关于Teacher Forcing 和Exposure Bias的碎碎念

Dreamin...
实不相瞒，我是打代码的

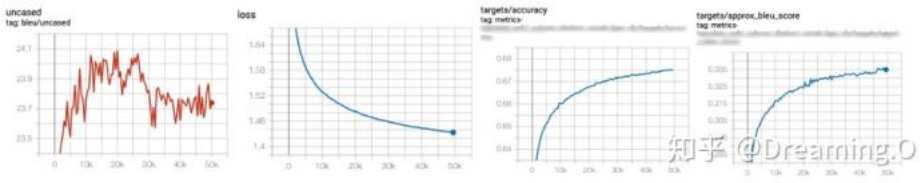
> 本文主要从实际工作过程中遭遇的Exposure bias的困境作为引子，讲述下对此的理解以及分享论文三篇。默认读者对 teacher-forcing 和 autoregressive 等概念有一定的认知。

0x01. 前言:

最近在训练 NMT 模型的过程中遇到一个问题：当基础模型训练完，利用特定语料 mixed-finetune的方式去进行 Domain-adpation，发现一些奇怪的现象。在开始阐述这些奇怪现象之初，我先对实验进行一些基础的说明：

- 测试集和训练集是一样的，由2000条样本构成。
- 重点关注指标有：
  - Inference模式下，测试集的 BLEU-Score
  - Evaluation模式下，校验集（测试集）的 loss, accuracy, approximate-bleu

所谓的奇怪现象，正是在finetune的后期，Inference模式下的 BLEU-Score 与 校验三指标（Loss、accuracy和approx-bleu）的变化方向不一致。请看下图：



实验过程中：几个重点关注指标的走势

这种现象在大规模通用语料训练基础模型的时候是较少发生的，基本上这几个指标的走势总体来看还是一致的（一致变好or变坏，虽然偶尔会有震荡）。但是在上图中可以发现： **BLEU分数在26.5K-steps后就开始发生明显的下降了，但是校验过程中的loss、approx-bleu等走势依然非常健康的向好的方向发展。**。

于是问题就变成了：**数据集中的校验 loss、appro-bleu 在持续变好，是否并不意味着最终inference过程中的 BLEU 也同步的变好呢？**

经过一番的调查后发现，这个问题下背后的真相还真别有洞天。不过在进一步探讨这个实验的之前，我们先来对一些概念温习一下。。。

0x02. Teacher Forcing 及其问题

> 在这里，我将简单的介绍下 Teacher Forcing 技术的背景，以及由此引申出来的一系列问题。

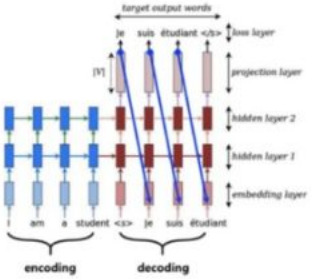
2.1 Teacher Forcing:

Teacher Forcing 是一种用于序列生成任务的训练技巧，与Autoregressive模式相对应，这里阐述下两者的区别：

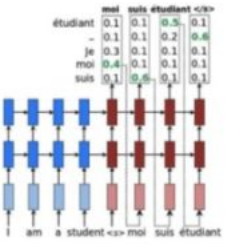
- Autoregressive 模式下，在 timesteps t decoder模块

y_{t-1} 。这时候我们称 y_{t-1} 为当前预测步的 context;
- Teacher-Forcing 模式下，在 timestep t decoder模块的输入是 Ground-truth 语句中位置的 y^*_{t-1} *单词。这时候我们称* y^*_{t-1} 为当前预测步的 context；

更具体的，我们可以看下图的例子：



Teacher-forcing技术：decoding阶段每一步的输入，都是 ground-truth-target 的 token



Autoregressive技术：decoding阶段每一步的输入，都是上一步decode预测最大概率的 token

Teacher-Forcing 技术之所以作为一种有用的训练技巧，主要是因为：

- Teacher-Forcing 能够在训练的时候矫正模型的预测，避免在序列生成的过程中误差进一步放大。
- Teacher-Forcing 能够极大的加快模型的收敛速度，令模型训练过程更加快&平稳。
- **Teacher-Forcing 技术是保证 Transformer 模型能够在训练过程中完全并行计算所有token的关键技术。**

