**实验五：诗词创作**

综合评分：

【实验目的】：

1. 了解深度神经网络的概念与原理。
2. 学习并使用tensorflow构建RNN模型
3. 实现基于RNN语言模型进行诗句的生成

【实验内容】：

|  |
| --- |
| 1. 数据预处理，将训练数据中所有的字生成了一个"文字==>数字"的词袋。 2. 构建生成器，诗词生成模型根据上一个字符生成下一个字符。 3. 构建RNN模型，并进行初始化，进行变量定义，embedding定义，模型计算图定义，权重及偏置定义，损失及优化图定义等。 4. 训练RNN模型。 5. 测试模型，生成古诗词。 |

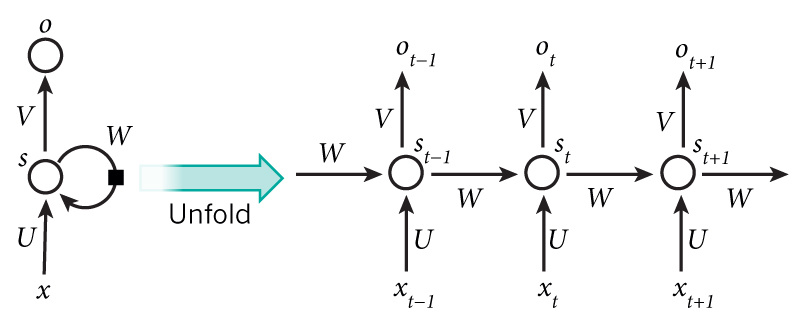
【实验工具及平台】：

Windows 10 + Anaconda + Tensorflow

【实验涉及到的相关算法】：

主要算法： RNN

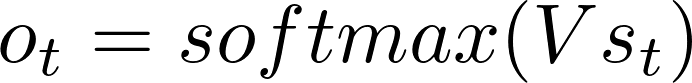
1.基本概念：循环神经网络，用于解决训练样本输入是连续的序列,且序列的长短不一的问题，比如基于时间序列的问题。基础的神经网络只在层与层之间建立了权连接，RNN最大的不同之处就是在层之间的神经元之间也建立的权连接。RNN神经网络的结构如下：



此图展示了RNN被展开成一个全网络后的结构。这里展开的意思是把针对整个序列的网络结构表示出来。例如，如果这里我们关心的是一个包含5个词的句子，那这里网络将会被展开成一个5层的网络，每个词对应一层。在RNN中进行的计算公式如下：

qt_temp是C:/Users/mcj11/AppData/Local/Temp/qt_temp.gX9132qt_temp时刻的输入。例如，C:/Users/mcj11/AppData/Local/Temp/qt_temp.Up9132qt_temp是句子中第二个词的one hot编码向量。

