Skim Text

标签：RNN，LSTM，NLP

1.Introduction

该模型基于传统 LSTM 模型，尝试解决传统模型难以使用大规模长序列输入文本的问题。该模型会选择去跳过几个 token，来进行下一次输入。

2.Detail

在训练该模型之前，我们需要给定一下几个参数：

K：maximum size of jumping
N：The number of jumps allowed
R：The number of tokens read between two jumps

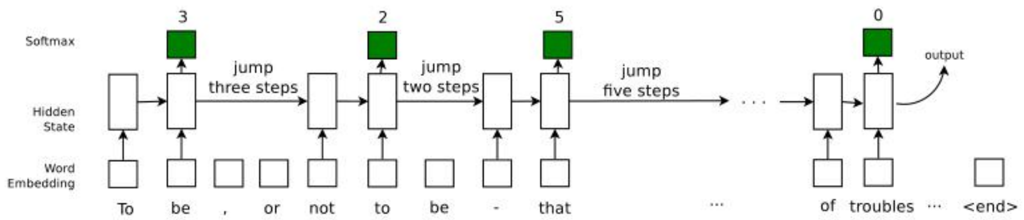
在这几个参数之中，K 是固定值，而超参数 N 和 R 是可以在训练和测试的过程中进行修改的。

什么时候，整个过程停止呢？

三个条件：1）softmax 函数取样为 0
2）跳跃的次数超过 N
3）模型已经取到最后一个 token

过程结束之后，得到输出 Hidden State，用以预测目标，针对具体任务具有具体的作用。例如：在分类问题中，可以直接产生 softmax 函数，在自动问答中，可以用来计算候选答案的正确性，以选择最好的一个。

具体举一个例子，如图所示：



在这个例子中，输入被写为：$x_{1:p}$ 表示 $x_1, x_2, ..., x_p$；我们设置一次跳跃最大距离（maximum size of jump K）为 5，每次输入 token 数目 R（number of tokens read before a jump）为 2，允许跳跃的次数 N 为 10，绿色的部分就是预测跳跃的 softmax 函数。

那么这些参数具体怎么预测呢？

3.Estimate parameter

因为 NLP 的特性是离散的，所以带来一个难题不可微，所以在这里，我们使用增强学习来进行参数估计。

我们需要训练 LSTM 的参数，描述单词嵌入的可能性 θ_m，以及描述跳跃行为的参数 θ_a。对于 θ_m 的估计，可以使用交叉熵来进行估计，瞬间变成一个可对 θ_m 微分的东西，所以我们直接使用 backpropagation 来进行最小化。

而对于 θ_a 的估计，因为跳跃行为的步数是一个离散的值，所以我们把它党走一个增强学习的问题，并且采取了梯度策略的方法来解决。

对 θ_a 最大化一个反馈函数：

$$J_2(\theta_a) = \mathbb{E}_{p(j_{1:N};\theta_a)}[R].$$

$$\text{where } p(j_{1:N};\theta_a) = \prod_i p(j_{1:i}|h_i(j_{1:i-1});\theta_a).$$

其中：

$p(j_i|h_i(j_{1:i-1});\theta_a)$ 代表：jump steps 服从的多项分布（由 jump softmax 决定的），h_i 为 LSTM 的恰好在第 i 次 jump j_i 的之前的隐藏态，$j_{i:N}$ 表示跳跃行为的序列，R 是在现有策略下完成对序列的操作之后所得到的一个反馈系数。

优化这个目标需要对其进行梯度的计算，但期望其中所包含的高维度的相互作用序列让优化变得比较棘手，所以我们使用强化学习的算法来计算一个近似的梯度。公式如下：

$$\nabla_{\theta_a} J_2(\theta_a) = \sum_{i=1}^{N} \mathbb{E}_{p(j_{1:N};\theta_a)}[\nabla_{\theta_a} \log p(j_{1:i}|h_i;\theta_a)R]$$

$$\approx \frac{1}{S}\sum_{s=1}^{S}\sum_{i=1}^{N}[\nabla_{\theta_a} \log p(j_{1:i}^s|h_i^s;\theta_a)R^s]$$

其中 S 为样本的数量， $\nabla_{\theta_a} \log p(j_{1:i}|h_i;\theta_a)$ 可以使用 BP 算法来计算。

上述对于 $\nabla_{\theta_a} \log p(j_{1:i}|h_i;\theta_a)$ 的估计是无偏的，但是很大可能器方差会很大，所以我们通常将我们的得到的反馈值 R_i^s 去减去一个基准值 b_i^s （baseline value），得到

$$\nabla_{\theta_a} J_2(\theta_a) \approx \frac{1}{S}\sum_{s=1}^{S}\sum_{i=1}^{N}[\nabla_{\theta_a} \log p(j_{1:i}^s|h_i^s;\theta)(R^s - b_i^s)]$$

b_i^s 服从无偏估计 $b_i^s = w_b \hat{h}_i^s + c_b$ ，其中参数 θ_b 由最小化 $\hat{(}R^s - b_i^s)^2$ 得到，所以最终的目标变成：

$$J(\theta_m, \theta_a, \theta_b) = J_1(\theta_m) - J_2(\theta_a) + \sum_{s=1}^{S}\sum_{i=1}^{N}(R^s - b_i^s)^2$$

该式子式完全可微的，所以最小化这个目标可以通过标准的 bp 算法来解决。