人工智能与自然语言处理
第六期-项目1

非监督文本自动摘要模型的构建

开课吧人工智能学院

CONTENTS

本指导手册将指导大家完成第一个课程项目

1. 项目背景介绍

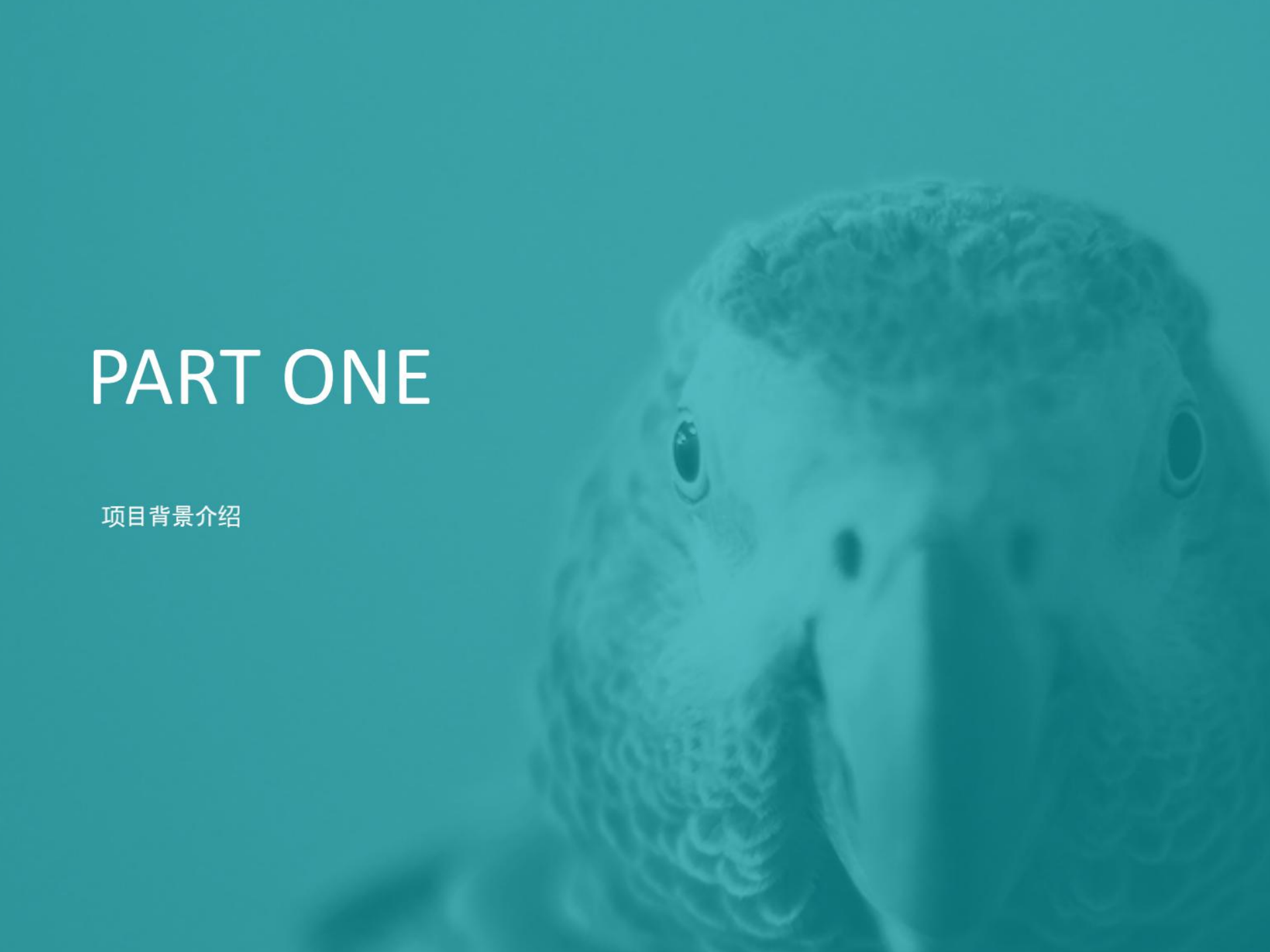
2. 关键技术点

3. 项目思路指导

4. 分组安排

PART ONE

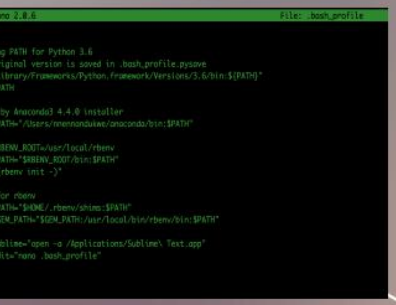
项目背景介绍



自动摘要问题简介

自动摘要问题是NLP领域的一个经典问题，简单的说，就是输入一段长文字，输出对这段长文字的一个总结概要。 在新闻，语音播报、文档信息提取、公司报表、上市公司分析等等领域具有很多的应用场景。

自动摘要模型的问题定义



实验室模型

公司项目和实验室模型的区别

这个项目是来源于企业实际的项目，企业实际的项目和咱们实验室做的模型的区别有这么几个：

1. 相关的数据预处理会比较多；
2. 模型的方法要尽可能简单、快速；
3. 要用多种方法融合来解决问题
