Seq2Seq

2021年2月4日 14:24

1. Sequence-to-Sequence

- 链接：https://www.youtube.com/watch?v=gxXJ58LR684&t=916s

前面两个模型的最主要的问题：如果训练的数据集不够，而多层lstm叠加让embedding层的参数过多，就会出现过拟合的现象，导致实验结果不理想

- Tokenization & build dictionary

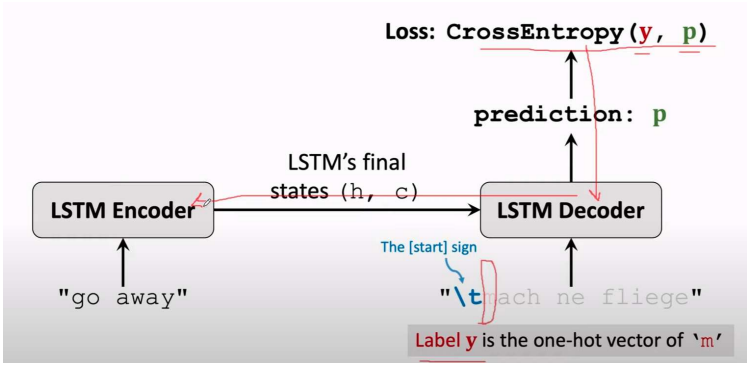
  - English: A a, B b, C c ..., Z z. (26 letters ×2).
  - German: 26 letters, 3 umlauts (Ä,Ö,Ü), and one ligature (ß).
  - Greek: A α, B β, Γ γ, Δ δ, ..., Ω ω. (24 letters ×2).
  - Chinese: 金 木 水 火 土 ... 赵 钱 孙 李 (a few thousands characters).
  - Japanese: あ い う え お ... (46 Hiragana, 46 Karagana, hundreds 漢字).

1. tokenization 可以是char-level，也可以是 Word level，在此以char-level 为例子
2. 各个语种字符数量的统计
2. 然后对每个字符编号，组成dictionary
3. 注：起始符 start sign和终止符 stop sign 用于目标语言

- One-hot encoding

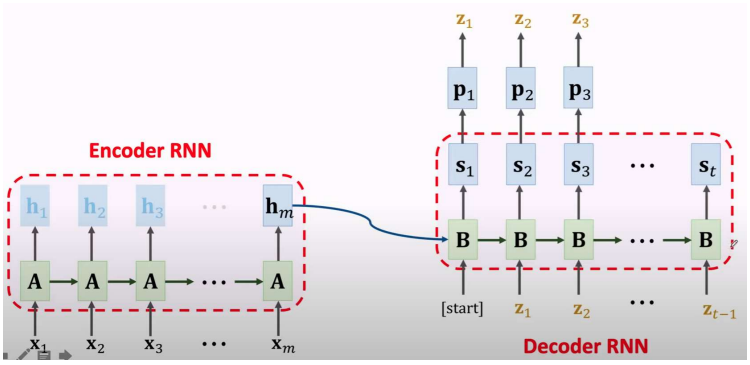
  根据vocabulary 把输入字符编码成特征向量

- Training Seq2Seq model



1. Decoder的初始状态是Encoder的最后一个状态h
2. 训练的流程如下：
   - 每次decoder得到encoder的状态向量h之后，首先会生成一个开起始字符
   - 然后预测得到下一个字符p
   - 标签y为正确的字符，计算标签y和预测值p之间的交叉熵，得到损失函数
   - 然后调整decoder参数，再梯度下降调整encoder参数，让损失函数最小
   - 然后输入字符“\tm”，预测得到p，再算a和p之间的交叉熵，梯度反向传播到decoder和encoder，调整参数
   - 然后输入字符“\tma”，balabla.....以此往复
   - 最终遇到[stop] sign便停止训练，得到训练好的模型

- Seq2Seq model inference

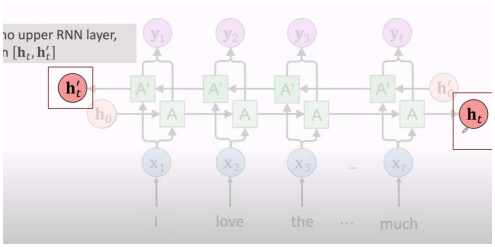


1. 流程如下：
   - 首先在encoder中输入每个token，获得最后一个状态h_m，该状态包含encoder输入文本的全部信息
   - 将h_m输入到decoder中，首先生成起始符，然后decoder RNN更新状态为s_1，在全连接层，生成概率向量p_1，选择最大概率的z_1为预测值。
   - 然后 decoder网络以z_1为输入，更新状态为s_2，预测生成z_2，以此往复
   - 得到终止符[stop]，返回序列 z_i

- How to improve
  - 用双向lstm代替单向的lstm，只能在encoder部分！！因为decoder是按照序列生成的，必须是单向的！
  - 使用word level的tokenization代替char-level
    因为平均英文单词的长度为4.5个字符，用word level的话，输入会短4.5倍，不容易被遗忘
    但是word level的高准确性是建立在有足够large dataset的基础上的，平常word level的维度为10000，embedding的话数据一定得够，不够的话容易产生overfitting的问题

- multi-task learning 来训练来针对不同的任务训练同一个encoder 的embedding层（比如英文翻译成多个语种，虽然decoder不同，但是embedding是同一个）

1. Bidirectional RNN (双向RNN)



2. Pretrain the embedding layer