深度之眼
deepshare.net

Transformer-xl

导师：Yamada

《Transformer-XL: Attentive Language Models Beyond a Fixed-Length Context》

超出固定长度上下文的注意力语言模型

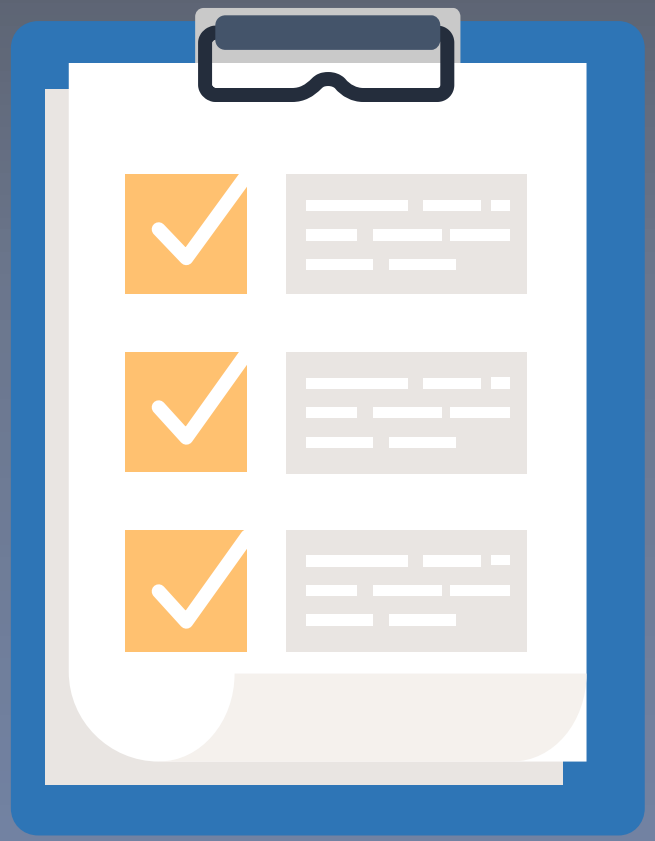
作者：Zhilin Yang、Zihang Dai

单位：google brain && cmu

发表会议及时间：ACL，2019

上节回顾

Review in the previous lesson



01 Transformer-xl背景介绍

介绍了语言模型数据集、翻译衡量指标ppl和bpc、transformer-xl的历史意义等等

02 论文泛读

快速浏览整篇论文、介绍了论文的摘要以及各个小标题并且分析了一下论文的实验结果

03 Transformer以及Vanilla Model回顾

回顾了Self-attention的流程以及对Vallina Model的几种loss