4. 要给出比较好的交互效果

>

企业实际项目

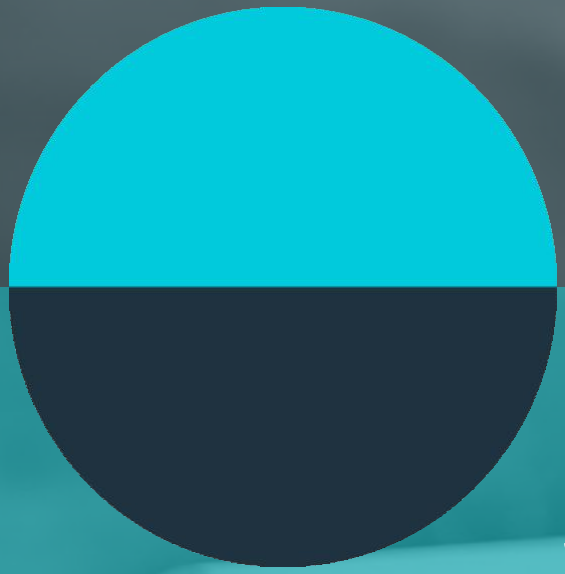
良好的数据可视化

所以，我们除了要完成模型部分，我们还要使用Flask, Bottle, Bootstrap等工具，进行良好的交互和数据可视化。

可视化是非常重要的！

Step1: 数据预处理

Step2: 核心模型的搭建



Step3: 数学模型调优

Step4: 可视化

所以，我们的项目分为这么4个部分

>Step1: 数据预处理

Step2: 核心模型的搭建

Step3: 数学模型调优

Step4: 可视化

所以，我们的项目分为这么4个部分

Step1: 数据处理部分

- 数据处理部分我们需要用到两个数据：
  - 1. 维基百科中文语料库；
  - 2. 汉语新闻语料库；
- 其中，维基百科中文语料库+汉语新闻语料库进行词向量的训练，汉语新闻语料库亦是我们此次处理的数据源，也就是说，我们这一次要进行的就是新闻数据的自动摘要

