Transformer及其变体

Vanilla Transformer

Vanilla Transformer: Why

字符级语言建模：

- 模型需要从头学习大量词汇
- 长期依赖，前后文本可能依赖数百或者数千个时间步长
- 字符序列比单词序列长，需要更多计算

1. LSTM和其他RNN变体在字符级语言建模上已经表现出强大的性能。（使用截断的反向传播来进行训练）

How neural language models use context (ACL): a word-based LSTM language model only effectively uses around 200 tokens of context (even if more is provided), and that word order only has an effect within approximately the last 50 tokens.

2. 本文使用Transformer在字符级语言建模上达到了最佳性能。该模型在常用基准测试（enwik8 和 text8）上的表现优于 RNN 模型。

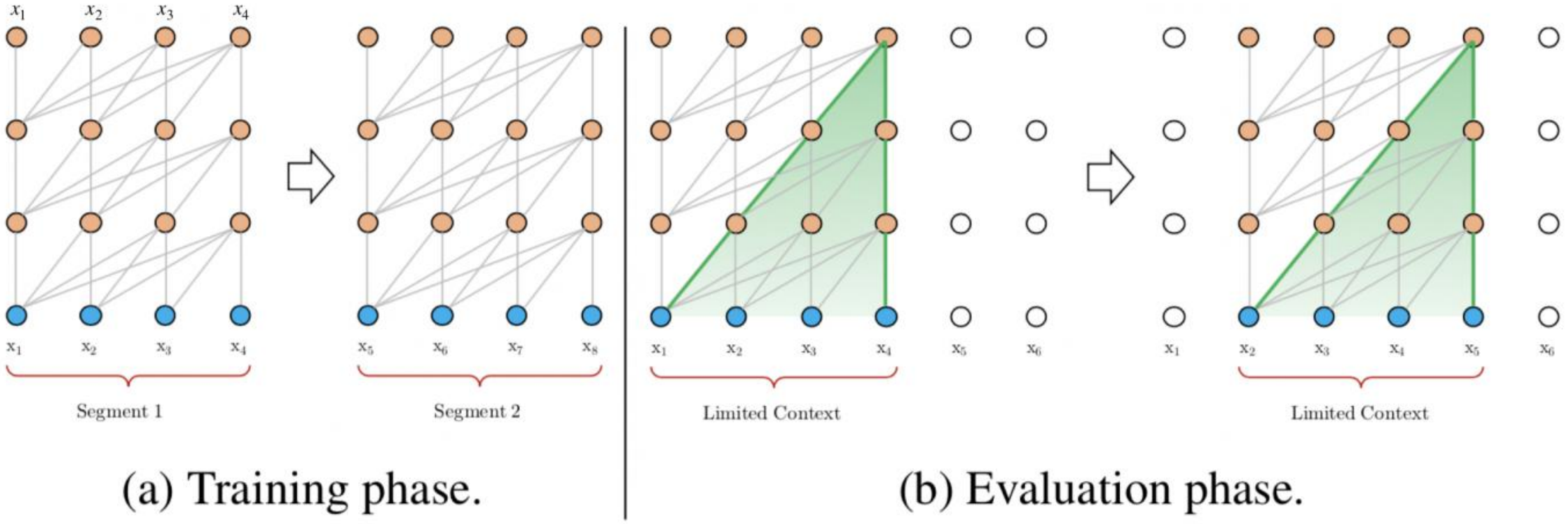
Vanilla Transformer: What

本文采用方法的特点：

- Transformer（64层）（处理 512 个字符）
- 增加辅助loss进行训练（loss (i) intermediate sequence positions (ii) intermediate network layers (iii) at target positions multiple steps in the future ），从而加快收敛速度，并且可以训练更深的网络。
- 位置向量
- 非递归
- 固定上下文
- causal attention

所以类似采用Transformer的Decoder结构。

这种 64 层变换器模型仅限于处理 512 个字符这种相对较短的输入，因此它将输入分成段，并分别从每个段中学习。如果在测试阶段需要处理较长的输入，该架构会在每一步中将输入向右移动一个字符，以此实现对单个字符的预测。



(a) Training phase.

(b) Evaluation phase.