深度之眼
deepshare.net

深度之眼
deepshare.net

第二课：论文精读

The second lesson: the paper in detail

深度之眼
deepshare.net

目 录

1 Vallina Transformer

2 片段级递归机制

3 Relative Positional Encodings

4 小trick

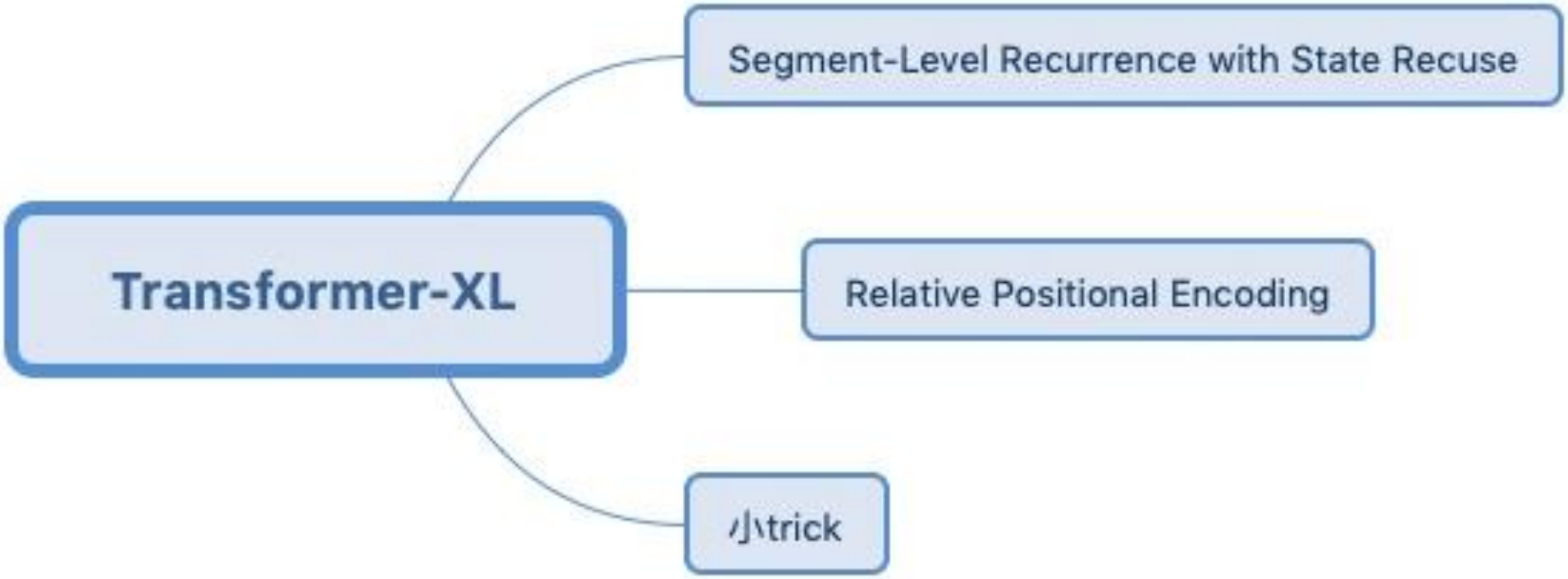
5 实验设置和结果分析

6 论文总结

7 本课回顾及下节预告

知识树

深度之眼
deepshare.net



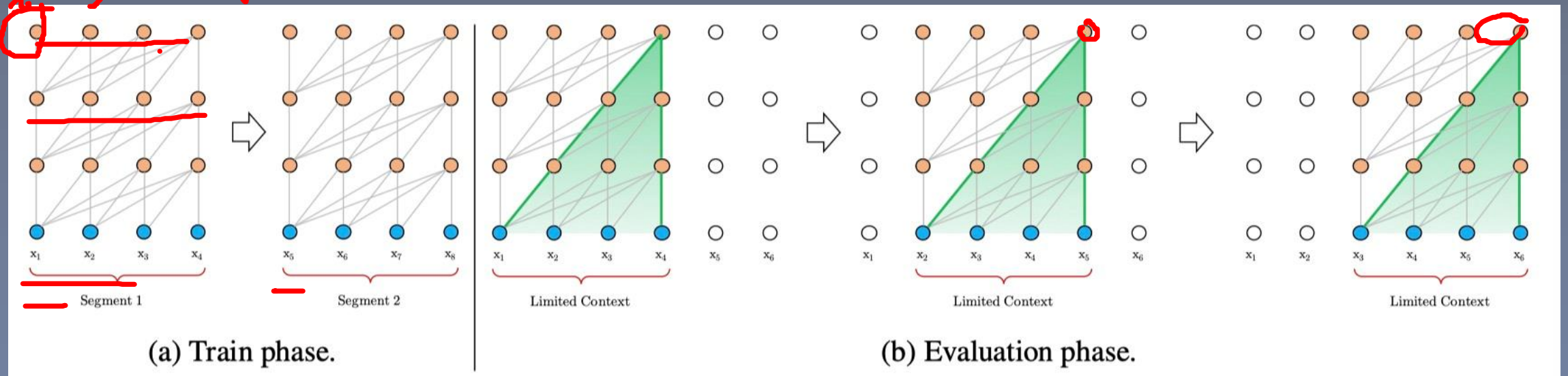
深度之眼
deepshare.net

Vallina Transformer

Vallina Transformer

Architecture of VallinaTransformer

Vallina Transformer的train阶段和evalue阶段



(a) Train phase.