Step1-1: 维基百科中文语料的处理

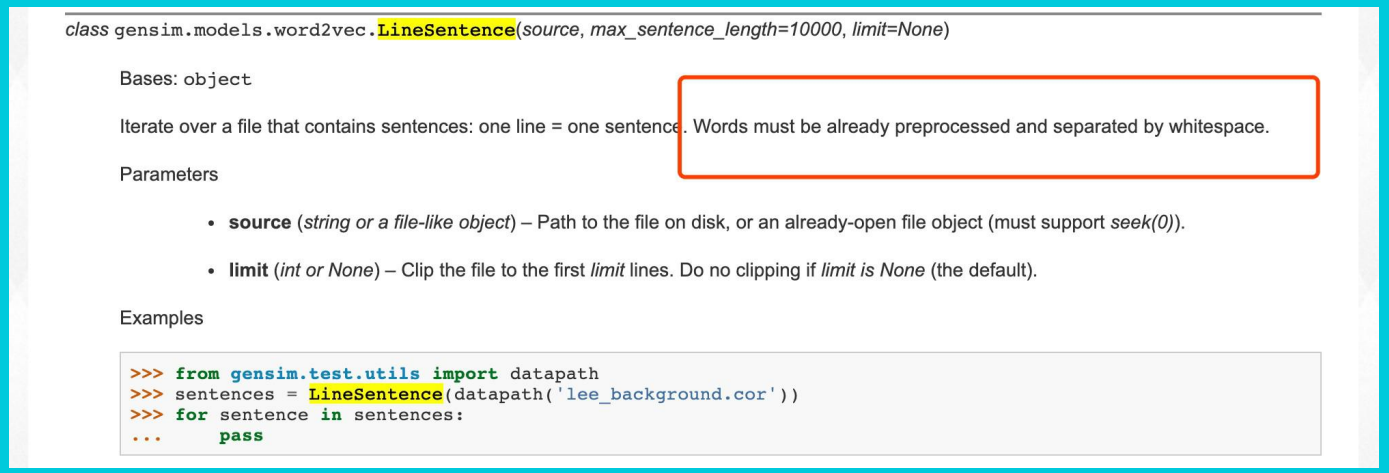
- 1. 数据库下载链接：
  - https://ftp.acc.umu.se/mirror/wikimedia.org/dumps/zhwiki/20191120/zhwiki-20191120-pages-articles-multistream.xml.bz2
  - 数据提取：
  - 维基百科的信息结构比较负责，我们需要用到专门的提取工具：https://github.com/attardi/wikiextractor

Step1-2: 汉语新闻语料库的处理

- 1. 下载地址
  - https://github.com/Computing-Intelligence/datasource/blob/master/export_sql_1558435.zip
- 2. 数据提取
  - 按照第一节课，第二课节的内容，进行数据清除和token操作
- 3. 数据清理
  - 将content内容专门存在一个单独的文件中

Step1-3: 使用Gensim训练词向量

- 1. 下载安装Gensim: $ pip install gensim
- 2. 将Step1-1和Step1-2Th成的结果重新整理为Gensim能够接受的数据格式:
  - https://radimrehurek.com/gensim/models/word2vec.html



class gensim.models.word2vec.**LineSentence**(*source*, *max_sentence_length=10000*, *limit=None*)

Bases: object

Iterate over a file that contains sentences: one line = one sentence. Words must be already preprocessed and separated by whitespace.

Parameters

- **source** (*string or a file-like object*) – Path to the file on disk, or an already-open file object (must support *seek(0)*).
- **limit** (*int or None*) – Clip the file to the first *limit* lines. Do no clipping if *limit is None* (the default).

Examples

```
>>> from gensim.test.utils import datapath
>>> sentences = LineSentence(datapath('lee_background.cor'))
>>> for sentence in sentences:
...     pass
```

- 3. 使用Gensim训练词向量

Step1-4: 测试词向量的效果

- 1. 词向量的语义相似性

```
In [7]: model.wv.most_similar('勇敢')

[('勇于', 0.5452967882156372),
 ('坚毅', 0.544731855392456),
 ('坚强', 0.5447058081626892),
 ('勇气', 0.537638783454895),
 ('果敢', 0.5369020700454712),
 ('善良', 0.5327401757240295),
 ('坚忍不拔', 0.5091272592544556),
 ('豁达', 0.5033485293388367),
 ('真诚', 0.5024720430374146),
 ('追爱', 0.4999522566795349)]
```

```
In [31]: model.wv.most_similar('美女')

[('帅哥', 0.5578837990760803),
 ('校花', 0.5232111215591431),
 ('女演员', 0.5189783573150635),
 ('金发碧眼', 0.5069225430488586),
 ('超模', 0.502299964427948),
 ('舞女', 0.5002666711807251),
 ('甜心', 0.4993056654930115),
 ('男模', 0.4970622658729553),
 ('女主播', 0.49386513233184814),
 ('嫩模', 0.4927775263786316)]
```

