**项目的优缺点和模型分析报告**

组名：达芬奇密码

队员：李嘉能、赵君和、盛亮倩、项洁洁

1. **项目优缺点**

**1.1优点**

1. 摘要生成效果良好，与全文与标题具有明显相关性；

2. 对各种标点符号有良好的处理，摘要中出现标点符号有问题的情况较少。

3. 项目前端简洁美观，易于使用；

4. 具有网站部署，有服务器提供访问支持。

**1.2缺点**

1. 因为句子按照逗号分开计算，排序结束后句子里面某些部分会因为得分低而被删除，所以摘要的句子有时候会有点奇怪，比如有时摘要结尾出现逗号；

2. 此摘要的生成技术只能是把重要性高的原句选出，而不能缩句。

**2.项目分析**

**2.1数据预处理**

1. 维基百科中文语料库 <https://dumps.wikimedia.org/> 下载最新版本（增量更新，可以直接下更新数据）；

2. 数据文件是xml.bz2 需要运用一个包 wikiextractor；

地址：https://github.com/attardi/wikiextractor

3. 运行python WikiExtractor.py -b 500M -o zhwiki filename

4. 正则提取之后，wiki数据需要进行简繁转换；

PS：简繁转换包 opencc c = OpenCC（‘mode’）mode=tw2s、hk2s 等

5. 将处理好的文件进行jieba分词后保存为token文件；

PS：1）即使保存时指定utf-8编码，windows也会有问题，所以保存后一定要另存编码。

2）大文件的打开可以用PilotEdit、Notepad++等。

**2.2模型构建**

|  |
| --- |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |

1. 导入预先训练好的词向量文件整理成字典的形式方便后续调用 2. 导入新闻的csv文件

3. 利用第二课学习的内容计算每个词的词频，存在了word\_frequency这个txt里面，后续直接从里面拿，就不用再重新计算

4. split\_sentences\_2此函数作用是对整段文字进行句子分割

5. SIF\_sentence\_embedding函数是用到了指导书里面提到的sif方法

6. knn是后续做平滑处理用的函数

7. get\_corr函数计算每个句子的分数，分数包括了：题目的相似程度和整段文字的相似程度加权以及knn平滑，

8. 最后get\_summarization两个函数就是综合了前面的过程，输出摘要，摘要的长度也是可以修改的。

2.3参数调参

1. 对一些转折词，连词我们进行了减少相关度的处理（句子全文相关度=句子全文相关度\*75%）；

2.考虑了和位置有关的句子重要性：对每个段首句子进行了标注，在knn平滑函数中，增加了句子的全文相关度；

3. 对get\_corr()和knn平滑中参数进行调整；

**3.改进方向**

1. 计算全文相关度时，我们可以用其他句子重要性排序方法，进行组合；
2. 对摘要进行缩句，或许能采用LSTM等神经网络进行语句生成。