**《****自然语言处理》大作业**

**自实现单双层LSTM模型**

学 院： 计算机科学与工程

班 级： 人工智能2001

姓 名： 许子强

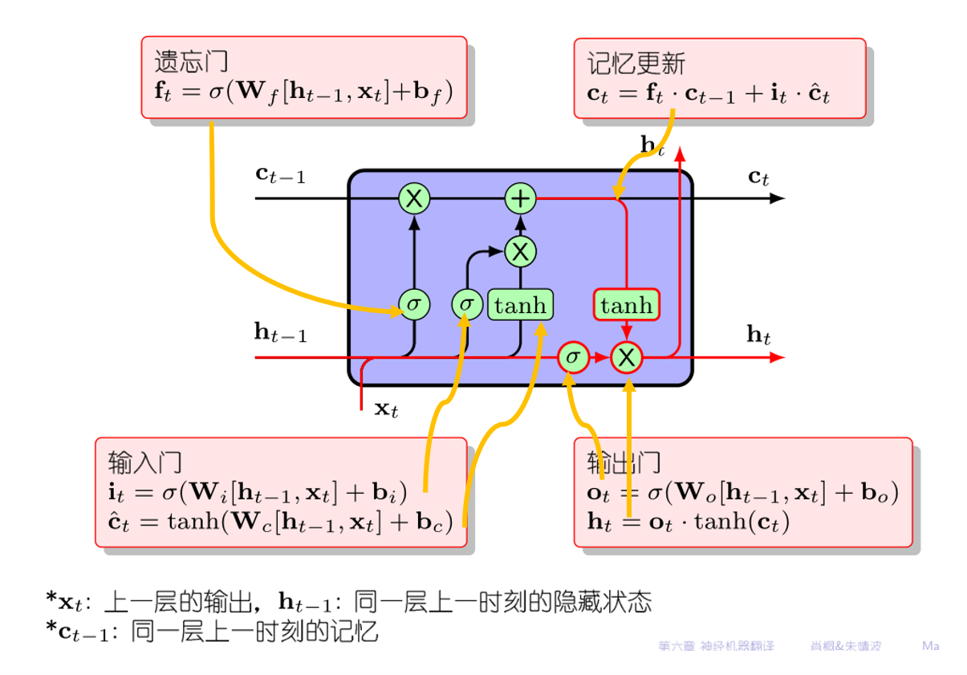
学 号： 20201111

1. **系统设计**
   1. **单层LSTM**

LSTM的关键是记忆状态，表示为Ct，用来保存当前LSTM的状态信息并传递到下一时刻的LSTM中。当前的LSTM接收来自上一个时刻的记忆状态Ct−1，并与当前LSTM接收的信号输入xt共同作用产生当前LSTM的记忆状态Ct。

LSTM主要包括三个门结构：遗忘门、输入门、输出门。这三个门用来控制LSTM的信息保留和传递，最终反映到记忆状态Ct和输出信号ht。

1.1.1 计算公式



1.1.2 参数说明

n\_step（5）：一层LSTM的循环个数。

n\_hidden（5）：隐藏状态、记忆状态的特征数。

emb\_size（128）：词嵌入维度。

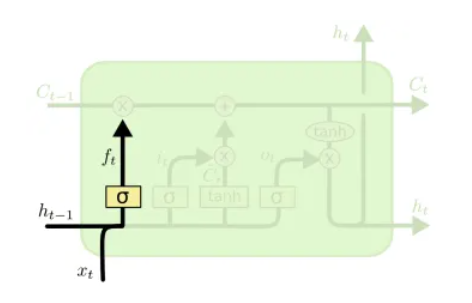
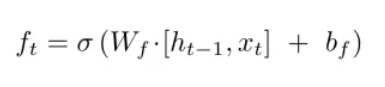
n\_class（7613）：词表长度。

batch\_size（512）：批训练时，一批的数据个数。

1.1.3 结构设计

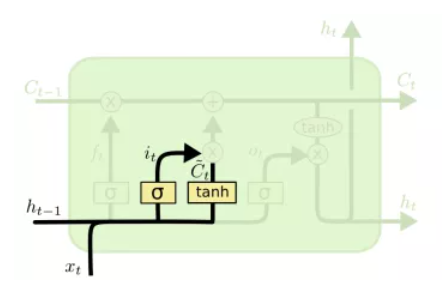
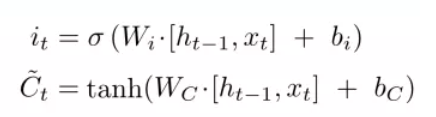
1. 遗忘门

