Chapter 10：Recurrent Neural Network

应用

- 机器翻译
- 序列标注
- 图像描述
- 推荐系统
- 智能聊天机器人
- 自动作词作曲等

RNN VS CNN

- CNN
  - 传统文本处理任务的方法中一般讲TF-IDF向量作为特征输入，丢失了输入的文本序列中单词间的顺序关系
  - 接受定长的向量作为输入，通过滑动窗口加池化，可以捕捉到局部信息，但是单词间的长距离依赖关系很难学习到
- RNN
  - 能够很好地处理文本数据变长且有序的输入序列，将输入中的有用信息编码到神经网络状态变量中去，从而拥有了一定的记忆能力
  - 在h_T后面直接接一个Softmax层，输出文本所属类别的预测概率y，f可以选择Tanh函数或者Relu函数，通过最小化损失误差(即输出的y与真实类别之间的距离)，就可以实现文本分类
  - 文本分类问题
    - $net_t = Ux_t + Wh_{t-1}$
    - $h_t = f(net_t)$
    - $y = g(Vh_T)$
    - h_t和y的计算公式
  - f和g为激活函数，U为输入层到隐含层的权重矩阵，W为隐含层从上一时刻到下一时刻状态转移的权重矩阵

Seq2Seq：Sequence to Sequence

- 核心思想
  - 通过DNN将一个作为输入的序列映射为一个作为输出的序列，该过程由编码输入和解码输出两个环节构成。编码器和解码器可以是传统RNN、LSTM等实现，两个RNN是同时训练的
- 解码方法
  - 贪心法
    - 选取一种度量标准后，每次都在当前状态下选择最佳的一个结果，直到结束
    - 计算代价地，获得的是一个局部最优解，对于实际问题，往往不能取得最好的结果
  - 集束搜索
    - 启发式方法，会保存b(beam size)个当前的较佳选择，然后解码时每一步根据保存的选择进行下一步扩展和排序，接着选择前b个进行保存，循环迭代，直到结束时选择最佳的一个作为解码结果
    - b取1，则退化为贪心法，随着b的增大，其搜索的空间增大，最终效果会有所提升，但计算量也增大。实际中选择b=8-12
  - 堆叠RNN
  - 与编码器之间建立残差连接
  - 增加Dropout机制
  - 采用记忆网络，从外界获取知识
- 问题
  - 实际使用中Seq2Seq模型会随着输入序列的增长，模型的性能发生了显著下降，因为编码时输入序列的全部信息压缩到了一个向量表示中，序列越长，句子前面的词的信息丢失就越严重
  - 模型的输出序列中，常常会损失部分输入序列的信息，因为在解码时，当前词语对应的源语言词的上下文信息和位置信息在编解码过程中丢失了

改进

注意力机制Attention Mechanism

仍然使用普通的RNN对输入序列进行编码，得到隐状态，但是在解码时，每一个输出词都依赖于前一个隐状态以及输入序列每一个对应的隐状态

$$s_i = f(s_{i-1}, y_{i-1}, c_i)$$

$$p(y_i|y_1, y_2, ..., y_{i-1}) = g(y_{i-1}, s_i, c_i)$$

$$c_i = \sum_{j=1}^{T} \alpha_{ij} h_j$$

其中语境向量c_i是输入序列全部隐状态h_1,h_2,...,h_T的一个加权和

$$\alpha_{ij} = \frac{exp(e_{ij})}{\sum_{k=1}^{T} exp(e_{ik})}$$

$$e_{ij} = a(s_{i-1}, h_j)$$

注意力权重参数a_ij不是固定权重，而是另一个神经网络计算得到

使用双向RNN能够保证x_i时刻的前后隐状态信息都被考虑到

激活函数

- Relu
  - 将net_t的表达式展开，会发现式中包含t个W连乘，如果W不是单位矩阵，最终网络在T时刻的输出结果将会趋于0或者无穷，引发严重的数值问题
  - 对于梯度来说，只要W不是单位矩阵，梯度还是会出现消失或者爆炸的现象
  - 但是在CNN中没有该现象，因为CNN中每一层的权重矩阵W都是不同的，并且在初始化时它们是独立同分布的，因此可以相互抵消，在多层之后不会出现严重的数值问题
- Sigmoid与Tanh
  - 因此只有当W的取值在单位矩阵附近时才能取得比较好的结果，需要将W初始化为单位矩阵
  - 都是饱和的。在输入达到一定值得情况下，输出就不会发生明显变化，可以达到门控的效果
  - 使用非饱和的Relu函数则难以达到门控的效果

训练

RNN的训练可以采用BPTT(Back Propagation Through Time)，基于时间的反向传播算法实现

- 梯度爆炸
  - 预测误差沿着施神经网络的每一层反向传播，当雅克比矩阵的最大特征值大于1时，随着离输出越来越远，每层的梯度大小会呈指数增长，导致梯度爆炸
  - 通过梯度裁剪来缓解，即当梯度的范式大于某个给定值时，对梯度进行等比收缩
- 梯度消失
  - 当雅克比矩阵的最大特征值小于1时，随着离输出越来越远，每层的梯度大小会呈指数缩小，产生梯度消失
  - 需要对模型本身进行改进
    - 上短时记忆模型LSTM
    - 加入门控机制，很大程度上弥补了梯度消失所带来的损失

LSTM：Long Short Term Memory

- 输入门i_t — 控制当前计算的新状态以多大程度更新到记忆单元中
- 遗忘门f_t — 控制前一步记忆单元中的信息有多大程度被遗忘掉
- 输出门o_t — 控制当前的输出有多大程度上取决于当前记忆单元

Sigmoid函数

- 内部记忆单元c_t
- 仍然基于x_t和h_{t-1}来计算h_t

Tanh函数

$$i_t = \sigma(W_i x_t + U_i h_{t-1} + b_i)$$

$$f_t = \sigma(W_f x_t + U_f h_{t-1} + b_f)$$

$$o_t = \sigma(W_o x_t + U_o h_{t-1} + b_o)$$

$$c_t' = \tanh(W_c x_t + U_c h_{t-1})$$

$$c_t = f_t \odot c_{t-1} + i_t \odot c_t'$$

$$h_t = o_t \odot \tanh(c_t)$$

2018.12.23 Sunday. By KuKuXia@github.com

ML/DL/RL深度交流讨论群: 622545311，加群时请备注: 姓名-方向-公司or高校