> 如果要用比较不太严谨的比喻来说，Teacher-Forcing 技术相当于就是小明学习的时候旁边坐了一位学霸，当发现小明在做序列生成题目的时候， 每一步都把上一步的正确答案给他偷看。那么小明当然只需要顺着上一步的答案的思路，计算出这一步的结果就行了。这种做法，比起自己每一步都瞎猜， 当然能够有效的避免误差进一步放大，同时在学习前期还能通过学霸辅导的这种方式快速学到很多的知识。

2.2 Teacher Forcing 的问题：

Teacher Forcing 最常见的问题就是 Exposure Bias 了。在严肃开始介绍这个问题的时候，我们继续下上面不太严谨的比喻：

> 由于小明平常的学习都是由超级学霸指导下完成的。但是在真正考试的时候，这种情况平常根本没出现过。。没有了超级学霸在旁边，心态容易崩，答案写起来起来也感觉容易崩，才发现自己原来一直学习在学霸的阴影下，从没真正的学习过自己的错误。。。

上面的『比喻』，其实就是不太严谨的 Exposure Bias 现象了。更严谨的表述，由于训练和预测的时候decode行为的不一致， 导致预测单词（predict words）在训练和预测的时候是**从不同的分布中推断出来的**。而这种不一致导致训练模型和预测模型直接的Gap，就叫做 Exposure Bias。

除了常见的 Exposure Bias 问题之外，今年的ACL2019 最佳paper中还指出好几个存在的问题：

1. Teacher-Forcing 技术在解码的时候生成的字符都受到了 Ground-Truth 的约束，希望模型生成的结果都必须和参考句一一对应。这种约束在训练过程中减少模型发散，加快收敛速度。但是一方面也扼杀了翻译多样性的可能。
2. Teacher-Forcing 技术在这种约束下，还会导致一种叫做 **Overcorrect(矫枉过正)** 的问题。例如：

> 1. 待生成句的Reference为: "We should comply with the rule."
> 2. 模型在解码阶段中途预测出来："We should abide"
> 3. 然而Teacher-forcing技术把第三个ground-truth "comply" 作为第四步的输入。那么模型根据以往学习的pattern，有可能在第四步预测到的是 "comply with"
> 4. 模型最终的生成变成了 "We should abide with"
> 5. 事实上，"abide with" 用法是不正确的，但是由于gro
> 于矫枉过正的状态，生成了不通顺的语句。

0x03. 论文三篇：

在学术届中，其实早就意识到 Teacher-Forcing 所带来的问题，今年的ACL2019 Best-Paper，也是主要建立在如何解决翻译问题上的teachering-forcing问题。在这里我分享个人认为比较有价值的三篇论文。不做详细解读，仅对其核心思路进行阐述。

3.1. Scheduled Sampling:

这篇论文全称为 Scheduled Sampling for Sequence Prediction with Recurrent Neural Networks，是Google于2015年发表的一篇解决 exposure-bias 的论文。 该论文Google-Scholar上引用次数680次，目前也算是这个问题最solid的方案了，之后的论文idea中都存在着不少他的身影。因此，目前基本上大家谈及 exposure-bias，最常说起的就是 Scheduled-Sampling 方案了。

这个方案本身其实也是很朴素。既然 Teacher-Forcing 技术在训练前期的确是能够很大的加速模型收敛的， 那么能否设计出一个方案：

1. 模型在训练过程中的每一个steps，有 p 的概率选择使用 teachering-forcing，有 $1-p$ 的概率选择使用 Autoregressive。
2. 模型在训练前期， p 应该尽可能的大，这样能够加速收敛；而在快要结束训练的时候，$1-p$ 尽可能的小，让模型在 Autoregressive 的方案中尽可能的修复自身生成的错误。

更确切的，这个 p 概率可以随着训练的Steps or Epoch 进行衰减，而衰减的方式也可以分为：Exponential Decay, Inverse Sigmoid decay 和 Linear decay 三种方式：