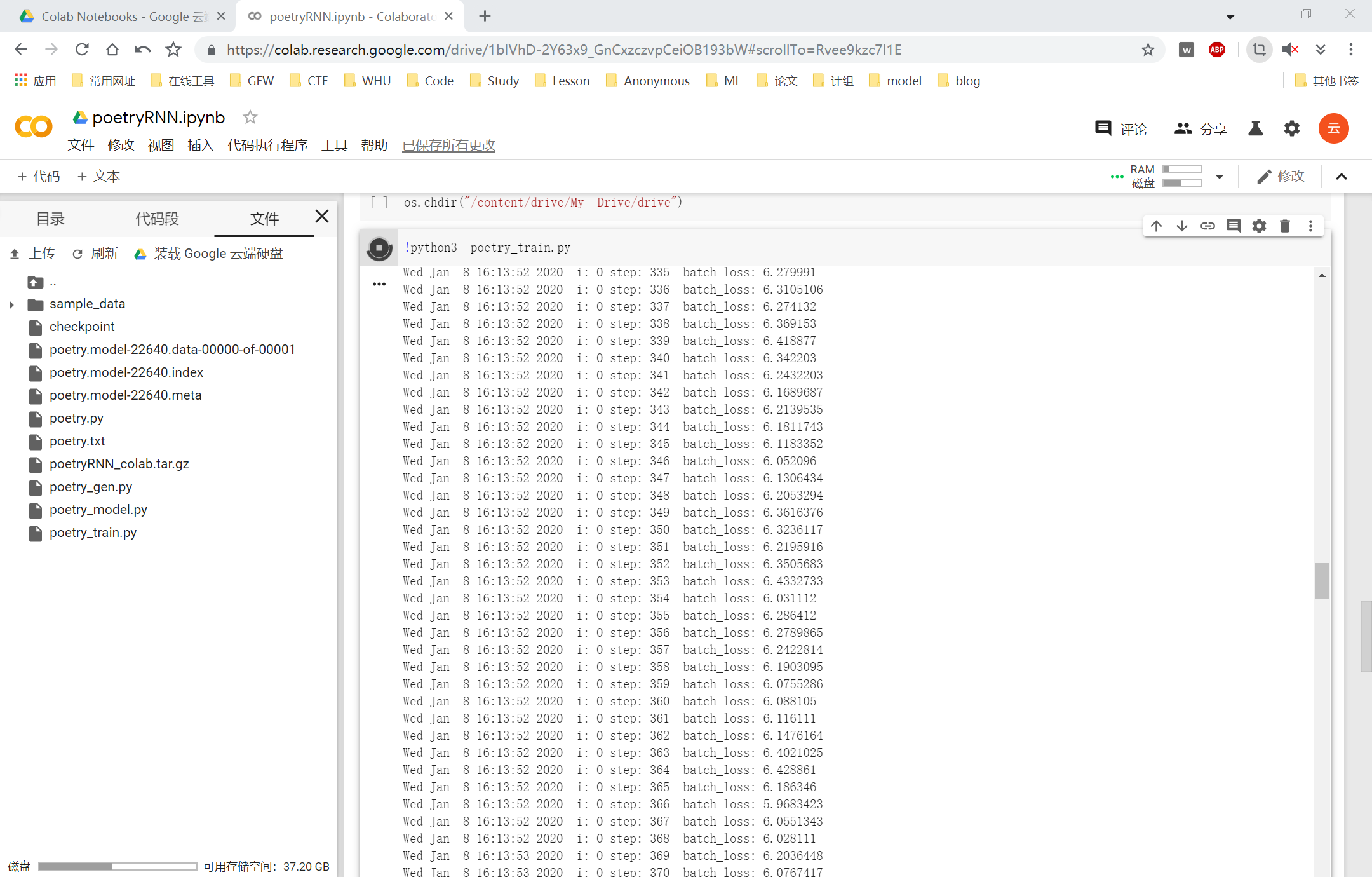
C:/Users/mcj11/AppData/Local/Temp/qt_temp.Pq9132qt_temp是对应C:/Users/mcj11/AppData/Local/Temp/qt_temp.gX9132qt_temp时刻的隐藏状态，是网络的记忆单元。C:/Users/mcj11/AppData/Local/Temp/qt_temp.Pq9132qt_temp通过前一步的隐藏状态和当前时刻的输入得到：qt_temp。函数C:/Users/mcj11/AppData/Local/Temp/qt_temp.ql9132qt_temp通常是非线性的，例如tanh和ReLU。C:/Users/mcj11/AppData/Local/Temp/qt_temp.yL9132qt_temp通常用来计算第一个隐藏状态，会被全0初始化。

C:/Users/mcj11/AppData/Local/Temp/qt_temp.Xb9132qt_temp是C:/Users/mcj11/AppData/Local/Temp/qt_temp.DN9132qt_temp时刻的输出。例如，如果想要预测句子中的下一个词，那么它就会是包含词表中所有词的一个概率向量，。