Step1-4: 测试词向量的效果

- 2. 词向量的语义线性关系
  - 创建以下函数

```
: def analogy(x1, x2, y1):
      result = model.most_similar(positive=[y1, x2], negative=[x1])
      return result[0][0]
```

  - 然后测试:
    - analogy('中国', '汉语', '美国')
    - analogy('美国', '奥巴马', '美国')

Step1-4: 测试词向量的效果

- 3. 词向量的可视化:
  - 我们使用t-sne进行高维向量的可视化
  - https://www.kaggle.com/jeffd23/visualizing-word-vectors-with-t-sne

- Tips:

  1.你可能需要减少词向量的单词量

  2.你可能需要让matplotlib能显示中文

Step1: 数据预处理

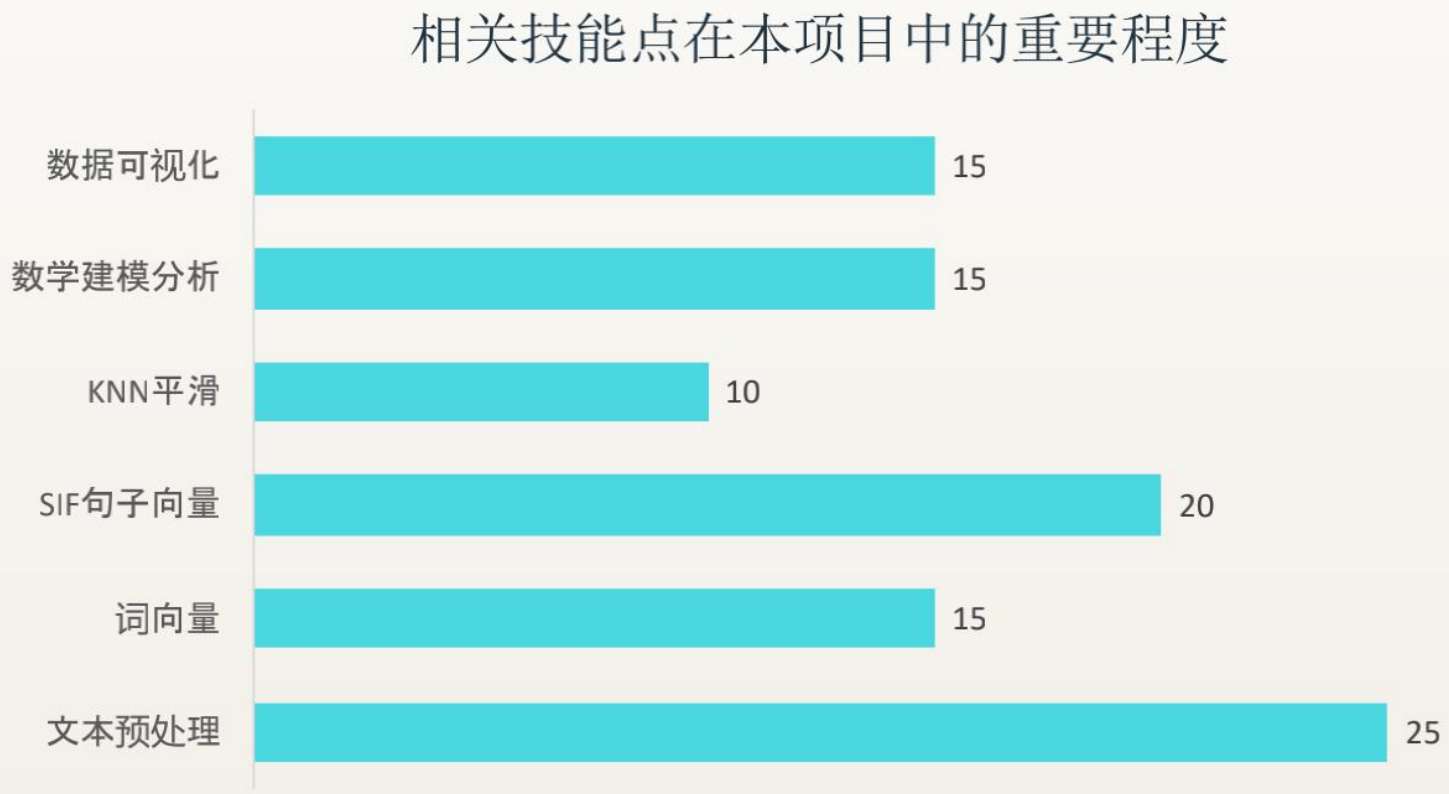
>Step2: 核心模型的搭建

Step3: 数学模型调优

Step4: 可视化

所以，我们的项目分为这么4个部分

核心模型的搭建


相关技能点在本项目中的重要程度
数据可视化
15
数学建模分析
15
KNN平滑
10
SIF句子向量
20
词向量
15
文本预处理
25

1. 句子的SIF向量化