Vanilla Transformer: What 1. Transformer

$$\Pr(t_{0:L}) = P(t_0) \prod_{i=1}^{L} \Pr(t_i | t_{0:i-1})$$

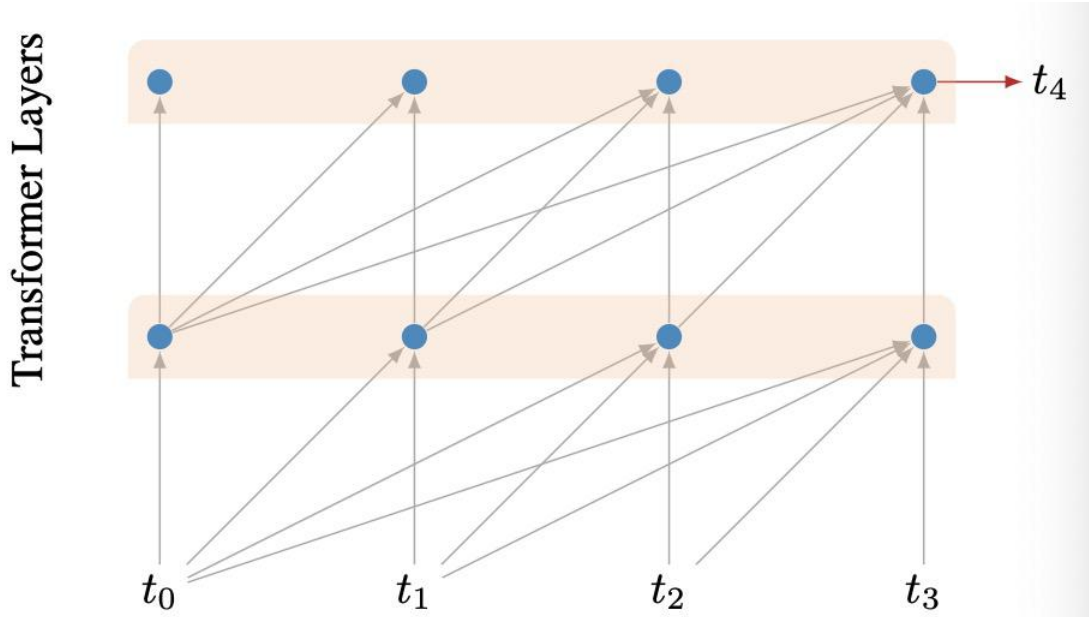


Figure 1: Character transformer network of two layers processing a four character sequence to predict t_4. The causal attention mask limits information to left-to-right flow. Red arrows highlight the prediction task the network has to learn.

Vanilla Transformer: What 2. 辅助loss

增加辅助损失使得可以训练更深的Transformer。（论文提出时是最深的Transformer模型 64）

辅助损失：

- intermediate positions
- intermediate layers（中间层的辅助损失逐步丢弃）
- nonadjacent targets（预测下下个字符损失减半，0.5权重）

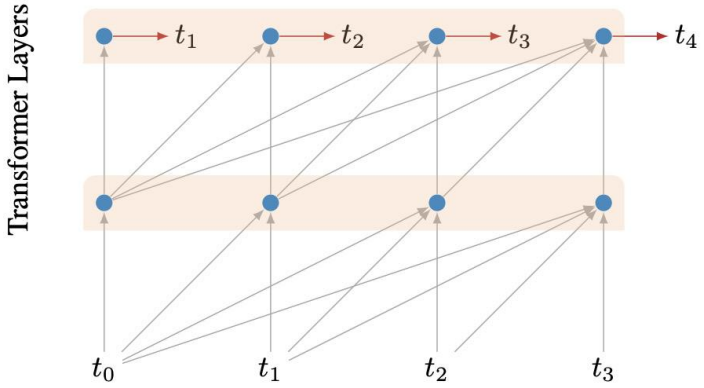


Figure 2: Adding the intermediate positions prediction tasks to our network. Now, we predict the final character t_4 and all intermediate characters $t_{0:3}$. t_3 has access only to $t_{0:2}$ because of the causal attention masks. All of these losses contribute equally during training.

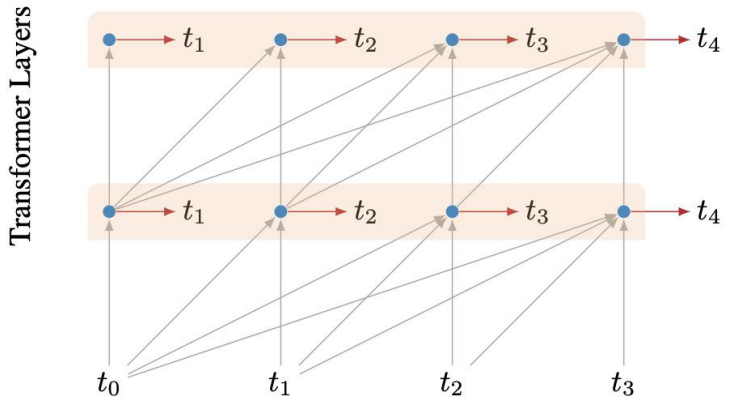


Figure 3: Our network after adding prediction tasks for the intermediate layers. For this example of two layers, the losses of the intermediate layer prediction tasks will be absent after finishing 25% of the training.

