含Emoji的垃圾文本检测系统

SUN YAT-SEN UNIVERSITY

组员：邓凯纳、邓博高、庞子良

目录 CONTENTS

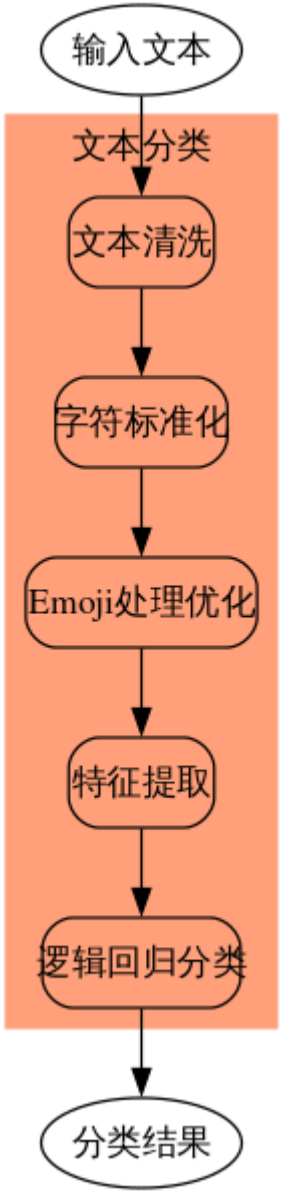
- 设计思路
- 结果展示
- 优化说明

设计思路

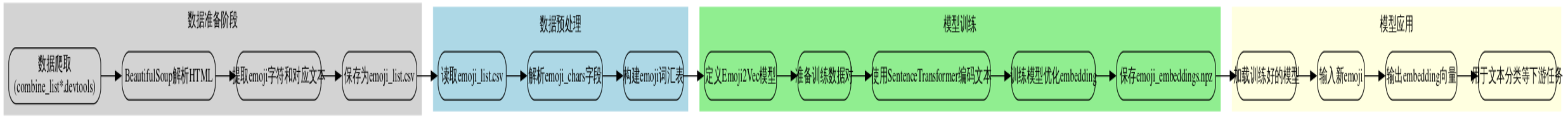
整个系统可以分为三个模块：

主流程 + emoji语义向量提取 + 汉字字符向量提取。

主流程从原始文本清洗开始，经过字符标准化和Emoji优化处理，
提取emoji和汉字各自的特征向量后，使用逻辑回归进行分类预测。



emoji语义向量提取

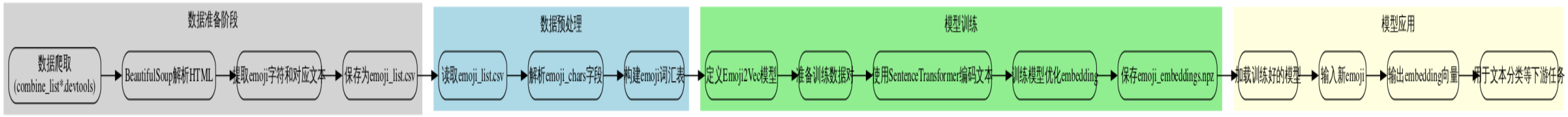


对于含有emoji的文本检测，我们需要训练一个模型，能够将 emoji 的 ID 映射到与其中文词意相匹配的向量表示。