值得注意的是，上面的这个概率 p ，是针对一个token而言的，而不是针对整句话。也就是说在解码过程中，每个token的生成，都要进行着这么一次概率的选择。论文中指出，如果是整句话进行概率选择的话，效果会比较差。。

Scheduled Sampling 在MSCOCO 图片标题生成实验中的结果

从上图中可以看到几个有趣的现象：

- Always Sampling：其实就相当于在训练过程只使用Autoregressive 方案（每次使用上一步的预测单词），可以发现模型效果非常差，收敛有问题。
- Uniform Scheduled Sampling：可以理解成每次都有0.5的概率选择 Teacher-Forcing，0.5的概率选择Autoregressive，效果也比 Scheduled-Sampling 要差

3.2. Scheduled Sampling for Transformers

论文 Scheduled Sampling for Transformers只是 ACL2019 workshop 中的一篇小论文。但是其立足点主要关注在更新的 Transformers 上，个人觉得还是值得一读。并且在 Tensor2Tensor 中也有 跟原文思路类似的 Scheduled Sampling 的代码实现。

回到论文本身，为什么要专门为Transformers 设计不同的Scheduled Sampling呢？原因有二：

1. 熟悉Transformers的同学会发现，采取Scheduled-Sampling这种方式来训练Transformers的话，会对其并行性产生极大的破坏，这对于训练效率来看是会有很大的降低的。 这个问题对于RNN来说是不存在的，因为RNN即使不等待上一个steps的输出 y_{t-1} ，也要等待上一步骤的hidden-state h_{t-1}
2. RNN每次只会依赖于上一步骤的结果，但是对于 Transformers 而言，由于SelfAttention的存在，解码阶段依赖的是前面所有steps的数据。因此 Scheduled-Sampling 中只把上一步的结果进行替换，是不是不太充足呢？（个人觉得这个理由好像成立，但是又有点牵强。。。）

据此，论文中设计了一个 two-pass 的解码方案：

1. 在每个training-steps，第一趟先利用teacher-forcing技术，计算出当前句子中每个解码位置所有单词的分数（logits）
2. 根据一定的概率 p ，选择第二趟解码时，是否用第一趟生成的单词作为decode输入，还是沿用ground-truth作为输入（只有第二趟解码会进行back-propogation）
3. 如果选择第一趟生成的结果，那么每个位置根据预测单词的分数（logits），可以有以下操作：

- 利用 argmax 选择每个位置中分数最大的单词，作为输入。
- 利用分数进行加权平均得到一个embedding向量，作为输入。
- 取topk结果，利用分数进行加权平均得到一个embedding向量，作为输入。
- 根据分数进行多项式采样，作为输入。

可以看到，这种方式和传统的 Scheduled Sampling 相比，模型只需要并行的运行两次decode过程即可，比起Autoregressive的方式对并行性的破坏很小。同时，这种方式是对整句的翻译进行概率为 p 的采样，而不是针对每个位置。 但是通过将 logits 进行一定程度上的处理（例如平均、采样等），加大了训练过程中的噪声，提高了健壮性。在 `tensor2tensor` 中，如果令 `scheduled_sampling_method="parallel"` ，则用的就是上面的这种采样方法。

3.3. Bridging the Gap between Training and Inference for Neural Machine Translation

这篇论文作为ACL2019的最佳论文，出自中科院 FengYang 老师的实验室。初读这篇论文，觉得idea 还是蛮惊艳的。但是随着后面又读了上两篇论文，不觉有些失望。总体而言，这片论文中Scheduled-Sampling 的影子还是很重的，只是 添加了一些特殊的trick。这里对这篇论文就不再评述了。

0x04. 再次回顾实验：

再次回到第一节中探讨到的实验，问题始终是：**数据集中的校验 loss、appro-bleu 在持续变好，是否并不意味着最终inference过程中的 BLEU 也同步的变好呢？** 这里先直接说下结果：