- 我们在前一章节，已经完成了单词的向量化。基于单词的向量化，我们使用普林斯顿大学提出来的SIF方法，进行句子的向量化：

  原文地址：https://openreview.net/pdf?id=SyK00v5xx

  大家参照文章中的方法，实现该算法

  - 注：sklearn 实现PCA的方法：https://scikit-learn.org/stable/modules/generated/sklearn.decomposition.PCA.html

  我们对每一篇新闻中的每一句话，都算出来它的向量: Vs_j,

  我们把这篇文章，当做一个完整长句, 算出来它的向量: V_c

  我们求得这边文章标题的长句，算出来它的向量 V_t，；

  思考题：包含书名号，引号等符号的句子，该怎么切分句子？

2. 依据句子向量，为每个句子赋予权值

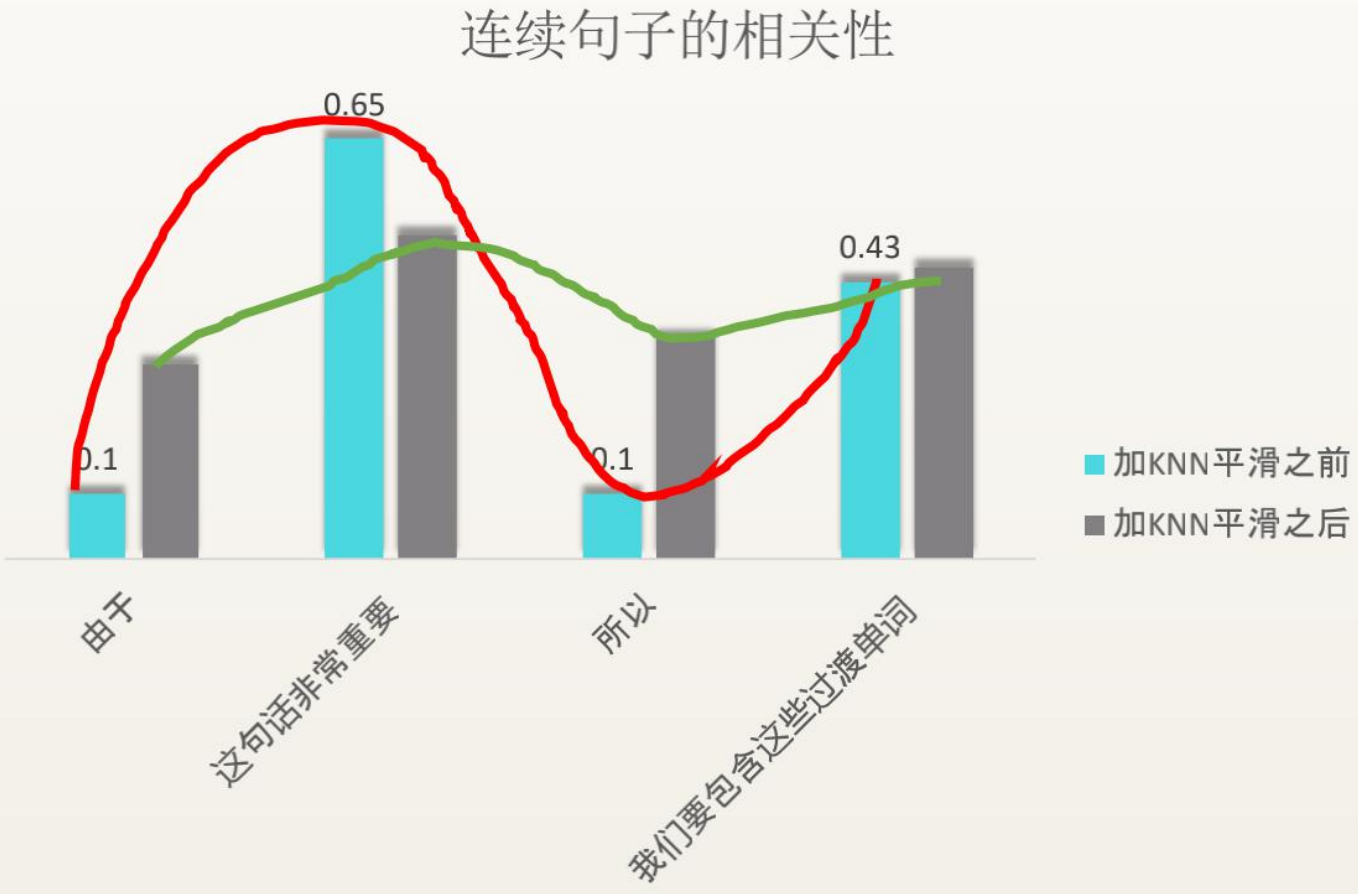
- 当我们获得了每个句子的向量: $< Vs_0, Vs_1, Vs_2, \ldots Vs_n >$
- 获得了每个文章的标题向量: V_t
- 获得了每个文章全文的向量: V_c
- 对于每一个 Vs_i, 需要各位同学涉及一个数学模型: f, 该 f 接受 f(Vs_i, V_t, V_c) 输出一个 0 ~ 1 的值，表示这句话与全文的相关度 C_i
- 我们按照C_i进行排序，我们取出来Top_n，就能过获得语义上最相关的句子了