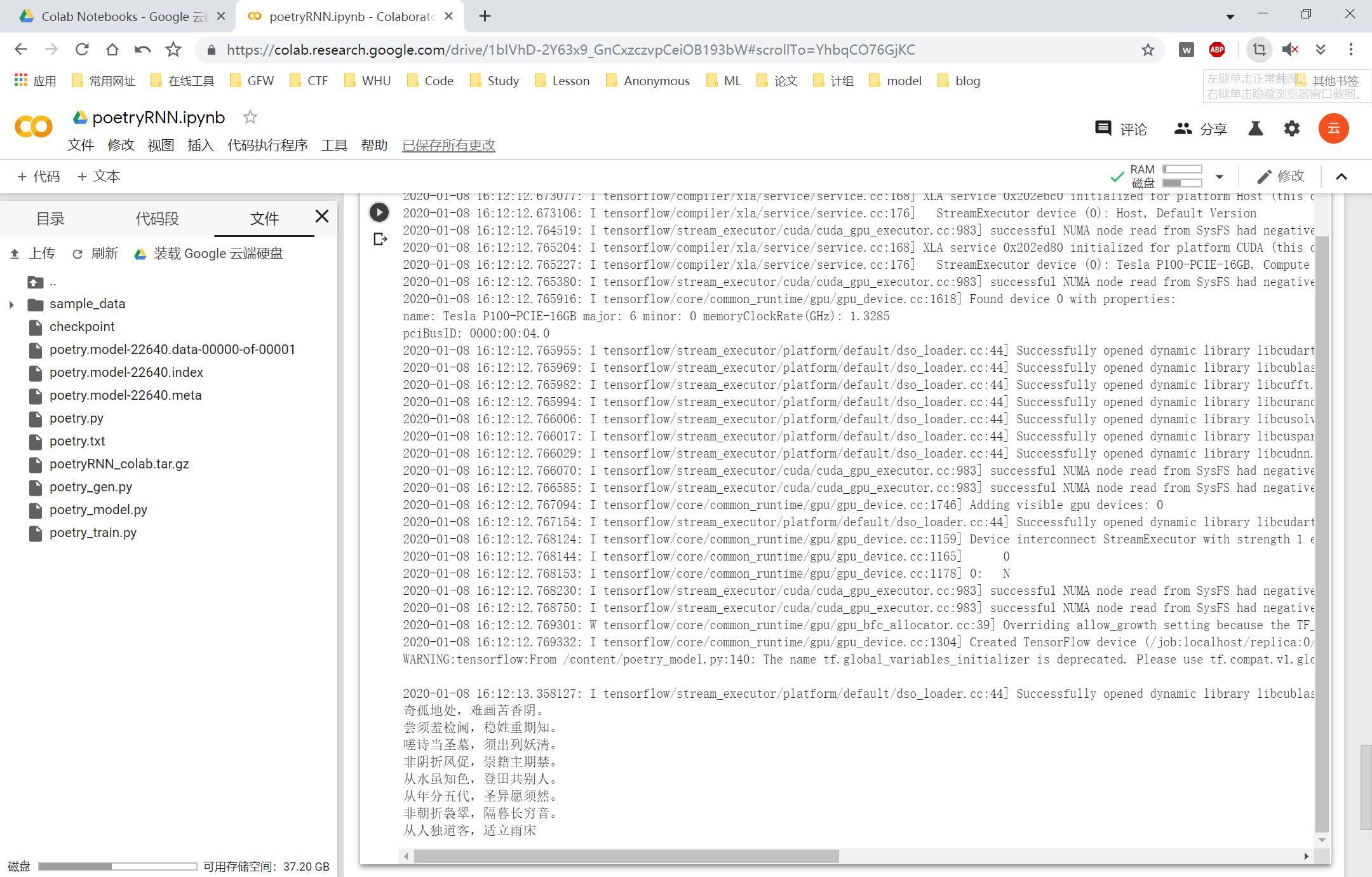
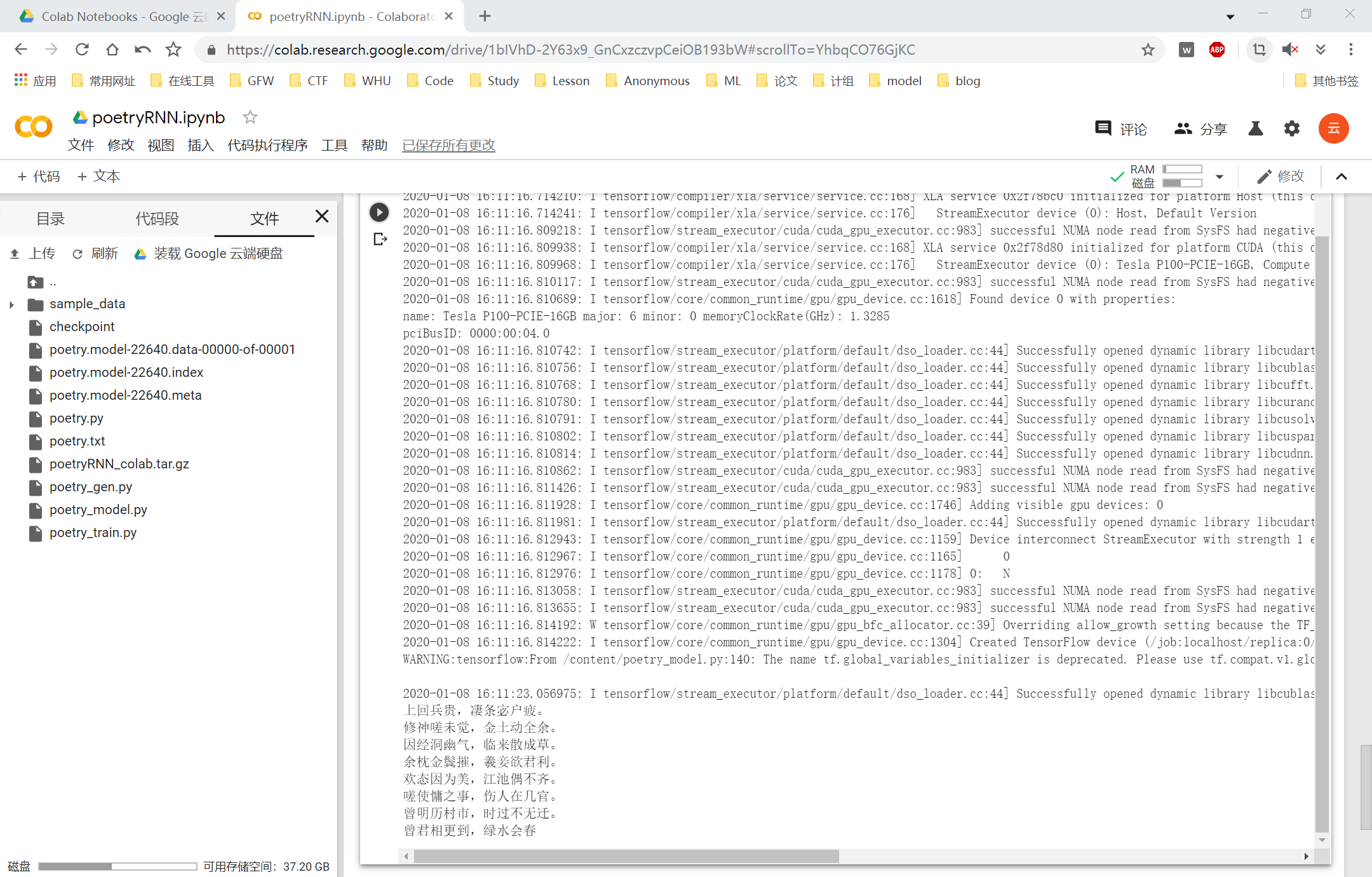
1. RNN训练：训练RNN和训练传统神经网络相似，同样要使用反向传播算法，但会有一些变化。因为参数在网络的所有时刻是共享的，每一次的梯度输出不仅依赖于当前时刻的计算结果，也依赖于之前所有时刻的计算结果。例如，为了计算C:/Users/mcj11/AppData/Local/Temp/qt_temp.RT9132qt_temp时刻的梯度，需要反向传播3步，并把前面的所有梯度加和。这被称作随时间的反向传播（BPTT）。

【实验分析】：

1. 模型训练过程：因为训练量有点大，而自己的笔记本没有安装CUDA，所以仅用CPU训练会比较慢，这里使用的谷歌colab平台提供的GPU进行训练，实测使用Tesla k80训练时间大概在30分钟左右。



1. 测试模型：



可以看出，训练的结果还是比较好的，还是大体能生成较好合乎规范的诗词的。

1. 此次实验按照老师的PPT和参考网上的代码下完成的，整个实验过程遇到的最大的困难就是训练模型，起初硬件条件不够好，训练时间太长，也容易崩溃，后来换了个平台，就顺利的多了，由此我深刻体会到好的硬件条件对深度学习的重要性，不过最重要的是在实验中了解熟悉循环神经网络，实现了诗词创作，对tensorflow等库的使用又更加熟练了，为以后的学习奠定了更好的基础。