1. BLEU 和 Loss，accuracy 从计算方法来看就不太一样，三者肯定是并非严格同步的。但是从长远时间来看，三者的优化方向应该还是同步的。
2. 在 tensor2tensor 中，approx-bleu 和 BLEU 本身就存在着一定程度的Gap，因为模型的输入是经过 BPE 分词后的subword， 而 approx-bleu 计算的是subword单元的分数，而 BLEU 本身计算的是合并 subword 后的token粒度的分数。
3. 无论在 tensor2tensor 还是 tensorflow 官方实现的Transformer模型，在默认的情况下，evaluation时候使用的是teacher-forcing技术（事实上，

设置参数 `--eval_run_autoregressive=True` 来令校验过程使用 autoregressive 而不是 teacher-forcing ）：

- 默认在 eval 的时候使用 teacher-forcing，个人猜测这么做的动机主要为了evaluation的效率着想，毕竟在teacher-forcing下，Transformer的校验过程就能做到完全并行，不再是 autoregressive 模式。
- 和Autoregressive相比，Teacher-Forcing存在类似作弊的行为，这才使得 Apprixate-BLEU 往往会比 BLEU 高 7 - 10 个点。

所以说如果BLEU已经开始下降了，然而校验过程中的一切参数（Accuracy、Approx-bleu）等还在健康变好，那就说明了 Exposure-Bias 现象在模型的已经比较严重了。**就好像小明在学霸的关照下觉得自己的知识水平在不断变好，但实际上，脱离学霸的小明总体知识水平开始下降了，因为他的思维潜在方式已经越来越依赖于学霸的存在，而不是真正关注在知识本身。**。这一点有点类似 overfitting。

基于这个现象，我在 mixed-finetune 的过程中引入了 Scheduled-Sampling，才所用的采样方式类似上面论文二中思路。可以看到，加入了 Scheduled-Sampling 后，BLEU结果有了明显的改善。

加入Scheduled-Sampling后，BLEU有了明显的改善（鲜红色的为改善结果）

上图暗红色的为没有采取 Scheduled-Sampling 的正常训练实验。可以看到在22.5k后，由于 Exposure-Bias 问题的存在， BLEU结果开始大幅下降（从24.0 下降到23.6 左右）。 但是加入 Scheduled-Sampling 后，随着 Trainning-steps 的增加，Teacher-Forcing 的比重开始下降，Exposure-Bias问题得到缓解，模型在inference时的BLEU得到进一步上升，甚至最高值能够到 24.51。 从峰值来看，提升有0.5个BLEU值，并且整个训练过程中BLEU分数更加平稳，健壮性进一步提升。

校验过程的其他参数对比

从上图可以看到，加入Scheduled-Sampling之后，并且随着trainning-steps的增大，Teacher-Forcing的比重下降后，模型在校验过程中类似 loss 和 accuracy 的值变差了。这是可以理解的。再次利用那个不太严谨的例子来结尾：

> 小明开始脱离学霸自己学习，尝试在日常生活中纠正自己的学习错误，虽然学习效率变低了（训练效率下降），但是闭卷考试的成绩得到了进一步的提升（Inference-BLEU）。但是在学霸关照模式下，成绩反而有些变差了(Evaluation)。那是因为小明开始有了自己的思路和想法，不再是为了最大化贴近学霸思路而给出答案。

最后，我们再来探讨一个问题：**为什么大规模语料训练通用模型的情况下，Exposure-Bias 的影响并不明显（指标优化同步），而在某些特定domain语料finetune的时候凸现出来了呢？**。个人猜测原因如下：

- 通用语料规模大，学习的容量也大。不容易使得Exposure-Bias问题凸现出来。
- Domain专业领域的翻译语料中，和通用语料的相比，**通用语料的翻译比较简单，而Domain翻译语料中存在着大量的意译、总结性翻译等问题。** 变数更多，对于机器翻译来说难度更大。正是因为难度的增大，在学霸的关怀下，表面来看还是蒸蒸日上，繁荣发展。但是在脱离学霸后，由于难度增大，Exposure-Bias的问题一下就凸现出来了。

最后不得不吐槽一下，小明太难了。。。。

0x05. Reference

- [1]. Scheduled Sampling for Sequence Prediction with Recurrent Neural Networks
- [2]. Scheduled Sampling for Transformers
- [3]. Bridging the Gap between Training and Inference for Neural Machine Translation

编辑于 2019-11-21

机器学习　神经机器翻译(NMT)