3. KNN平滑

- 上文中，我们实现了每个句子与文章的语义相关性 C_i，但是于此而言，带来一个问题，就是如果某个句子的C_i太高，而它前后的句子 C_i 过低，只拿出来 C_i 对于的句子会让句子变得不通顺。

- 我们使用KNN的思想，在求得了C_i 之后，其真实的C_i ，是这个C_i周围的若干C_j与自身的C_i的加权求和

KNN连续句子相关性的平滑



4. 获得end-to-end模型

- 把之前的步骤合并起来，我们就能过生成一个函数summarize(content, title), 这个函数输出文章的内容和标题，然后输出是一个string，这个string是和文章意义最相关的N个句子，我们作为摘要内容输出即可。

5. 使用flask, bottle, Bootstrap等进行可视化

- 最后一步，我们需要使用flask，bottle，bootstrap进行数据可视化。
- Flask或者bottle是一个简便的Python后端模型，能够在半个小时之内把我们的模型变成通过互联网访问的项目
- Bootstrap是twitter出的简易但是功能强大的网页前端框架，可以很快的做出来好看的页面。
- 我们每个项目分组里，尽可能有一个熟悉后端或者前端开发的同学，这样实现起来会很快速。

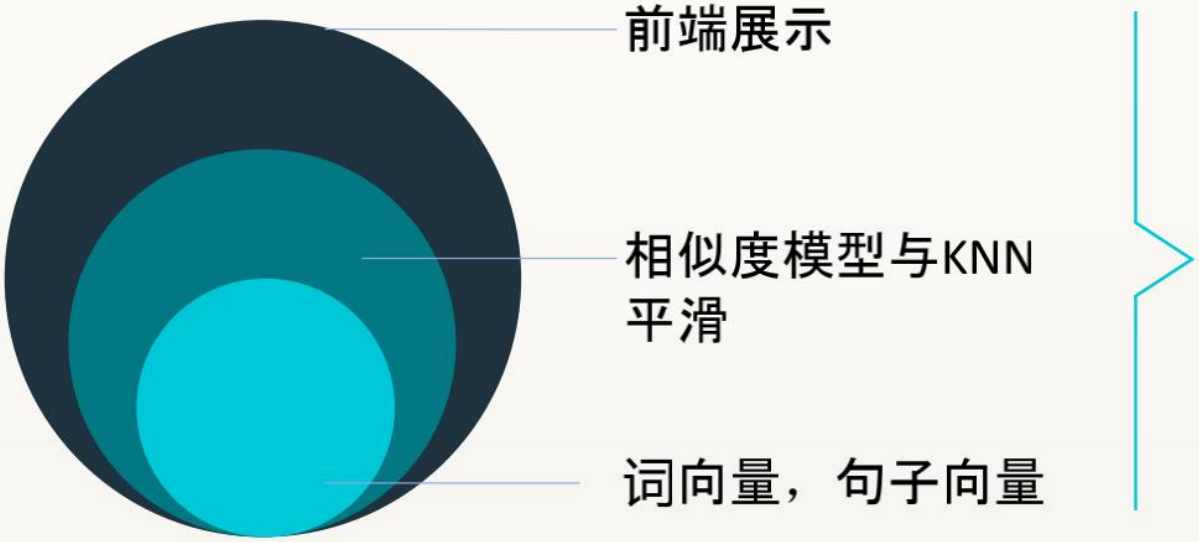
最终的效果

- 如果做完以上步骤，你应该能做出来这样的一个网页应用

网易娱乐7月21日报道 林肯公园主唱查斯特·贝宁顿Chester Bennington于今天早上，在洛杉矶 帕洛斯弗迪斯的一个私人庄园自缢身亡，年仅41岁。此消息已得到洛杉矶警方证实。
洛杉矶警方透露，Chester的家人正在外地度假，Chester独自在家，上吊地点是家里的二楼。一说是一名音乐公司工作人员来家里找他时发现了尸体，也有人称是佣人最早发现其死亡。
林肯公园另一位主唱麦克 信田确认了Chester Bennington自杀属实，并对此感到震惊和心痛，称稍后官方会发布声明。Chester昨天还在推特上转发了一条关于曼哈顿垃圾山的新闻。粉丝们纷纷在该推文下留言，不相信Chester已经走了。