(b) Evaluation phase.

Figure 1: Illustration of the vanilla model with a segment length 4.

Vallina Transformer

Architecture of VallinaTransformer

深度之眼
deepshare.net

Vallina Transformer存在以下问题:

(1) Vallina model不同的segment之间没有上下文联系，会导致训练效率低下，也会导致模型训练的结果不理想。

(2) Vallina model不同segement对应的相同的位置如果采用同样的positonal encoding就会导致难以区分。

(3) Vallina model在预测的时候每次都只能预测一个postion的概率，这样预测的效率低下。

深度之眼
deepshare.net

Segment-Level Recurrence with State Reuse

Segment-Level Recurrence with State Reuse

深度之眼
deepshare.net

$$\widetilde{h}_{\tau+1}^{n-1} = [SG(h_{\tau}^{n-1}) \cdot h_{\tau+1}^{n-1}]$$

$$q_{\tau+1}^{n}, k_{\tau+1}^{n}, v_{\tau+1}^{n} = h_{\tau+1}^{n-1} W_q, \widetilde{h}_{\tau+1}^{n-1} W_k, \widetilde{h}_{\tau+1}^{n-1} W_v$$

$$h_{\tau+1}^{n} = Transformer - Layer(q_{\tau+1}^{n}, k_{\tau+1}^{n}, v_{\tau+1}^{n})$$

Segment-Level Recurrence with State Reuse

深度之眼
deepshare.net

其中绿色部分为segemnt-level recurrence机制，实际使用时的缓存的segemnt可以是不仅限于上一状态。

rnn的lm的recurrence dependency在同一层layer之间，而本模型可以在不同layer之间，因此depency length为O(N*L)

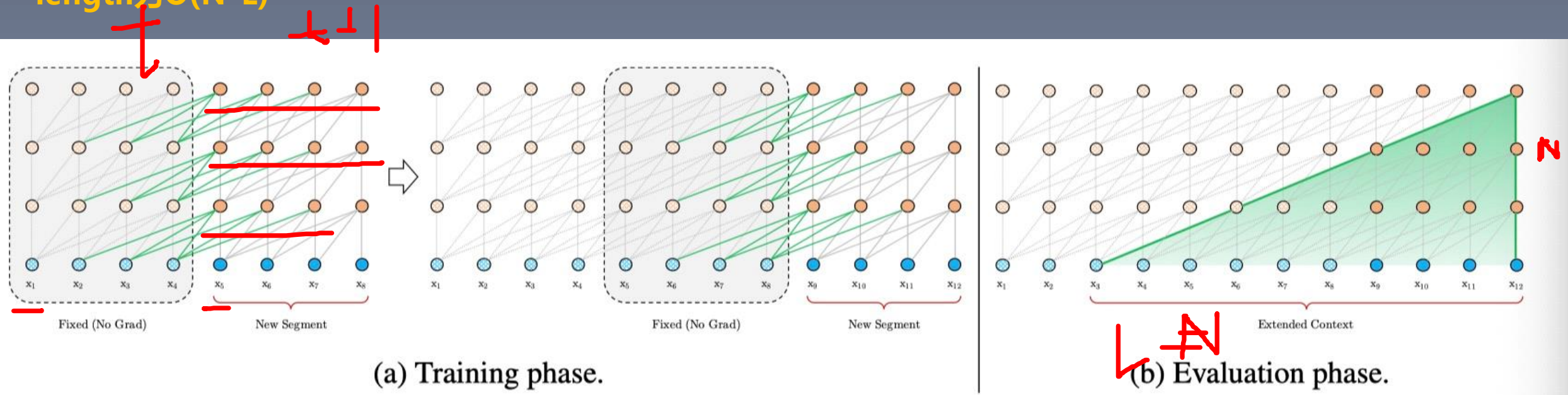
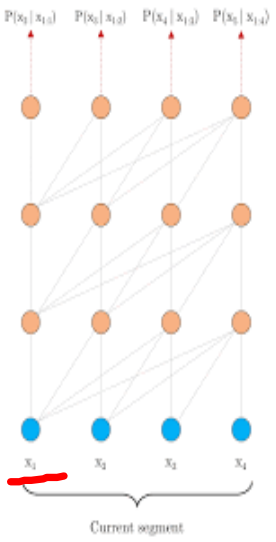
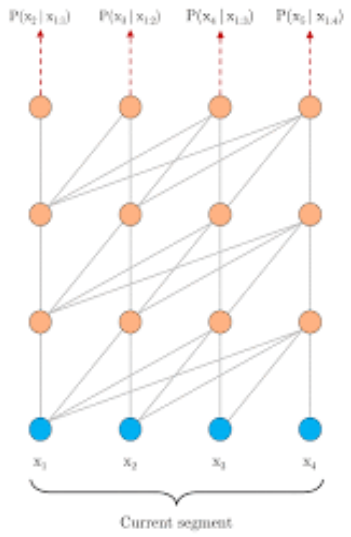


