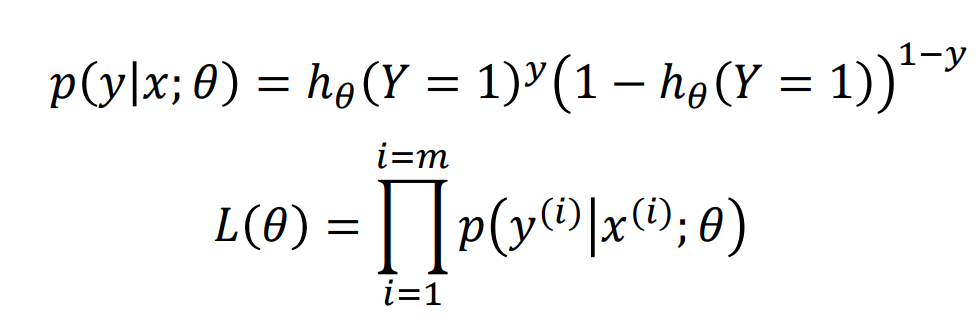
1. 概述

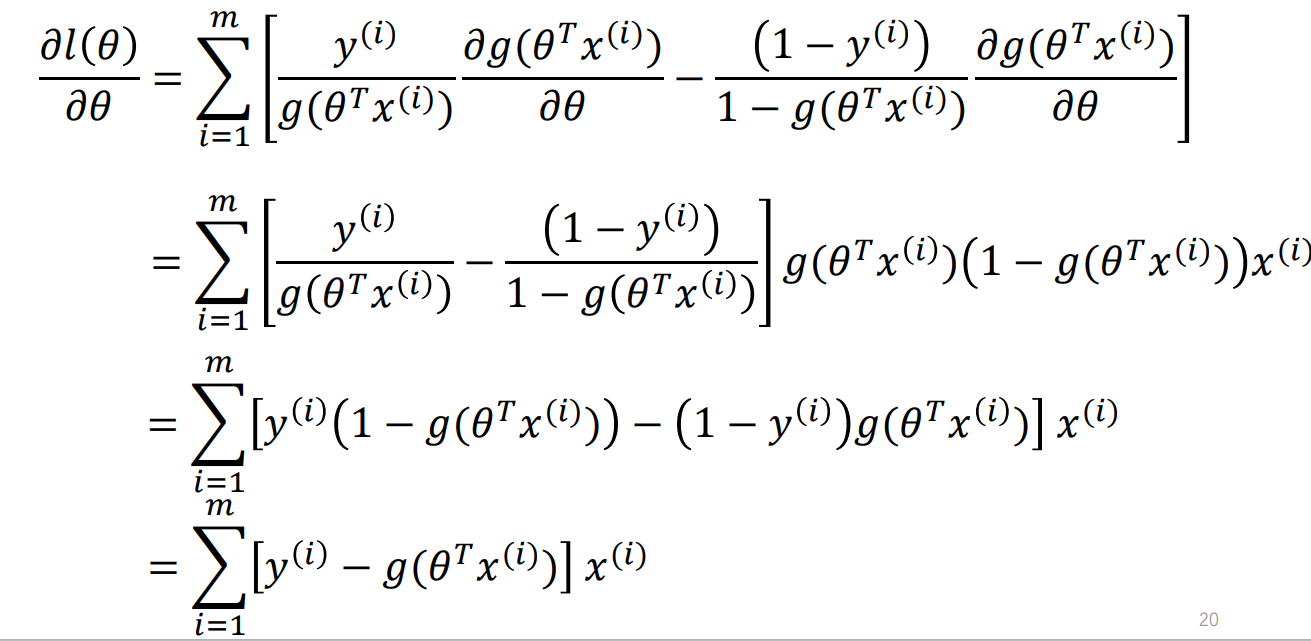
本次作业对1998年1月的人民日报做了明明命名实体识别，识别对象为机构名，使用one-hot模型表示词向量，对实体进行二分类，并使用sigmoid作为激活函数进行逻辑回归。数据集来自北京大学计算语言学教育部重点实验室 (pku.edu.cn)，使用时先进行了预处理，梯度下降使用了小批量梯度下降，测试集的f1-measure可训练至0.7或更高。

1. 模型原理

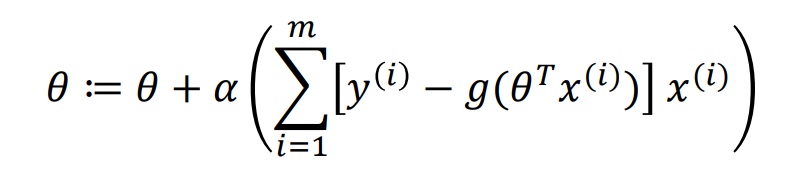
此次使用的one-hot模型中，我们对每一个实体构造一个5000维的向量，每一维代表一个词，用于标记其是否在实体中出现。在逻辑回归时，我们引入sigmoid函数，构造出一个以三个相邻实体的词向量拼接而成的向量为输入，参数为，关于整个样本集中所有实体的分类的似然函数。