遗忘门决定了细胞状态Ct−1中的哪些信息将被遗忘。由一个sigmoid神经网络层和一个按位乘操作构成。



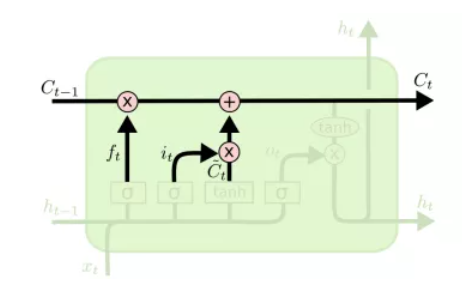
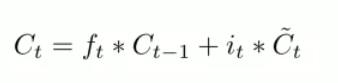
1. 输入门

输入门决定新输入的信息xt和ht−1中哪些信息将被保留。由一个sigmoid神经网络层、一个tanh神经网络层和一个按位乘操作构成。



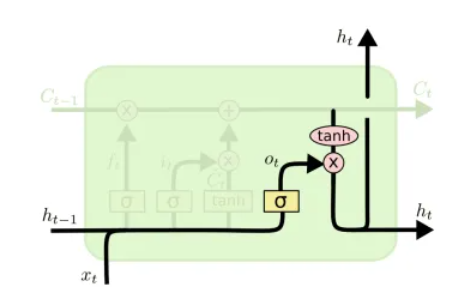
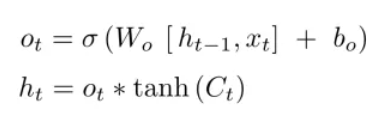
1. 更新记忆状态