Figure 2: Illustration of the Transformer-XL model with a segment length 4.

Segment-Level Recurrence with State Reuse



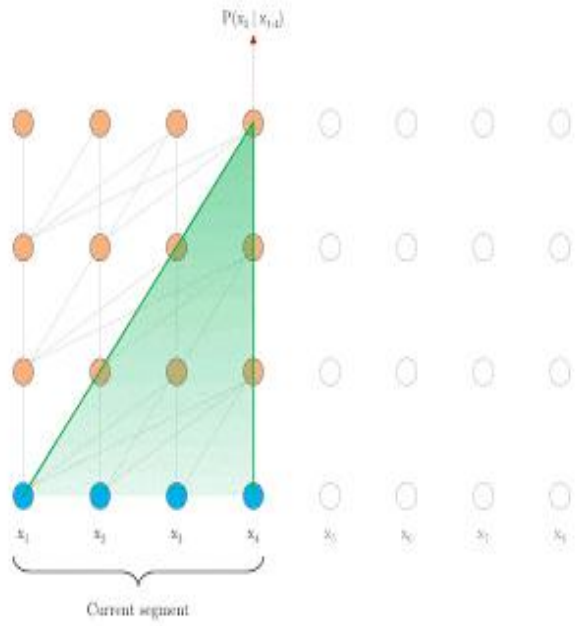


Vallina Model的训练阶段

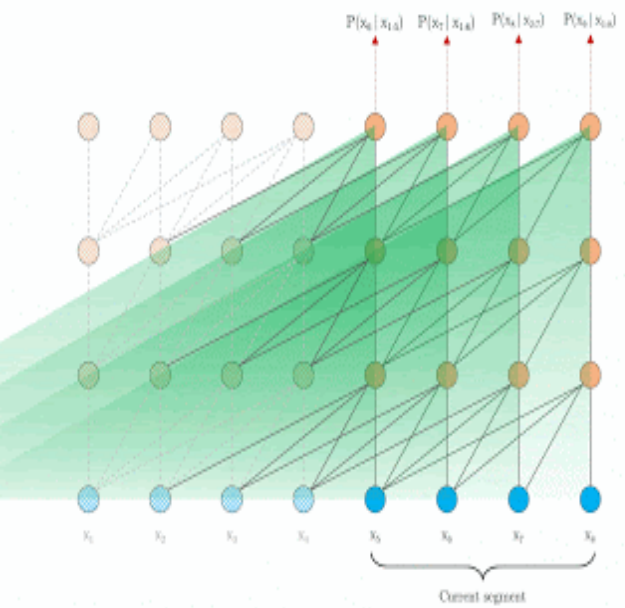
Transformer-xl的训练阶段

Segment-Level Recurrence with State Reuse

深度之眼
deepshare.net



Vallina Model的预测阶段



Transformer-xl的预测阶段

深度之眼
deepshare.net

Relative Positional Encodings

Relative Positional Encodings

深度之眼
deepshare.net

$$A_{i,j}^{abs} = \underbrace{\mathbf{E_{x_i}}^\top \mathbf{W_q}^\top W_k E_{x_j}}_{(a)} + \underbrace{\mathbf{E_{x_i}}^\top \mathbf{W_q}^\top W_k U_j}_{(b)} + \underbrace{\mathbf{U_i}^\top \mathbf{W_q}^\top W_k E_{x_j}}_{(c)} + \underbrace{\mathbf{U_i}^\top \mathbf{W_q}^\top W_k U_j}_{(d)}$$