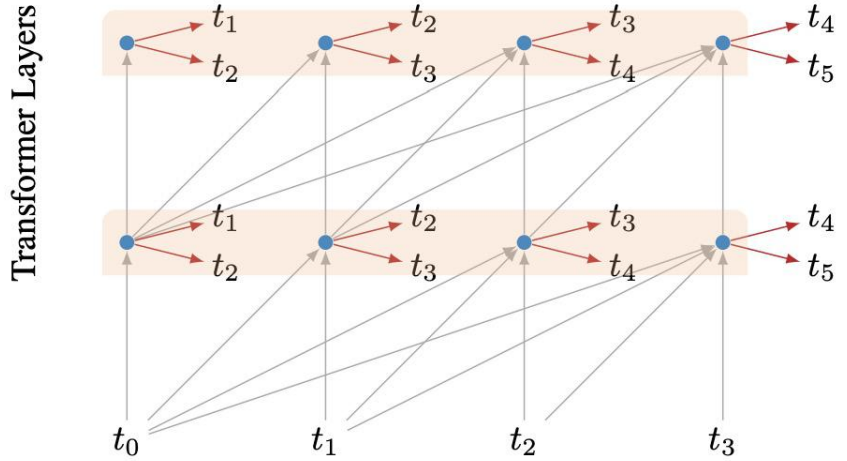


Figure 4: Our example network after adding two predictions per position.

Vanilla Transformer: What
3. 位置向量

假设时序信息可能会在通过层传播的过程中丢失，所以每层需要学习位置向量。

假设L个位置、N层、向量512维，参数量为 L * N * 512。（预测时也限制在长度L内）

Transformer XL

Transformer XL: Why

RNN的缺陷：
- 受梯度消失问题的影响，RNN 往往速度很慢，且其学习长期依赖的能力比较有限。

Transformer的缺陷：
- 无法计算超过固定长度的依赖关系。
- 序列截断后，造成段落边界破碎，从而造成低效优化，即使是短序列这也是严重问题。

Vanilla Transformer的缺陷：
- 上下文相关性有限。字符之间的最大依赖距离受输入长度的限制。例如，该模型不能“使用”出现在几个句子之前的单词。
- 上下文破碎。对于长度超过 512 个字符的文本，其每个段都是从头开始单独训练的。因此，对于每个段的第一个表征以及各个段之间，根本不存在上下文（依赖性）。这会使得训练效率低下，并会影响模型的性能。
- 推理速度慢。在测试阶段，每次预测下一个单词，都需要重新构建一遍上下文，并从头开始计算，这样的计算速度非常慢。

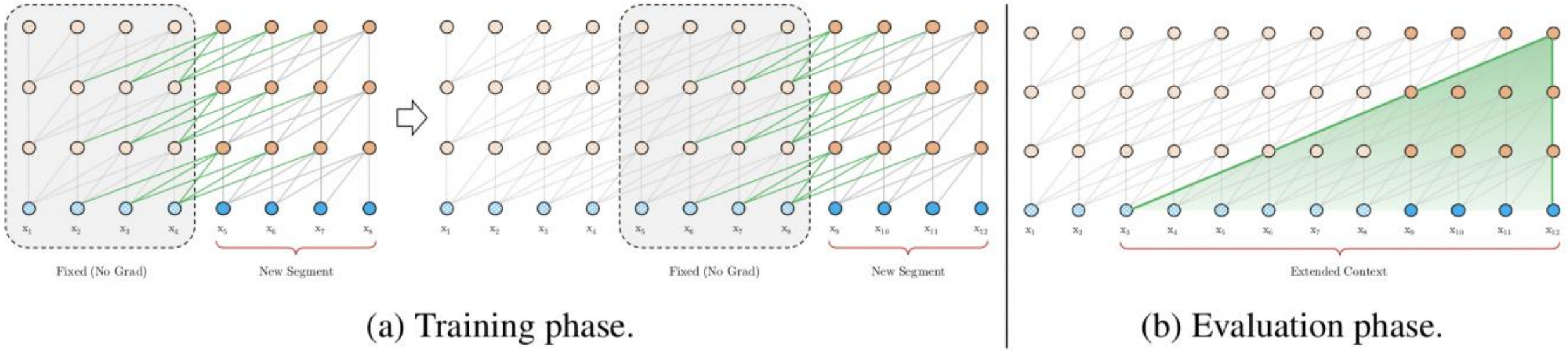
Transformer-XL 在多种语言建模数据集（如单词级别的 enwik8 和字符级别的 text8）上实现了最目前先进的结果，且该模型在推理阶段速度更快，比之前最先进的 Transformer 架构快 300 到 1800 倍。这是第一个在字符级和单词级建模方面比RNN结果更好的自注意力模型。

Transformer XL: What

本文采用方法的特点：

- Adaptive Softmax and Input Representations
- 循环机制：Segment-level Recurrence
- 相对位置编码： Relative Positional Encodings: Instead of having a single embedding represent each absolute position, the Transformer XL computes an embedding that represents the distance between any two tokens. This is used to compute the attention between the two words.