可以将emoji的表征向量训练流程概括为上述的步骤：

数据准备和预处理→模型训练→模型应用

emoji语义向量提取



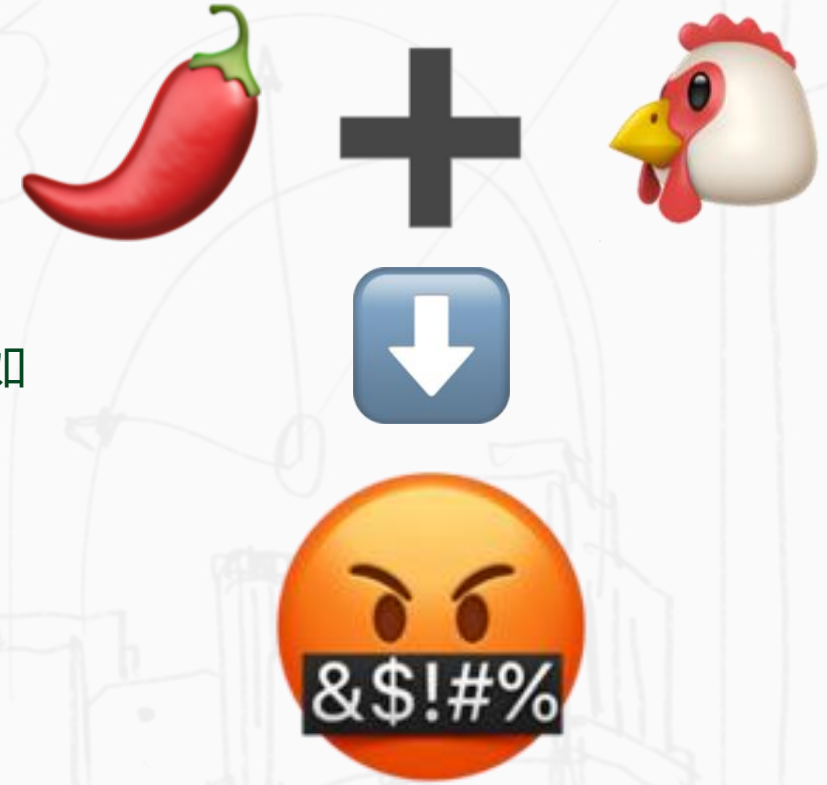
可以将emoji的表征向量训练流程概括为上述的步骤：

数据准备和预处理→模型训练→模型应用

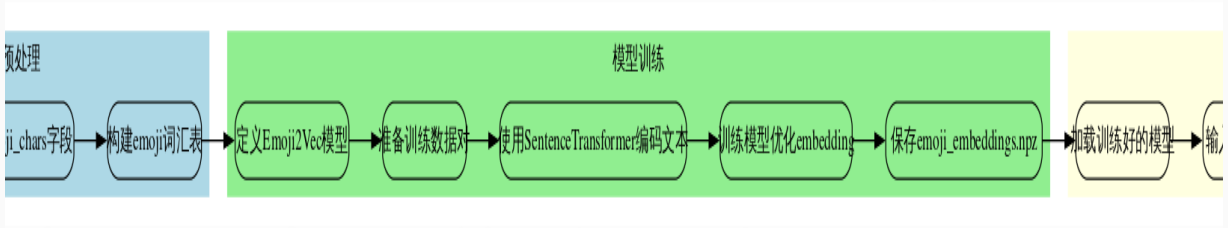
首先是数据收集，为解决缺少emoji数据集问题，我们需要编写**爬虫**从网页获取原始数据，要考虑处理**肤色变体**等**复合emoji**编码转换，并进行数据清洗后用于模型训练。

为了使模型能够识别 emoji 的中文含义，包括**谐音字和变体**，我们寻找形如 {"**":"辣鸡"} 的联想数据集进行训练。采用 **Skip-Gram** 风格的 **Emoji2Vec** 模型，将 emoji 映射到低维向量空间，使语义相似的 emoji 彼此接近。