式子1

$$A_{i,j}^{rel} = \underbrace{\mathbf{E_{x_i}}^\top \mathbf{W_q}^\top W_k E_{x_j}}_{(a)} + \underbrace{\mathbf{E_{x_i}}^\top \mathbf{W_q}^\top W_k R_{i-j}}_{(b)} + \underbrace{\mathbf{u}^\top W_{k,E} E_{x_j}}_{(c)} + \underbrace{\mathbf{v}^\top W_{k,R} R_{i-j}}_{(d)}$$

式子2

$$A_{i,j}^{abs} = \mathbf{q_i}^\top * k_j = (\mathbf{W_q}(\mathbf{E_{x_i}} + \mathbf{U_i}))^\top W_k(E_{x_j} + U_j))$$

式子3

Relative Positional Encodings

深度之眼 deepshare.net

Relative Positional Encodings

式子1和2的区别：

1）b和d两项中，绝对位置向量U_j转换成相对位置向量R_{i-j}

2) 在c项中，将查询的$U_i^T W_q^T$向量转换成一个需要学习的参数向量u,因为在考虑相对位置的时候，不需要考虑绝对位置i，d项中也是如此。

3) 将key的权重变换矩阵W_k转为$W_{k,E}$和$W_{k,R}$为content-based key vectors和location-based key vectors.

从另一个角度理解：

1）a 基于内容的”寻址”,即没有添加原始位置编码的原始分数

2) b 基于内容的位置偏置，即相对于当前位置的位置偏差

3) c 全局的内容偏置，用于衡量key的重要性

4）d 全局的位置偏置，根据query和key之前的距离调整重要性

Relative Positional Encodings

深度之眼
deepshare.net

式子4 $$\widetilde{h}_{\tau}^{n-1} = [SG(m_{\tau}^{n-1}) \cdot h_{\tau}^{n-1}]$$

式子5 $$q_{\tau}^{n}, k_{\tau}^{n}, v_{\tau}^{n} = h_{\tau}^{n-1} W_{q}^{n}, \widetilde{h}_{\tau}^{n-1} W_{k,E}^{n}, \widetilde{h}_{\tau}^{n-1} W_{v}^{n}$$

式子6 $$A_{\tau,i,j}^{n} = \mathbf{q}_{\tau,\mathbf{i}}^{\mathbf{n}\top} k_{\tau,j}^{b} + \mathbf{q}_{\tau,\mathbf{i}}^{\mathbf{n}\top} \boxed{W_{k,R}^{n} R_{i-j}} + \mu^{\top} k_{\tau,j} + \mathbf{v}^{\top} \boxed{W_{k,R}^{n} R_{i-j}}$$

式子7 $$a_{\tau}^{n} = Masked - Softmax(A_{\tau}^{n}) v_{\tau}^{n}$$

式子8 $$o_{\tau}^{n} = LayerNorm(Linear(a_{\tau}^{n}) + h_{\tau}^{n-1})$$

式子9 $$h_{\tau}^{n} = Positionalwise - Feed - Forward(o_{\tau}^{n})$$

Relative Positional Encodings

深度之眼
deepshare.net

总结

1）式子4表示的是recurrence结构，式子5表示的是计算q、k、v矩阵

2）式子6和式子6表示的就是self-attention结构，其中式子6表示的是q*k，式子7表示的softmax(q*k)*v，其中式子5是将式子2中的$E_{xi}W_q$重新转换为$q_{\tau,i}^n$，$W_{k,E}E_{xj}$重新转换为$k_{\tau,j}^n$

