# 1、Word Embedding理解

Word Embedding 词嵌入。

**目的：**

将词语的one-hot高维稀疏表达，变换为维数相对低（仍是高维）的稠密表达。

因为将词语用one-hot表达，计算量大（维数非常高），并且词之间的关联无法体现。

**Word2Vector：**

Word Embedding的一种，Google2013年开发的一个开源项目，是目前最成功有效简洁的词嵌入方式。

**Word2Vector有2种机制：**

Skip-gram

输入词去找和它关联的词，计算更快。

CBOW

输入关联的词去预测词。

# 2、代码理解Word\_Embedding.py

本次作业进行Word Embedding使用的代码为Word\_Embedding.py

Word\_Embedding的代码主要分为6步：

1. 读取数据

读取"QuanSongCi.txt"文件内容，并删掉所有非汉字的字符；

1. 建立词汇表

统计输入数据中所有字出现的次数，并保留次数排名前5000的汉字，其余的汉字均视为“UNK”；

1. 为Skip-gram模型生成training batch

生成batch为中心词，labels为上下文。

1. 创建和训练Skip-gram模型

* 随机初始化嵌入矩阵embeddings(vocabulary\_size\*embedding\_size)，vocabulary中每个字都对应一个embedding\_size维的字嵌入向量；
* 通过tf.nn.embedding\_lookup查询输入批次中每个字的嵌入向量；
* 使用噪声对比训练目标来预测目标字词，计算NCE loss；

NCE loss：

1. 可以把多分类问题转化成二分类，大大提高计算速度；
2. 将所有单词分为两类，正例和负例，word2vec中只需给出上下文和相关的正例，tf.nn.nce\_loss()中会自动生成负例。
3. 训练模型

迭代步数取150001

运行计算图

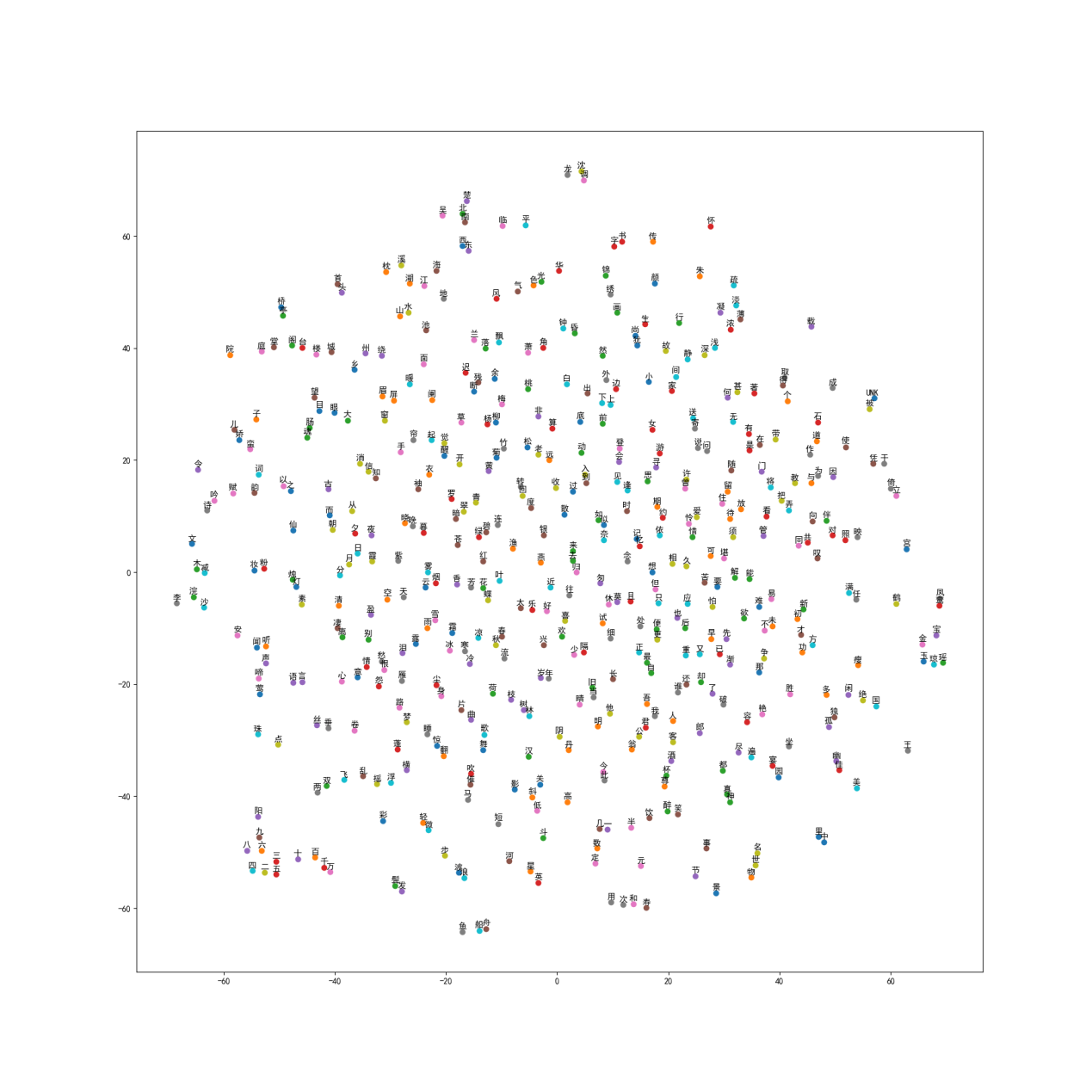
通过np.save('embedding.npy', final\_embeddings)保存最终生成的embeddings。