然后我们得到其等价函数，并对其求导得到梯度下降方向。



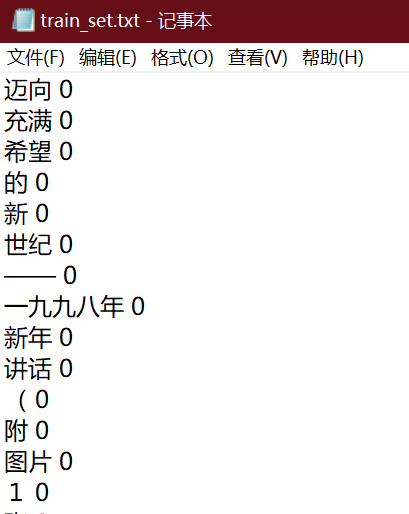
对于本次作业所选取的批量梯度下降，便有以下训练公式



用训练后得到的参数去计算，便能做出良好的预测，我们可以根据f1-measure来评价预测的准确性。

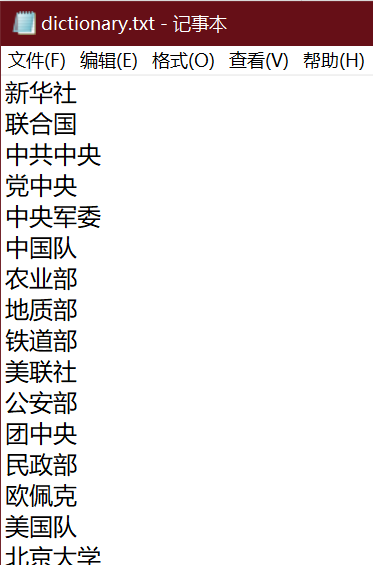
1. 实验流程
2. 处理原始数据集

将原始数据集命名为people\_daily.txt，取19980101-19980120为训练集，19980121-19980125为验证集，19980126-19980131为测试集。调用source目录下collect\_set.py即可完成分割，得到train\_set.txt，validation\_set.txt，test\_set.txt。在分割时，将’[’和’]’之间的词语分隔开，写在一行作为一个实体。对于每个实体，若为机构名称，则在后面写数字1，否则为0。处理后的数据集如图所示



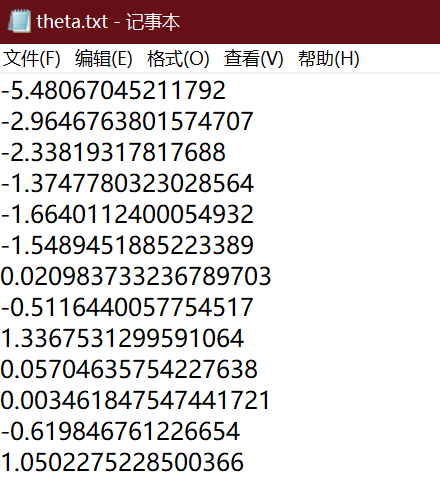
1. 获得词向量字典

对people\_daily.txt中的每一个词进行计数，优先将机构名放入词典，然后对非机构名排序，依次放入词典直到满足5000个，将完成后的词典存入dictionary.txt。调用source目录下collect\_dictionary.py即可。字典如图所示



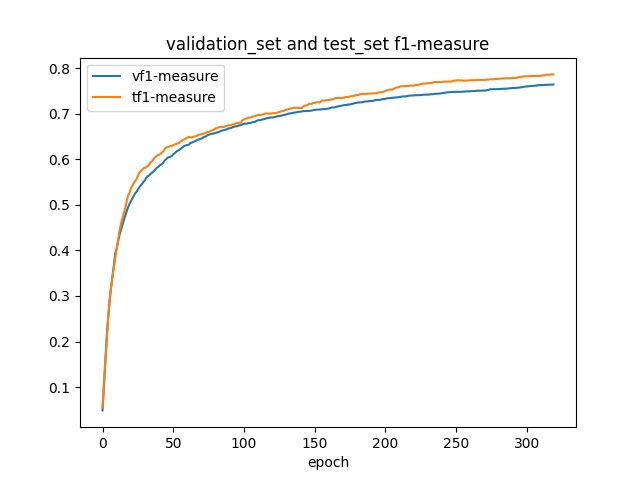
1. 训练模型

调用main.py，将训练集，验证集，测试集读入，若首次进行训练，则随机化theta并存入theta.txt，否则从theta.txt中直接读入theta。在训练时使用批量梯度下降，学习率取0.005，每次迭代使用个实体作为样本集更新参数。一次epoch训练完成后，分别计算在验证集和测试集上的新的f1-measure，并将f1-measure和新的theta写入对应文件。theta.txt如图所示



1. 结果分析

模型训练过后做出的预测效果良好，在验证集和测试集的f1-measure都超过了0.75，并仍有上升空间。猜测受限于时间和机器性能，此次未能观察到过拟合现象。



1. 总结与收获

这次作业让我第一次开始写“炼丹”的代码，并且出乎意料的有趣。在动手写代码之前，我首先花了大量的时间在网上学习相关的知识，有了一个更加清晰的思路，这使得后面用于写代码的时间并不算长。本次的模型并不算复杂，但却帮助我构建和完善了知识框架，课下也许可以试一试更多的技巧，比如重点学习pytorch自带的autograd，还可以尝试随机梯度下降，也可以试一试不同的字典选取方式。

这次遇到的最大的困难是我的设备，由于是集成显卡，无法支持cuda，并且很容易引起cpu温度过高，这让我花了很多时间思考解决方案。最后我租了一个云服务器，并在上面搭好了此次作业所需的环境，成功完成了模型的训练，之后可以尝试租用可以使用cuda的云服务器，应该可以达到更优的训练速度。