因此，我们构造 {"*":"辣鸡"} 和 {"*":"辣鸡"} 的数据对进行训练。



emoji语义向量提取



可以将emoji的表征向量训练流程概括为上述的步骤：

数据准备和预处理→模型训练→模型应用

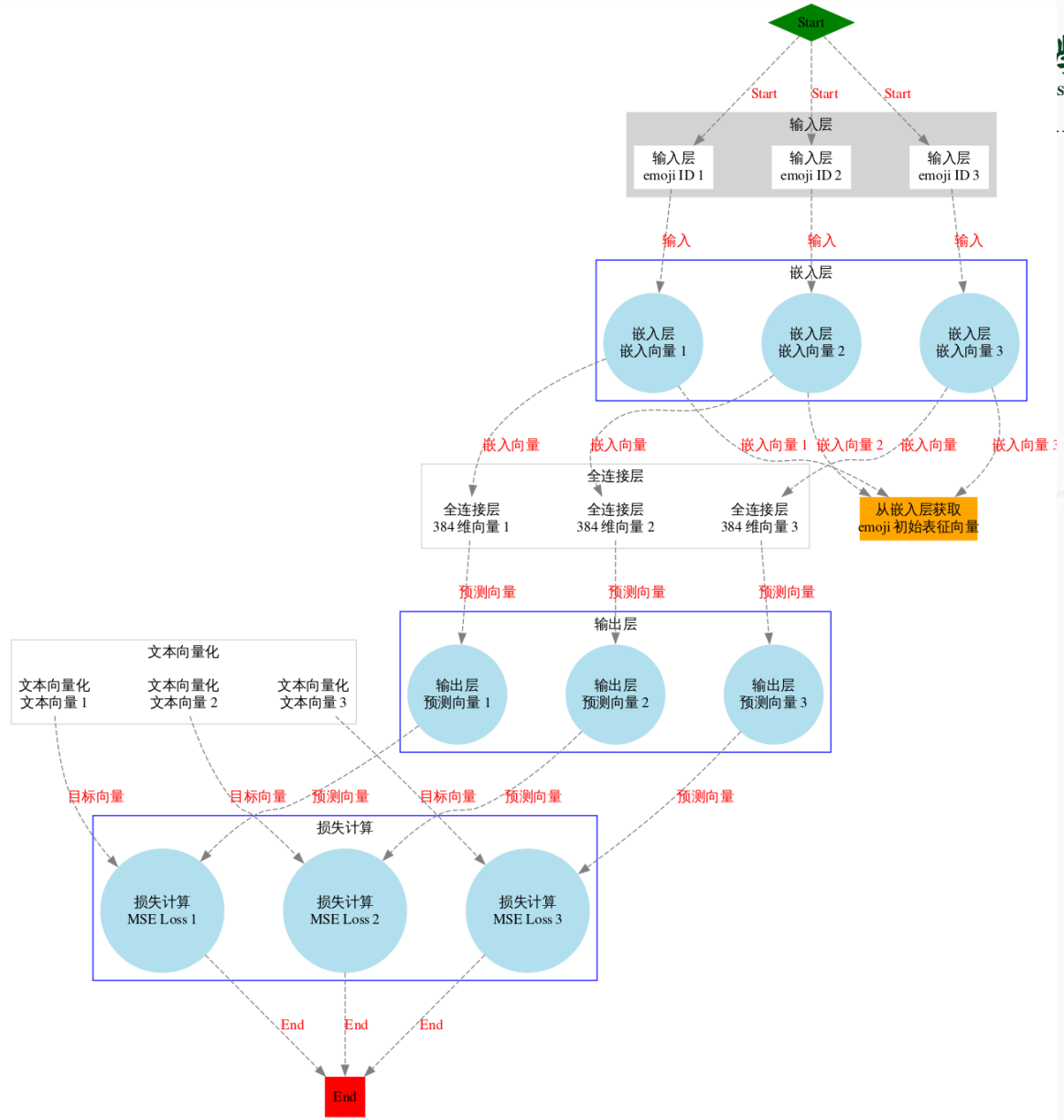
该神经网络模型主要有四层结构：

输入层：输入 emoji 的 ID

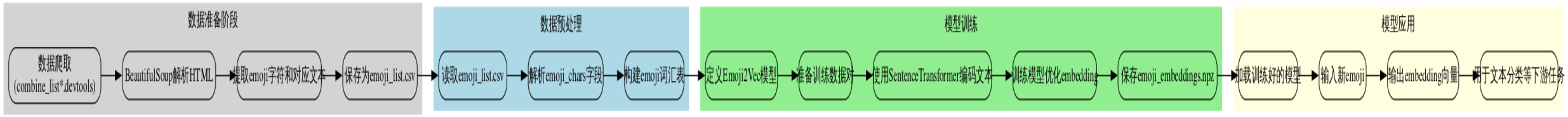
嵌入层：将 emoji 的 ID 映射到低维的稠密向量空间，使用PyTorch 的 **nn.Embeddin** 层模型实现，输出一个更低维的嵌入向量来表示对应的emoji