Transformer XL: What
1. 循环机制



(a) Training phase. (b) Evaluation phase.

处理每段文本时，每个隐藏层都会接收两个输入，这两个输入会被拼接；分别是：

- 该段的前一个隐藏层的输出，和 vanilla Transformer 相同。
- 上一个段的隐藏层的输出，可以使模型创建长期依赖关系。

该概念可以扩展到更长的依赖上。使用相同的方法，利用前面多个段的信息，只要 GPU 内存允许，在测试阶段也可以获得更长的依赖。

好处：

- 获得依赖，使用先前段的数据来预测当前段的表征。
- 测试速度快。预测阶段可以重复利用之前计算结果。

Transformer XL: What
2. 相对位置编码

Before：

$$A_{i,j}^{abs} = E_{x_i}^T W_q^T W_k E_{x_j} + E_{x_i}^T W_q^T W_k U_j + U_i^T W_q^T W_k E_{x_j} + U_i^T W_q^T W_k U_j$$

After：

$$A_{i,j}^{abs} = E_{x_i}^T W_q^T W_{k,E} E_{x_j} + E_{x_i}^T W_q^T W_{k,R} R_{i-j} + \mathbf{u}^T W_{k,E} E_{x_j} + \mathbf{v}^T W_{kR} R_{i-j}$$

- $R \in \mathbb{R}^{L_{max} \times d}$ 是与 U 对应的正弦曲线相对位置矩阵；
- $\mathbf{u}, \mathbf{v} \in \mathbb{R}^d$ 是可训练参数，分别限制和主导随 Query 位置变化导致的注意力偏差；
- $W_{k,E}$ 和 $W_{k,R}$ 是对 W_k 做的分化，分别产生以内容为偏向的 Key 和 以位置为偏向的 Key；

经过调整，新表达式的四个要素分别代表纯内容的点积，以内容为主的位置偏差，全局内容偏差，以及全局位置偏差。

Universal Transformer

Universal Transformer: Why

起因：

- RNN model耗时
- 一些前馈或者卷积神经网络（如Transformer）在序列建模上取得了出色的结果（神经翻译），并且易于并行，感受野涉及全部输入；但是当输入超过训练时观察到的长度时，对于一些简单的任务甚至无法处理。

UT：

- a parallel-in-time recurrent self-attentive sequence model
- combine the parallelizability and global receptive field of feed-forward sequence models like the Transformer with the recurrent inductive bias of RNNs
- recurrent transition function (The recurrent transition function in fact controls how steps communicate with each other in depth. For instance, the recurrent transition, can be a simple identity function which passes the output of a step as the input to next step. Or it can be an LSTM (flipped vertically) next to the transformer which controls how state of the model changes in depth.)

BERT

BERT: Why

BERT: What

训练：下面两者是独立进行的

- 预训练：为了在输入的词向量中融入上下文特征
- 微调：为了使 BERT 能适应不同的下游任务

向量：

- 词向量
- 位置向量
- Segment向量

预训练任务：

- Masked Token Prediction
- Next Sentence Prediction

XLNet

XLNet: Why

XLNet: What

- 有序因子排列
- 双流自注意力

参考

Transformer：

- 图解Transformer：https://blog.csdn.net/qq_41664845/article/details/84969266
- http://nlp.seas.harvard.edu/2018/04/03/attention.html
- https://github.com/leonardoaraujosantos/Transformers/blob/master/Vanila_Transformer.ipynb
- http://mlexplained.com/2017/12/29/attention-is-all-you-need-explained/
- https://jalammar.github.io/illustrated-transformer/

Vanilla Transformer：

- http://www.bdpt.net/cn/2019/08/16/%E6%B7%B1%E5%BA%A6%E5%AD%A6%E4%B9%A0%EF%BC%9A%E5%89%8D%E6%B2%BF%E6%8A%80%E6%9C%AF-vanilla-transformer/

Transformer xl：

- https://mlexplained.com/2019/07/04/building-the-transformer-xl-from-scratch/
- https://www.lyrn.ai/2019/01/16/transformer-xl-sota-language-model/
- Transformer-XL: Unleashing the Potential of Attention Model: https://segmentfault.com/a/1190000018141529
- https://www.cnblogs.com/huangyc/p/11445150.html