3）式子8表示的LayerNormalization

4）式子9表示的FeedForward-Network

深度之眼
deepshare.net

小trick

小trick

深度之眼
deepshare.net

M L

在计算$W_{k,R}R_{i-j}$可以进行矩阵操作使得计算更方便：

设M和L分别为memory和当前序列段的长度，则i-j的范围也就为0~M+L-1。下面的Q矩阵中的每一行都代表着$W_{k,R}R_{i-j}$中的一个i-j的可能性。

$$Q := \begin{bmatrix} \mathbf{R_{M+L-1}}^\top \\ \mathbf{R_{M+L-2}}^\top \\ \cdots\cdots \\ \mathbf{R_0}^\top \\ \mathbf{R_1}^\top \end{bmatrix} \mathbf{W_{k,R}}^\top = \begin{bmatrix} \mathbf{W_{k,R}R_{M+L-1}}^\top \\ \mathbf{W_{k,R}R_{M+L-2}}^\top \\ \cdots\cdots \\ \mathbf{W_{k,R}R_0}^\top \\ \mathbf{W_{k,R}R_1}^\top \end{bmatrix} \in R^{(M+L)*d}$$

小trick

$L \cdot (M+L) = L \cdot M + L \cdot L$

在对于式子6中的$q_i^T W_{k,R} R_{i-}$,其构成的所有可能向量矩阵为B矩阵，其形状为L*(M+L)。

$$B = \begin{bmatrix} \mathbf{q_0}^\top W_{k,R} R_M & \dots & \mathbf{q_0}^\top W_{k,R} R_0 & 0 & \dots & 0 \\ \mathbf{q_1}^\top W_{k,R} R_{M+1} & \dots & \mathbf{q_1}^\top W_{k,R} R_1 & \mathbf{q_1}^\top W_{k,R} R_0 & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots \\ \mathbf{q_{L-1}}^\top W_{k,R} R_{M+L-1} \dots & & \mathbf{q_{L-1}}^\top W_{k,R} R_{M+L-1} & \mathbf{q_{L-1}}^\top W_{k,R} R_{L-1} \dots & & \mathbf{q_{L-1}}^\top W_{k,R} R_0 \end{bmatrix}$$

$M+L$

$$= \begin{bmatrix} \mathbf{q_0}^\top Q_{L-1} \dots \mathbf{q_0}^\top Q_{M+L-1} & 0 & \dots & 0 \\ \mathbf{q_1}^\top Q_{L-2} \dots \mathbf{q_1}^\top Q_{M+L-2} & \mathbf{q_1}^\top Q_{M+L-1} \dots & & 0 \\ \dots \quad \dots & \dots & \dots & \dots \\ \mathbf{q_{L-1}}^\top Q_0 \dots \quad \mathbf{q_{L-1}}^\top Q_M & \mathbf{q_{L-1}}^\top Q_{M+1} \dots & & \mathbf{q_{L-1}}^\top Q_{M+L-1} \end{bmatrix}$$

小trick

深度之眼 deepshare.net

进一步定义矩阵如下：

$$\widetilde{B} = q\mathbf{Q}^\top = \begin{bmatrix} \mathbf{q_0}^\top Q_0 & \dots & \mathbf{q_0}^\top Q_M & \mathbf{q_0}^\top Q_{M+1} & \dots & \mathbf{q_0}^\top Q_{M+L-1} \\ \mathbf{q_1}^\top Q_0 & \dots & \mathbf{q_1}^\top Q_M & \mathbf{q_1}^\top Q_{M+1} & \dots & \mathbf{q_1}^\top Q_{M+L-1} \\ \dots & \dots & \dots & \dots & \dots & \dots \\ \mathbf{q_{L-1}}^\top Q_0 \dots & \mathbf{q_{L-1}}^\top Q_M & \mathbf{q_{L-1}}^\top Q_{M+1} \dots & \mathbf{q_{L-1}}^\top Q_{M+L-1} \end{bmatrix}$$