1. 可视化学到的字词嵌入

使用 t-SNE 降维技术将字词嵌入投射到二维空间；

通过设置plt.rcParams['font.sans-serif'] = ['SimHei']，使matplotlib绘制的图能够正常显示中文。

# 3、字词嵌入图

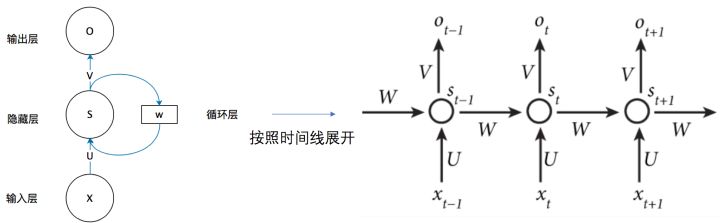


如图所示，可以看出图片中意义接近的词，如数字等（图中左下角），距离比较近（一这个数字是个特例，离其他数字比较远）。

# 4、RNN理解

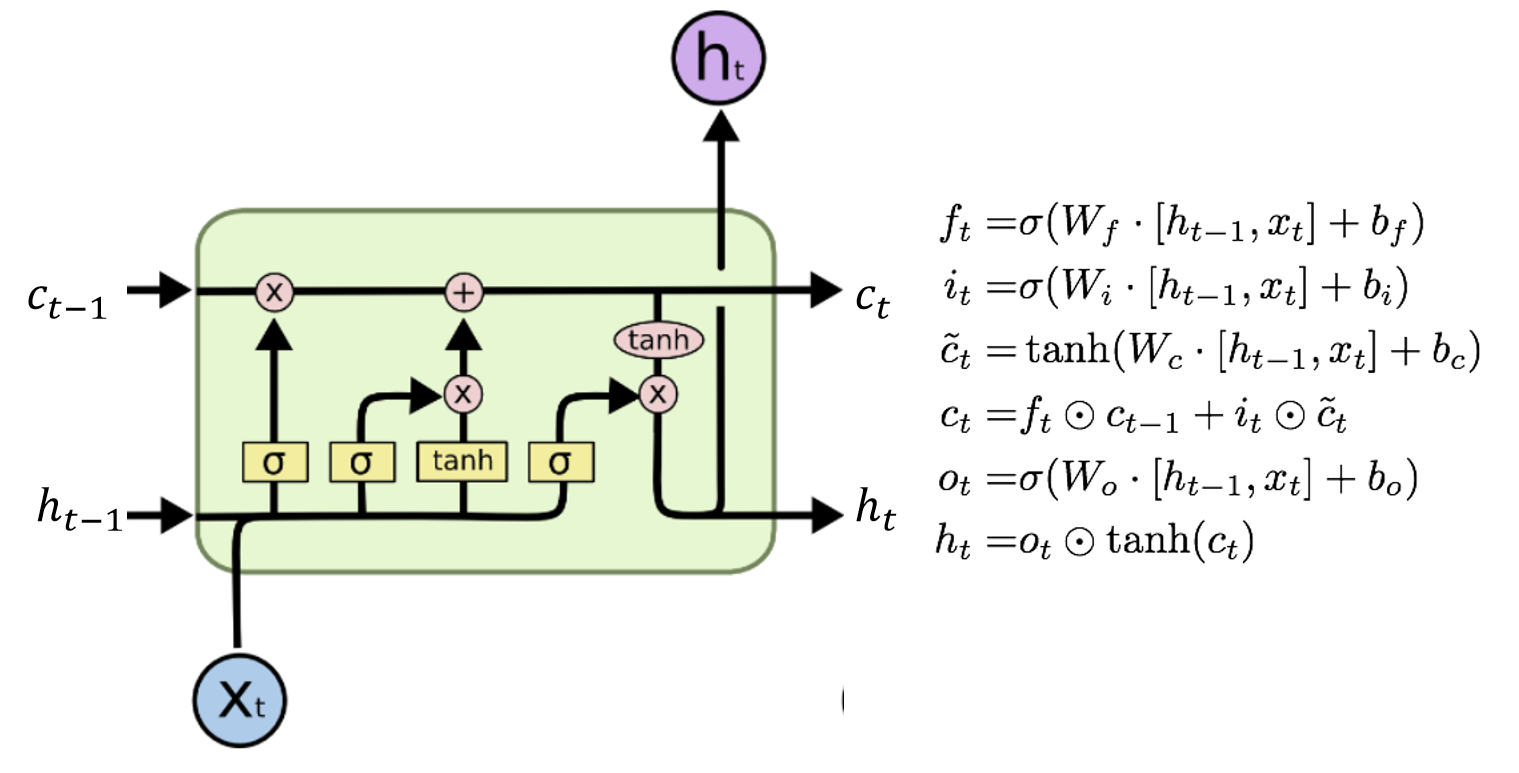
RNN即Recurrent Neural Network，是循环神经网络，具有短期记忆能力，适用于文本和视频相关应用。