全连接层：将嵌入向量转换为与文本向量相同的维度（384 维），以便与文本向量进行比较

输出层：输出emoji的表征向量



emoji语义向量提取



可以将emoji的表征向量训练流程概括为上述的步骤：

数据准备和预处理→模型训练→模型应用

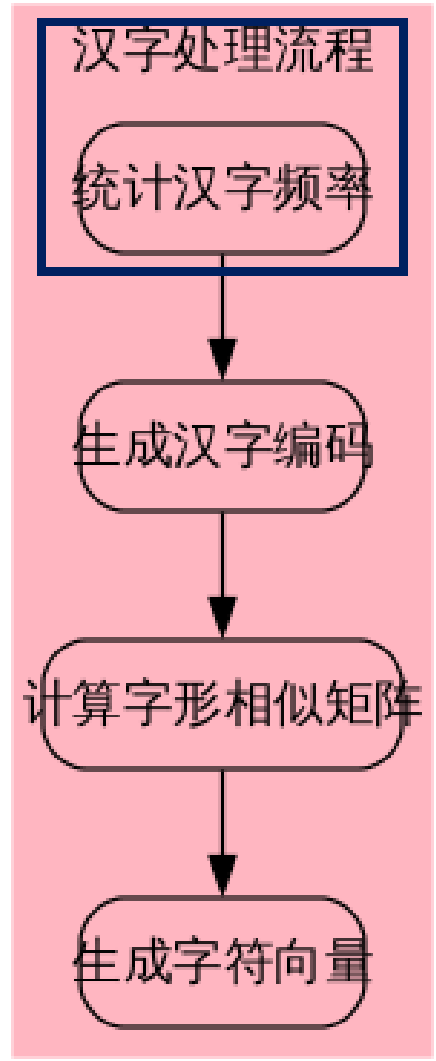
我们在主流程中需要通过训练好的emoji模型来将文本中的emoji转换为对应的特征向量，结合汉字向量来实现后序的分类任务。

汉字字符向量提取

该模块实现从文本数据中提取有用的特征，生成用于模型训练的嵌入向量。需要统计汉字并计算字形相似矩阵生成**Word2Vec**初始字符向量。

汉字字符向量的提取流程同样从**数据收集**的文本清洗开始，中文文本虽无数值缺失或异常，但存在**空白文本**、**错误编码**和**意外字符**等问题，需处理这些缺失值和异常值。

接着需要定义汉字**编码**，并通过计算**字符相似性**矩阵最终得到汉字的**特征向量**。



汉字字符向量提取

接着，我们需要定义汉字的特征来进行提取和分类，这步可以从汉字独有的**字音**和**字形**出发。

汉字的字形通常包括以下三个要素：
汉字结构、汉字形状和汉字笔画数

汉字字音可以直接用四个要素概括：**声母、韵母、补码以及声调**。前两者很好理解，声调也可以用数字1-4表示，主要是**补码**，这个一般作用于于声母和韵母之间还有一个辅音的情况（如 guang中的u就是补码）。

将上述两种编码组合起来，就可以得到汉字的完整编码。
接着我们需要计算两个汉字之间的**相似度**，由此类比用户对变体汉字的联想过程。

图 4-5 “辉”字的四角编码示意图

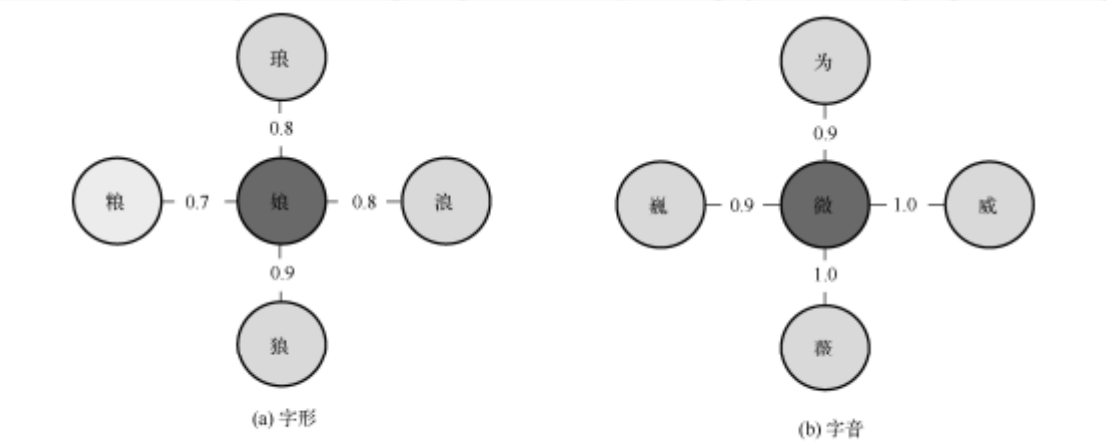
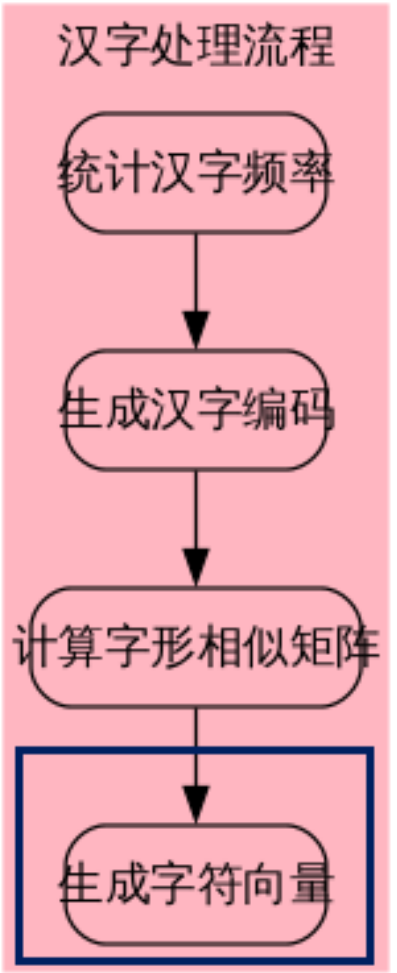