前面的矩阵B的每一行是由上面的矩阵向左shift下，可以直接进行矩阵操作。

设R_i**的纬度为**d_R**,**q_i的纬度为d_q，$W_{k,R}$矩阵的纬度为$d_q * d_R$

矩阵B的时间复杂度为：$2*d_q*d_R*L*(M+L)$

上述矩阵的时间复杂度为: $L*d_{\mathbf{q}}*(M+L)+d_R*d_q*(M+L)$

小trick

深度之眼
deepshare.net

对于式子6中的$v^T W_{k,R} R_{i-}$定义矩阵D的纬度**为**$L*(M+L)$

$$D = \begin{bmatrix} \mathbf{v}^\top Q_{L-1} \ldots \mathbf{v}^\top Q_{M+L-1} & 0 & \ldots & 0 \\ \mathbf{v}^\top Q_{L-2} \ldots \mathbf{v}^\top Q_{M+L-2} & \mathbf{v}^\top Q_{M+L-1} \ldots & & 0 \\ \ldots \quad \ldots \quad \ldots & \ldots & \ldots & \ldots \\ \mathbf{v}^\top Q_0 \ \ldots \ \mathbf{v}^\top Q_M & \mathbf{v}^\top Q_{M+1} & \ldots \mathbf{v}^\top Q_{M+L-1} & \end{bmatrix}$$

可以用以下的矩阵进行shift得到

$$\widetilde{d} = [\mathbf{Q}\mathbf{v}]^\top = [\mathbf{v}^\top Q_0 \ldots \mathbf{v}^\top Q_M \ \mathbf{v}^\top Q_{M+1} \ldots \mathbf{v}^\top Q_{M+L-1}]$$

深度之眼
deepshare.net

实验设置和结果分析

Experiment results

实验结果及分析

Research Results

Model	#Param	PPL
Grave et al. (2016b) - LSTM	-	48.7
Bai et al. (2018) - TCN	-	45.2
Dauphin et al. (2016) - GCNN-8	-	44.9
Grave et al. (2016b) - LSTM + Neural cache	-	40.8
Dauphin et al. (2016) - GCNN-14	-	37.2
Merity et al. (2018) - QRNN	151M	33.0
Rae et al. (2018) - Hebbian + Cache	-	29.9
Ours - Transformer-XL Standard	151M	**24.0**
Baevski and Auli (2018) - Adaptive Input◇	247M	20.5
Ours - Transformer-XL Large	257M	**18.3**

Model	#Param	bpc
Ha et al. (2016) - LN HyperNetworks	27M	1.34
Chung et al. (2016) - LN HM-LSTM	35M	1.32
Zilly et al. (2016) - RHN	46M	1.27
Mujika et al. (2017) - FS-LSTM-4	47M	1.25
Krause et al. (2016) - Large mLSTM	46M	1.24
Knol (2017) - cmix v13	-	1.23
Al-Rfou et al. (2018) - 12L Transformer	44M	1.11
Ours - 12L Transformer-XL	41M	**1.06**
Al-Rfou et al. (2018) - 64L Transformer	235M	1.06
Ours - 18L Transformer-XL	88M	1.03
Ours - 24L Transformer-XL	277M	**0.99**

1 在WikiText-103数据集上的ppl为18.3，start-of-the-art为20.5。

2 在enwik8数据集上的bpc为1.06，start-of-the-art为1.11，12层layer的效果达到了start-of-the-art的64层的效果并且只用到了后者17%的参数。

深度之眼
deepshare.net

实验结果及分析

Research Results

Model	#Param	bpc
Cooijmans et al. (2016) - BN-LSTM	-	1.36
Chung et al. (2016) - LN HM-LSTM	35M	1.29
Zilly et al. (2016) - RHN	45M	1.27
Krause et al. (2016) - Large mLSTM	45M	1.27
Al-Rfou et al. (2018) - 12L Transformer	44M	1.18
Al-Rfou et al. (2018) - 64L Transformer	235M	1.13
Ours - 24L Transformer-XL	277M	**1.08**

