# NLP Lab #4 - Skip-thoughts 模型

## Part #1 实验简介

本次实验要求使用 Skip-thoughts 架构，在 Large Movie Review Dataset 英文语料上进行文本向量学习，并运用文本向量对于文本的情感色彩进行分类，以评估该架构的学习能力。

本次实验的重心在于测试不同训练策略的组合，以获得对于 Skip-thoughts 架构的感性认知。

## Part #2 技术细节

### a) 模型选择

在本次实验中，本人使用了 DSXiangLi/Embedding: Embedding模型代码和学习笔记总结 (github.com) 仓库中对于 Skip-thoughts 的实现，原因主要有：

1. 实现简单，便于理解，同时作者在其博客中有详细的实现解释；
2. 实现时间较近，使用的 Tensorflow 库的版本较新（很多实现基于 Tensorflow v1，语法变化很大）；
3. 支持 GPU 加速（很多其它版本的实现并不支持 GPU 加速或者 GPU 加速环境难以配置）。

### b) 文本预处理

在文本预处理中，我们需要对来源于多个文件的语料进行学习，本人采用逐个打开每个文件，将所有语料合并的方法来处理所有文本。值得注意的是，在实际实现中，本人最初使用的是每读取一段新的语料，即将其插入列表的存储方式。但是本人发现这种实现策略在运行时效率很低，于是最后本人使用了 Python 的 yield 语法，利用 generator 迭代器，大大提高了文本收集的效率。

从所有文本中成功提取文本后，还需要做以下文本操作：

1. 去掉文本中所存在的类似于 `<...>` 的类 HTML 字符串；
2. 将文本中所有的英语字母小写化；
3. 将文本按照完整的句子逐一拆分；
4. 由于仓库的实现原因，需要将训练所需的文本按照对应关系放入到输入数据和待预测数据两个文件中，即如果 $s_i$ 在训练数据的第 $k$ 行，则待预测数据中 $s_{i-1}$ 和 $s_{i+1}$ 被放在第 $2k$ 和第 $(2k+1)$ 行。

### c) 文本向量的获取方式

本次实验的最终目的是获得文本向量以对文本进行分类，但是我们的 Skip-thoughts 模型只能获得句子的向量表示，而一段文本通常由多个句子组成。如何用一段文本中多个句子的向量得到整个文本的向量呢？在本次试验中，由于不同文本中的句子数量不一样，所以无法使用向量拼接的方法。为了得到长度相同的文本向量，本人采用的方法是取所有文本向量的平均值。

### d) 模型结构的选择

根据原论文的设计，作者最终给出的句子向量是两种模型给出向量的拼接结果，即采用了单层 GRU 和双层 GRU 的编码器分别给出的编码结果的拼接向量。在本次实验的时间关系，本人考虑采用一种模型的编码结果，进一步考虑到原论文中提到双层 LRU 的结果优于单层的结果，本人最终选择了双层 GRU 的编码结果。

### e) 无标记文本的处理

考虑到在上次实验中，对于无标记文本的预测无法有效帮助最终的分类任务。于是，本人最终不训练，也不预测无标记文本的向量。

### f) 逻辑回归模型

本人使用 sklearn.linear_model.LogisticRegression 模型作为本次实验总使用到的逻辑回归模型。在参数选择中，本人没有充分搜索此模型的参数，仅使用了模型的默认参数。

## Part #3 实验结果与分析

基于四种学习策略所获得的文本情感分类模型在测试集上的分类准确率如下表所示：

| **分类准确率*（%）** | **dim = 100** | **dim = 200** |
|---|---|---|
| train | 66.863 | 68.941 |
| train + test | 70.097 | 71.255 |

每个数据项仅对应一次 Skip-thoughts 模型训练，逻辑回归模型的分类准确率在小数点后三位小数内无变化；

### 结果分析

1. 从结果来看，是否在待预测数据集上预测是更为重要的因素；向量维度对于分类准确率的贡献较小，原因可能是句子情感在向量空间中存在稀疏表示。
2. 模型的分类结果相较于 Doc2vec 结果较差，本人认为一个重要原因是，本人对多个句子向量转化为文本向量的策略存在一些不足。