图 4-6 具有相似字形和字音的字符示例

汉字字符向量提取

接着我们需要将汉字的初始特征向量与相似性信息结合起来，生成更准确的**字符向量**，这些向量将用于后续生成句子嵌入向量。

这些向量不仅包含每个汉字的初始特征向量，还考虑了汉字之间的**相似性**，从而能够更好地反映汉字的语义信息。



设计思路

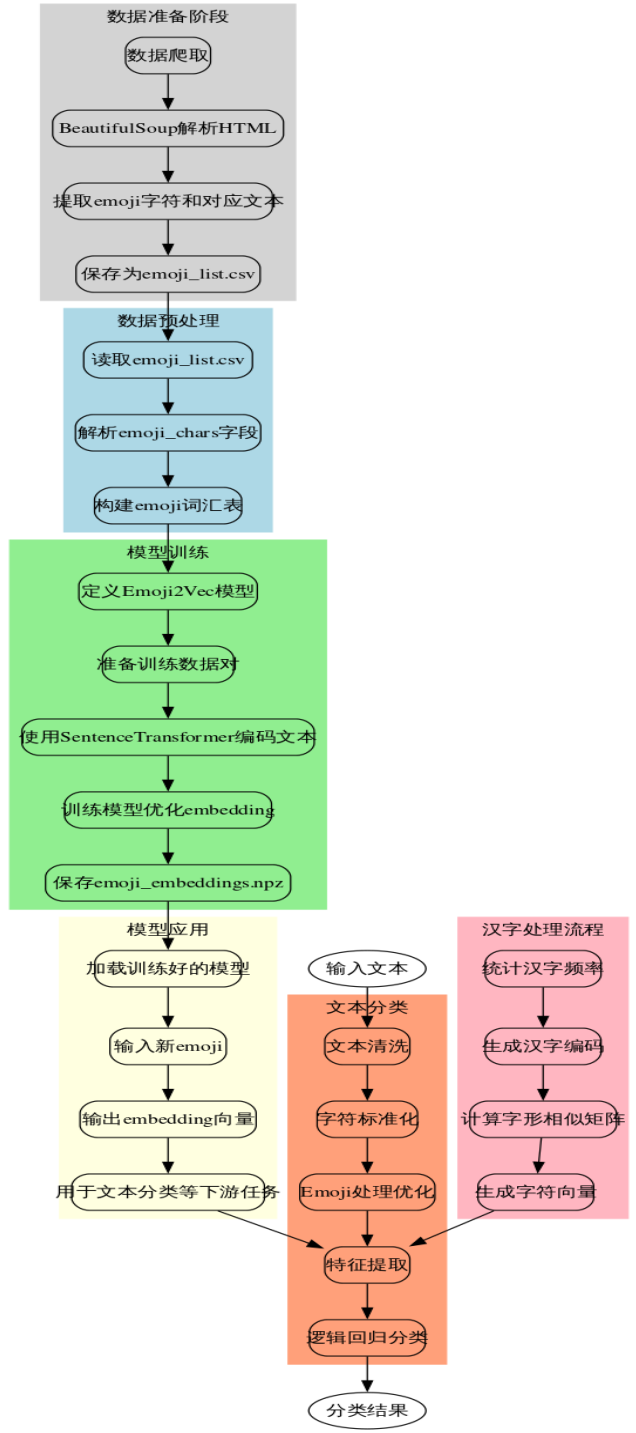
整个系统可以分为三个模块：

主流程 + emoji语义向量提取 + 汉字字符向量提取。

综上，将到的字符向量通过注意力机制生成**句子嵌入向量**，这些句子嵌入向量能够捕捉句子的整体语义信息，用于后续的模型训练和预测任务。

其中**注意力机制**是部分的关键，它能够让模型在处理输入数据时，可以分配不同的权重给输入数据的不同部分。

二分类问题可以使用逻辑回归、SVM、MLP等模型。



目录 CONTENTS

- 设计思路
- 结果展示
- 优化说明

结果展示

做法	原版本	版本1	版本2
字向量的表达	word2vec	word2vec 基于字符共现频率，捕获字形/上下文关联	sentence transformer 基于文本描述语义
emoji向量的表达	无	使用sentence transformer进行训练，得到中间层输出作为向量	使用sentence transformer进行训练，得到中间层输出作为向量

使用不同的编码模型，比较其效果。

Emoji2Vec 的中间层输出是经过汉字释义（如“笑脸”）监督微调
由于版本1的emoji向量和字向量语义空间不统一，向量空间不对齐
若两者量级差异大（如 emoji 向量范数远大于汉字向量），会导致：

1. 相似度计算（如余弦相似度）被 emoji 向量主导。
2. 模型注意力机制出现偏差。

因此我们在合并两个表之前要进行**归一化**：将所有向量归一化为单位长度

结果展示

做法	原版本	版本1	版本2
字向量的表达	word2vec	word2vec 基于字符共现频率，捕获字形/上下文关联	sentence transformer 基于文本描述语义
emoji向量的表达	无	使用sentence transformer进行训练，得到中间层输出作为向量	使用sentence transformer进行训练，得到中间层输出作为向量