Model	#Param	PPL
Shazeer et al. (2014) - Sparse Non-Negative	33B	52.9
Chelba et al. (2013) - RNN-1024 + 9 Gram	20B	51.3
Kuchaiev and Ginsburg (2017) - G-LSTM-2	-	36.0
Dauphin et al. (2016) - GCNN-14 bottleneck	-	31.9
Jozefowicz et al. (2016) - LSTM	1.8B	30.6
Jozefowicz et al. (2016) - LSTM + CNN Input	1.04B	30.0
Shazeer et al. (2017) - Low-Budget MoE	~5B	34.1
Shazeer et al. (2017) - High-Budget MoE	~5B	28.0
Shazeer et al. (2018) - Mesh Tensorflow	4.9B	24.0
Baevski and Auli (2018) - Adaptive Input$^{\diamond}$	0.46B	24.1
Baevski and Auli (2018) - Adaptive Input$^{\diamond}$	1.0B	23.7
Ours - Transformer-XL Base	0.46B	23.5
Ours - Transformer-XL Large	0.8B	**21.8**

3 在text8数据集上的bpc为1.08，start-of-the-art为1.13。

4 在One Billion Word数据集上的ppl为21.8，start-of-the art为23.7。

5 在Penn Treebank数据集上的ppl为54.5，是在没有经过fine-tuning的情况下。

Model	#Param	PPL
Inan et al. (2016) - Tied Variational LSTM	24M	73.2
Zilly et al. (2016) - Variational RHN	23M	65.4
Zoph and Le (2016) - NAS Cell	25M	64.0
Merity et al. (2017) - AWD-LSTM	24M	58.8
Pham et al. (2018) - Efficient NAS	24M	58.6
Liu et al. (2018) - Differentiable NAS	23M	56.1
Yang et al. (2017) - AWD-LSTM-MoS	22M	55.97
Melis et al. (2018) - Dropout tuning	24M	55.3
Ours - Transformer-XL	24M	**54.52**
Merity et al. (2017) - AWD-LSTM+Finetune†	24M	57.3
Yang et al. (2017) - MoS+Finetune†	22M	**54.44**

深度之眼
deepshare.net

论文总结

论文总结

Summary of the paper

深度之眼
deepshare.net

A 关键点

- 片段级递归机制
- 相对位置编码机制
- Vallina Model的存在的问题

B 小细节

- 小trick怎么实现的
- transformer-xl预测时attention的长度可以和训练时的长度不一样

论文总结

Summary of the paper

深度之眼
deepshare.net

C

启发点

- 可以在写模型的时候加入 bigram作为特征
- 可以在模型中加入类似于segment-level recurrence reuse这种缓存机制

深度之眼
deepshare.net

本课回顾及下节预告

Review in the lesson and Preview of next lesson

本节课回顾

Review in the lesson

01 Vanilla Transformer

回顾Vanilla Transformer的结构，分析该模型存在的问题，并提出Transformer-xl模型

02 Segment-Level Recurrence with State Recuse

提出片段级递归机制解决Vallina Model的segment之间联系丢失的问题，提出相对编码机制。

03 实验设置及结果分析

比较了模型在几个数据集上的表现情况，并且展示了模型在文本生成任务上的表现情况。

04 论文总结

总结论文中创新点、关键点及启发点

深度之眼
deepshare.net

下节预告

Preview of next lesson

01 搭建Transformer-xl网络代码介绍

02 介绍两种机制的实现

03 基于语言模型数据集训练Transformer-xl模型

深度之眼
deepshare.net

下节课前准备

Preview of next lesson

- 再次阅读Transformer-xl论文
- 熟悉Transformer-xl模型结构及数据预处理方式
- 配置PyTorch开发环境
- 下载Transformer-xl代码
- https://github.com/kimiyoung/transformer-xl

深度之眼
deepshare.net

结 语

循循而进，欲速则不达也。

深度之眼
deepshare.net

联系我们:
电话：18001992849
邮箱：service@deepshare.net
Q Q： 2677693114

公众号

客服微信