RNN网络模型结构如下图所示



RNN具有时许概念，每个数据产生时即会输入网络中，在网络内部存储之前输入的所有信息作为短期记忆。

本次作业使用的是RNN中的LSTM模型，如下图所示

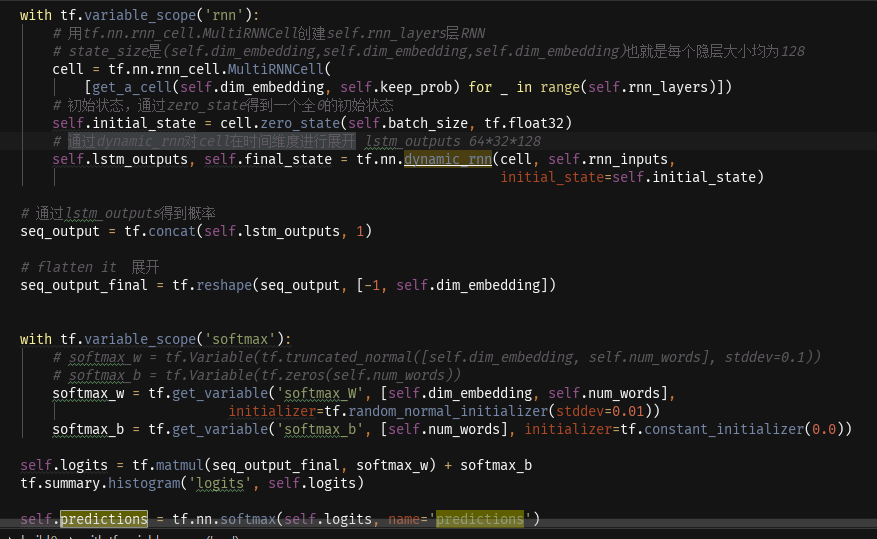


# 5、RNN代码

作业代码主要修改model.py、train.py和utils.py.

**model.py**

构建多层LSTM网络，用tf.nn.rnn\_cell.MultiRNNCell创建self.rnn\_layers层RNN，通过zero\_state得到一个全0的初始状态，通过dynamic\_rnn对cell在时间维度进行展开。



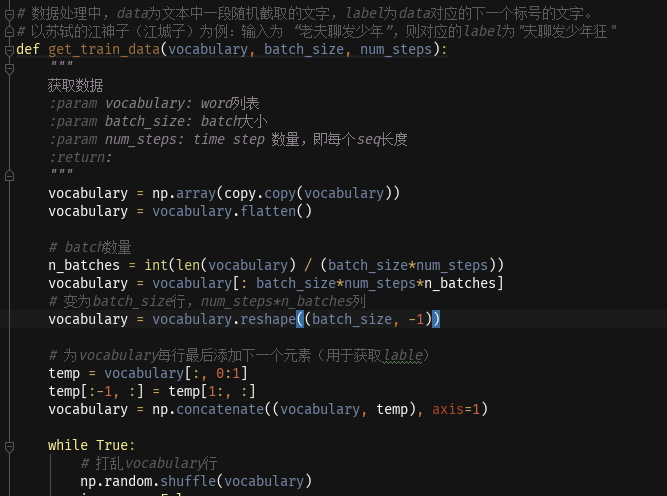
**train.py**

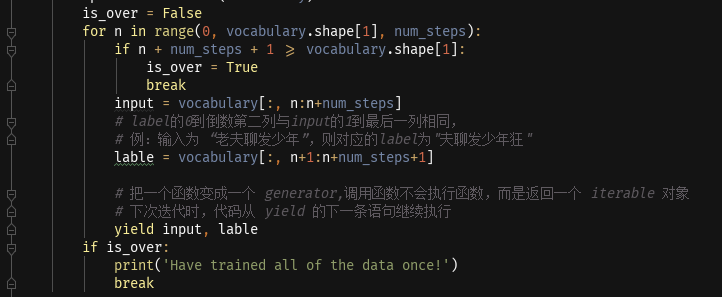
读取训练数据，feed给模型，需要注意的是，每次训练完成后，把最后state的值再赋值回去供下次训练使用，代表了state在时间轴上的传递。



**utils.py**

数据预处理，生成输入数据，data为文本中一段随机截取的文字，label为data对应的下一个标号的文字。以苏轼的江神子（江城子）为例：输入为 “老夫聊发少年”，则对应的label为"夫聊发少年狂"。





# 6、RNN模型训练心得

（1）一开始建立模型时，输入了之前做好的embedding矩阵，并不随模型一起训练，导致模型训练结果不好，几个epoch后，输出一直是UNK，令embedding矩阵随模型一起训练和更新就正常了；