分类报告：

	precision	recall	f1-score	support
0	0.93	0.87	0.90	2498
1	0.94	0.97	0.96	5506
accuracy			0.94	8004
macro avg	0.94	0.92	0.93	8004
weighted avg	0.94	0.94	0.94	8004

	precision	recall	f1-score	support
0	0.92	0.92	0.92	2580
1	0.96	0.96	0.96	5561
accuracy			0.95	8141
macro avg	0.94	0.94	0.94	8141
weighted avg	0.95	0.95	0.95	8141

分类报告：

	precision	recall	f1-score	support
0	0.85	0.88	0.86	2580
1	0.94	0.93	0.94	5561
accuracy			0.91	8141
macro avg	0.90	0.90	0.90	8141
weighted avg	0.91	0.91	0.91	8141

整体性能对比→类别性能分析→字向量生成方法影响→分类器选择影响

结果展示

做法	原版本	版本1	版本2
字向量的表达	word2vec	word2vec 基于字符共现频率，捕获字形/上下文关联	sentence transformer 基于文本描述语义
emoji向量的表达	无	使用sentence transformer进行训练，得到中间层输出作为向量	使用sentence transformer进行训练，得到中间层输出作为向量

第一版 (SVM + W2V)	0.92	0.92	0.92
第二版 (SVM + Char)	0.85	0.88	0.86
原始版 (LR + W2V)	0.93	0.87	0.90

对于不同的类别分析如下：

类别0为**少数类，非垃圾文本。**

第一版最佳：所有指标均最高（F1=0.92），表明Word2Vec向量能更好地区分非垃圾文本。

第二版最弱：Precision最低（0.85），说明字符级向量导致大量“误报”（将非垃圾预测为垃圾）。Recall略高（0.88）可能是因为字符级特征捕捉了部分通用模式，但噪声较大。