外媒猜测，Chester选择在7月20日自杀的原因跟他极其要好的朋友、Soundgarden（声音花园）乐队以及Audioslave乐队主唱Chris Cornell有关，因为7月20日是Chris Cornell的诞辰。而Chris Cornell 于今年5月17日上吊自杀，享年52岁。Chris去世后，Chester还为他写下悼文。
对于Chester的自杀，亲友表示震惊但不意外，因为Chester曾经透露过想自杀的念头，他曾表示自己童年时被虐待，导致他医生无法走出阴影，也导致他长期酗酒和嗑药来疗伤。目前，洛杉矶警方仍在调查Chester的死因。
据悉，Chester与毒品和酒精斗争多年，年幼时期曾被成年男子性侵，导致常有轻生念头。Chester生前有过2段婚姻，育有6个孩子。
林肯公园在今年五月发行了新专辑《多一丝曙光One More Light》，成为他们第五张登顶Billboard排行榜的专辑。而昨晚刚刚发布新单《Talking To Myself》MV。

林肯公园主唱查斯特·贝宁顿Chester Bennington，今天早上，在洛杉矶帕洛斯弗迪斯的一个私人庄园自缢身亡，年仅41岁。粉丝们纷纷在该推文下留言，不相信Chester已经走了。外媒猜测，Chester选择在7月20日自杀的原因跟他极其要好的朋友、Soundgarden乐队以及Audioslave乐队主唱Chris Cornell有关，因为7月20日是Chris Cornell的诞辰。今年5月17日上吊自杀，享年52岁，去世后，Chester还为他写下悼文。

项目整体的层级



如何提交项目

- 首先需要进行项目分组，我们会统一组织大家进行项目分组
- 提交项目应该是一个压缩包，该压缩包包含以下内容：
  - 1. 项目源代码（不需要包含数据）
  - 2. 项目的PPT效果展示
  - 3. 你的参数调整记录表
  - 4. 该项目能够访问的网站链接
  - 5. 该项目的优缺点和模型分析报告
- 之后将该Zip压缩包上传到开课吧后台， 一组只要有一位同学提交即可。
- 项目接受截止日期：2020.03.15

THANK YOU