将遗忘门的输出ft与上一时刻的细胞状态Ct−1相乘来选择遗忘和保留一些信息，将记忆门的输出与从遗忘门选择后的信息加和得到新的细胞状态Ct。

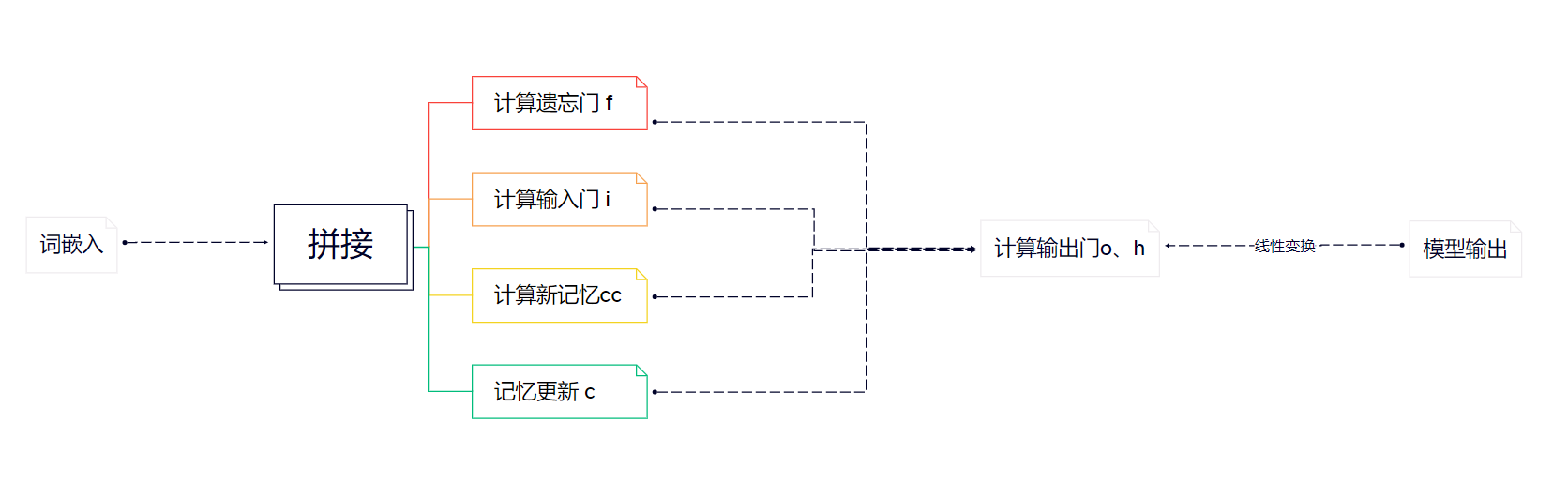


1. 输出门

输出门与tanh函数以及按位乘操作共同作用将细胞状态和输入信号传递到输出端。由一个sigmoid神经网络层、一个tanh函数和一个按位乘操作构成。



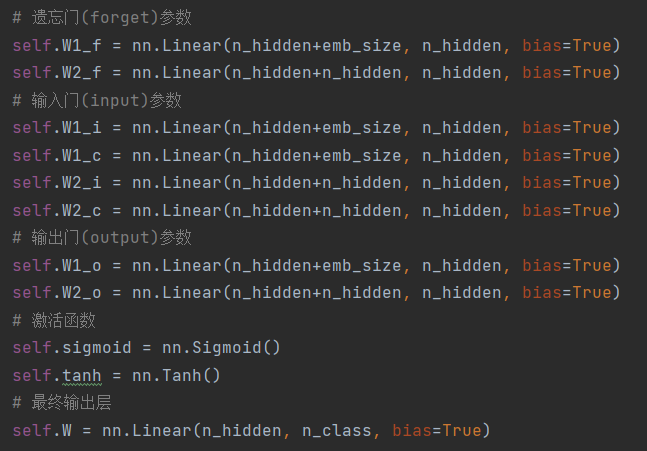
1.1.4 流程概要



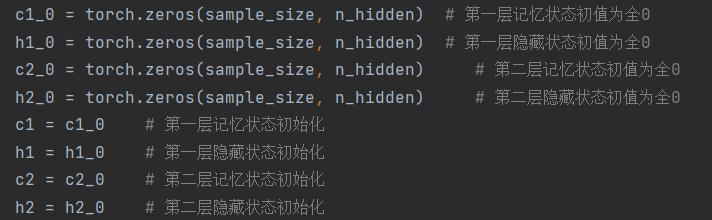
**1.2 双层LSTM**

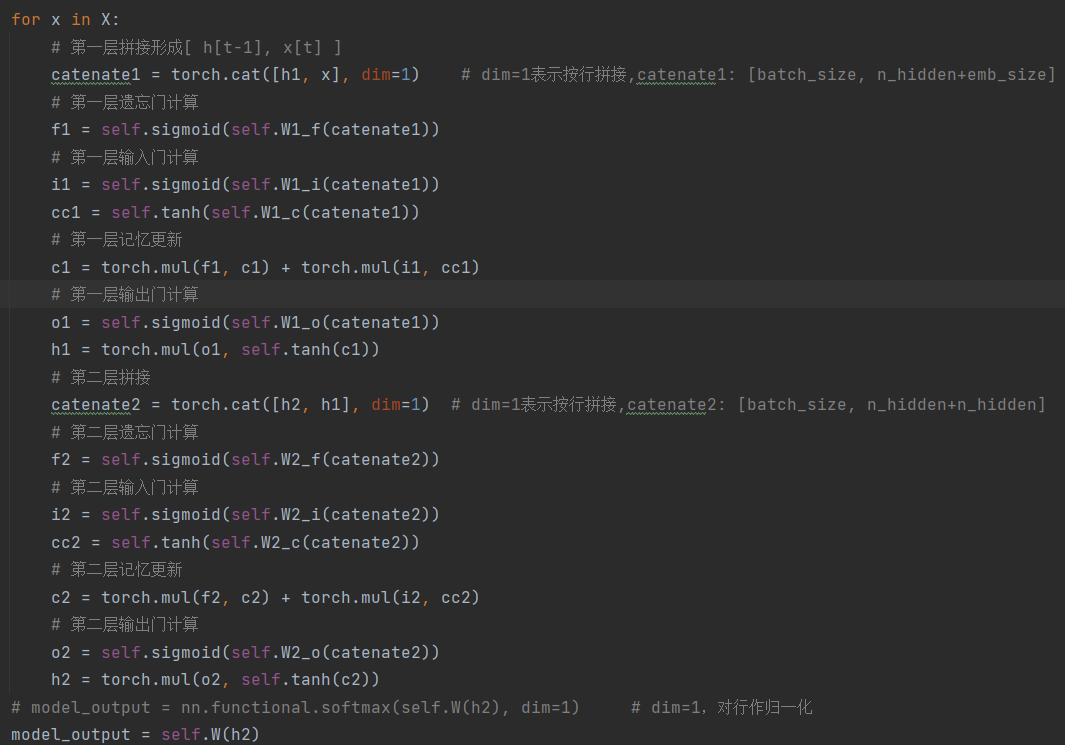
双层LSTM整体思路与单层并无大的差异，只是额外把第一层的输出作为第二层的输入，整体上大致相当于用相同的流程处理了两次，只是第二层参数的输入维度发生了变化——由[n\_hidden+emb\_size, n\_hidden]变为[n\_hidden+ n\_hidden, n\_hidden]。在此只简略给出相关设计。

