder；相应地，使用 enc_2 和 dec_2 表示上层模块的encoder和decoder。并且在两个decoder中均使用encoder-decoder attention机制，分别用 $attn_1$ 和 $attn_2$ 表示。

模型通过两阶段训练构建，在stage 1中，底层模块（即 enc_1 和 dec_1 ）训练完成；在stage 2中，冻结底层模块，仅仅训练上层模块（即 enc_2 和 dec_2 ）。

使用 x 和 y 代表源序列和目标序列的embedding， l_y 表示 y 中单词的数量， $y_{<t}$ 表示时间步 t 之前的词语。使用下列方式构建模型：

$$h_1 = enc_1(x); h_2 = enc_2(x + h_1); \quad (1)$$

$$s_{1,t} = dec_1(y_{<t}, attn_1(h_1)), \forall t \in [l_y]); \quad (2)$$

$$s_{2,t} = dec_2(y_{<t} + s_{1,<t}, attn_2(h_2)), \forall t \in [l_y]; \quad (3)$$

这是为深度模型构建的三个组件：

- **(1) 跨模块的残差链接**

encoder部分使用方程 **(1)** 底层模块 enc_1 将输入embedding x 编码成隐层表示 h_1，然后将跨模块的残差连接引入，最终生成 enc_2 的输出 h_2。

decoder部分使用 **(2)** 和 **(3)** 进行残差连接。

这样使得上层模块直接获得word embedding的信息，同时也能获得由底层模块生成的高阶信息。

- (2)**层级encoder-decoder attention**

使用 **(2)** 和 **(3)** 计算分层级的attention。其中，在底层模块中，使用 h_1 作为 $attn_1$ 的key和value；在上层模块中，使用 h_2 作为 $attn_2$ 的kay和value。同时在 $attn_1$ 和 $attn_2$ 中使用相应的解码层的隐藏层状态作为解码模块的quaries。

这种方式使得底层模块的能力得到保存，同时利用了更高阶的上下文表示。

- (3)**Deep-shallow 解码**

在解码阶段，使用 **(1)** 和 **(2)** 作为浅模型 net_S，使用 **(1) (2) (3)** 作为整体模型 net_D。 net_S 和 net_D 通过reranking共同生成最终的翻译结果。

- **Discussion**
    - 训练复杂度：分阶段训练缓解了优化难度，也降低了训练复杂度
    - 集成学习：我们提出的是带有不同层级上下文信息的单个deeper模型，这和这和推理复杂度的集成方法相似，但不一样。

- **3. 实验**

- **3.1 实验设计**

数据集

- 训练集
    - WMT14 en-de 4.5M sentence pairs
    - WMT14 en-fr 36M sentence pairs
- 验证集
    - Newstest2012 and newstest2013
- 测试集
    - Newstest2014

单词使用BPE编码， 源语言和目标语言共享词典（32k tokens for en-de, 45k tokens for en-fr）

架构

使用Transformer作为基本的encoder-decoder框架。采用Vaswani et al. (2017) big transformer的超参数配置，word embedding, hidden states, non-linear layer 的维度分别为1024， 1024，4096.

dropout rate = 0.3 for en->de

deopout rate = 0.1 for en->fr

number of encoder/decoder blocks for bottom module = N = 6

number of additional blocks for top module = M = 2

代码基于PyTorch版的Transformer

训练

使用Adam优化器，学习率调度采用Vaswani et al.(2017)中的默认设置。

所有模型在8张M40 GPU上训练。

评估

Table 1: The test set performances of WMT14 En→De and En→Fr translation tasks. '†' denotes the performance figures reported in the previous works.

Model	En→De	En→Fr
Transformer (6B)†	28.40	41.80
Transformer (6B)	28.91	42.69
Transformer (8B)	28.75	42.63
Transformer (10B)	28.63	42.73
Transparent Attn (16B)†	28.04	–
Ours (8B)	**29.92**	**43.27**

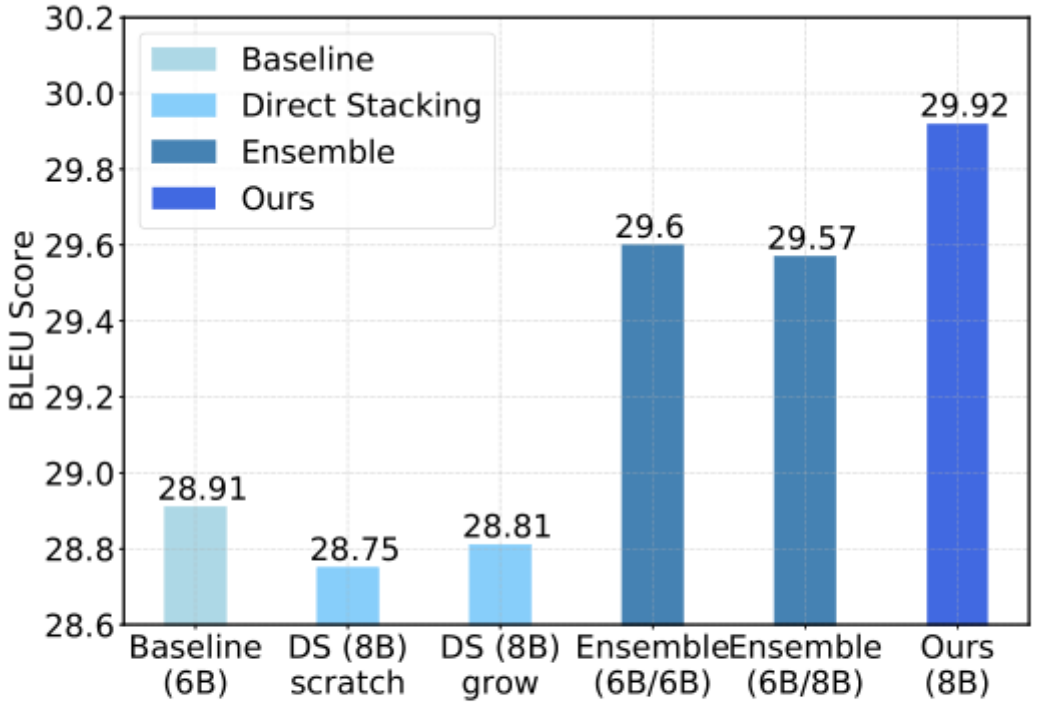

Figure 3: The test performances of WMT14 En→De translation task.

使用tokenized case-sensitive BLEU评分(Papineni et al., 2002)

beam size = 5 with no length penalty.