原始版问题：Recall最低（0.87），表明LR模型漏检较多非垃圾文本（假阴性高）

整体性能对比→类别性能分析→字向量生成方法影响→分类器选择影响

结果展示

做法	原版本	版本1	版本2
字向量的表达	word2vec	word2vec 基于字符共现频率，捕获字形/上下文关联	sentence transformer 基于文本描述语义
emoji向量的表达	无	使用sentence transformer进行训练，得到中间层输出作为向量	使用sentence transformer进行训练，得到中间层输出作为向量

对于类别1即**多数类，垃圾文本**而言：

第一版最优：所有指标最高（F1=0.96），Word2Vec向量精准捕捉垃圾文本特征。

第二版稍弱：所有指标略低于第一版，字符级向量对垃圾文本的区分能力不足。

原始版特点：Recall极高（0.97），但Precision较低（0.94），表明LR倾向于将更多样本预测为垃圾（假阳性高），可能因类别不平衡导致。

性能对比：

版本	Precision	Recall	F1-score
第一版 (SVM + W2V)	0.96	0.96	0.96
第二版 (SVM + Char)	0.94	0.93	0.94
原始版 (LR + W2V)	0.94	0.97	0.96

整体性能对比→类别性能分析→字向量生成方法影响→分类器选择影响

结果展示

做法	原版本	版本1	版本2
字向量的表达	word2vec	word2vec 基于字符共现频率，捕获字形/上下文关联	sentence transformer 基于文本描述语义
emoji向量的表达	无	使用sentence transformer进行训练，得到中间层输出作为向量	使用sentence transformer进行训练，得到中间层输出作为向量

分类报告：

	precision	recall	f1-score	support
0	0.93	0.87	0.90	2498
1	0.94	0.97	0.96	5506
accuracy			0.94	8004
macro avg	0.94	0.92	0.93	8004
weighted avg	0.94	0.94	0.94	8004

	precision	recall	f1-score	support
0	0.92	0.92	0.92	2580
1	0.96	0.96	0.96	5561
accuracy			0.95	8141
macro avg	0.94	0.94	0.94	8141
weighted avg	0.95	0.95	0.95	8141

分类报告：

	precision	recall	f1-score	support
0	0.85	0.88	0.86	2580
1	0.94	0.93	0.94	5561
accuracy			0.91	8141
macro avg	0.90	0.90	0.90	8141
weighted avg	0.91	0.91	0.91	8141

整体性能对比→类别性能分析→字向量生成方法影响→分类器选择影响

目录 CONTENTS

- 设计思路
- 结果展示
- 优化说明

优化说明

1.垃圾文本有很多明显且有力的特征可以用于鉴别，比如：

假设有一个敏感词词库，如果文本中含有**敏感词**，如果文本中有**长串的数字/字母串**（电话号码/网址/微信），则该文本大概率是垃圾文本；某些 **Unicode** 字符（如 零宽字符 U+200B）不占视觉宽度，可能被滥用于逃避检测。

如："正□常□文□本"（□ = U+200B，视觉不可见）

如果这个文本大量使用零宽字符，这个文本大概率是垃圾文本

在本实验中，我们实现长串数字/字母串的识别，并作为一个特征接入分类器。

特征工程通过人为设计特征提高分类准确率。

我们基于大模型的emoji编码，利用emoji的语义信息和预训练大语言模型 paraphrase-multilingual-MiniLM-L12-v2 对emoji进行编码有效识别使用emoji表达垃圾信息的文本，并比较不同编码方法的效果。

优化说明

2.原项目着重于字符相似性网络的使用，因此是对每一个汉字进行word2vec编码，缺乏了对语义的捕捉。（字符级向量忽略词序和组合语义（如“彩票” vs “票彩”），而垃圾文本常依赖特定词组合）因此可以引入**多头注意力机制**，具体思路如下

- 对文本进行**分词操作**
- 其中一个注意力头计算词和词之间中的**汉字编码的相似度**。之所以直接点积是因为这样可以强化“语义上重要字符”对句子表示的贡献 —— 因为它们在相似度加权中会显得更突出
- 引入**可训练的QKV**矩阵，对词进行QKV得到q,k,v后再计算对齐分数，这样可以捕捉深层的词的语义关联
- 综合多个**注意力头**得到句子表征

3.知识增强，利用敏感词词库进行检索； 利用**贝叶斯概率公式/先验概率**提高准确性

感谢聆听

SUN YAT-SEN UNIVERSITY

组员：邓凯纳、邓博高、庞子良