1.2.1 参数设计



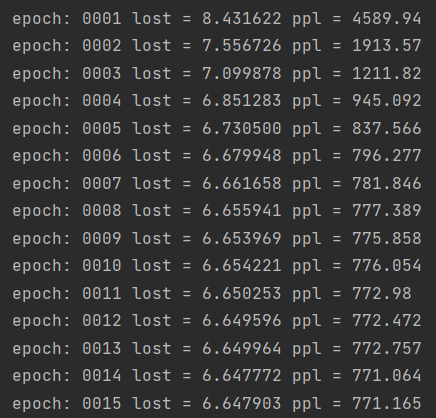
1.2.2 计算流程



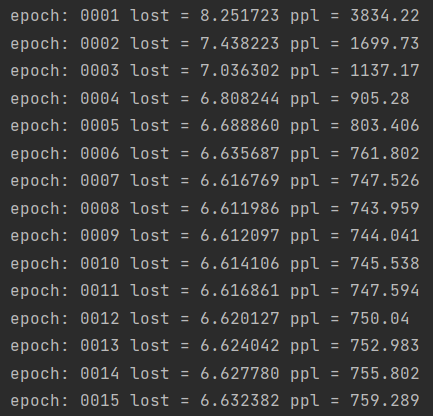


1. **实验结果**

**2.1单层LSTM**

****

**2.2双层LSTM**



1. **实验分析**
2. 维度分析

在实践课上听学长说要看懂参数的维度，当时不以为意。然而在我刚开始做实践作业1时，就因为维度问题头疼了很久。在本次大作业期间也出现过因为tensor的维度不匹配而报错，当即明白了——弄明白参数的维度很重要。

1. 参数不宜过多

如果每层的参数都较大且层数较多，如将多个隐藏层输出维度设置成n\_class（7613），则训练时显存不够，无法训练。

1. 模型输出层是否使用Softmax

在训练时发现个有意思的现象——双层LSTM模型中，如果在模型的最终输出层上，先用线性变换拟合输出概率、再用Softmax归一化，那么loss、ppl会非常非常缓慢地减小；但如果只用线性变换去拟合而不用Softmax，那么loss、ppl会先迅速减小、再缓慢增大。很遗憾目前不清楚原因。

1. 单、双层LSTM结果对比

在不使用Softmax的情况下，单层LSTM训练时loss、ppl相对较大，但一直保持着减小的趋势；而双层LSTM虽然loss、ppl较小，但有先减小、后增大的趋势。

1. 总结感悟

为期八周的NLP课程已经结束了，我很高兴能选到这样一门压力相对较小、又能学到理论知识、实践经验的好课。虽然刚开始搭建环境、接触pytorch框架的时候很痛苦，但经过实践课的学习，我对编程环境、pytorch框架都更熟悉了，也对神经网络的参数、前向传播有了一定认识。很感谢细心的老师、助教学长们，考虑到包括我在内的许多学生初次接触python编程，而为我们减轻工作量。

总之，通过NLP课程我收获了很多，无论是肖老师细致入微的讲解，还是博士生学长们亲自指导的实践，都让我印象深刻。很庆幸能在这个团队的指导下学习NLP课程，也希望以后能有更多的机会接触这样的团队。

1. 附录

源码GitHub地址：<https://github.com/Sleepyhead1111/NLPwork>

参考资料1：LSTM-muyongyu.pptx

参考资料2：[Pytorch官方文档](https://pytorch.org/docs/stable/generated/torch.nn.LSTM.html#torch.nn.LSTM)

参考资料3：[深入浅出LSTM](https://zhuanlan.zhihu.com/p/104475016)