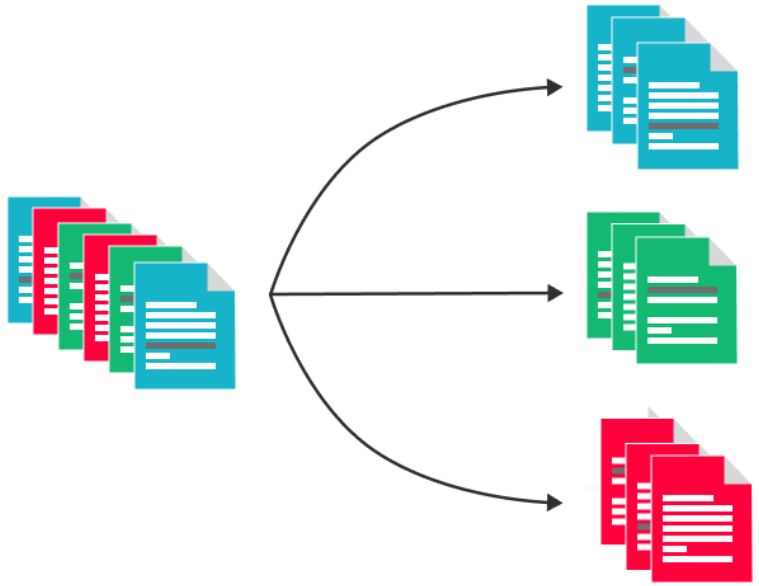
实验3

文本分类

前言

文本分类，也称为文本分组或文本标记，是将文本文档分配给一个或多个类别的过程。



- 运行环境

```
python3.6+
pytorch 1.4+
transformers
AllenNLP
sklearn
fire
```

注意！！Numpy版本可能存在不兼容，以下版本确定兼容，若出现问题可调整到如下版本

```
 numpy==1.19.5
 pip install spacy==3.0.6
 thinc==8.0.3
 scipy==1.5.4
```

- # 实验数据

电影评论情感分类数据集包括 IMDb 数据集、Rotten Tomatoes 数据集、Amazon 评论数据集等。

识别文本的情感极性，如正面、负面。本次实验采用2分类的情感分类数据，总计约10k条数据。评价指标使用准确率(Accuracy)。

- # 文本分类主要包含三部分：

word embedding 比如使用经典的word2vec, glove或者近几年很火的ELMo和Bert等将word转换为词向量。

Encoder 对word Embedding的输入进行下一步的特征提取，比如常用的CNN,RNN还有近年来很火的Transformer架构，都是能够抽取词的更high-level的语义特征。

池化，将多个词特征进行池化得到一个句向量，进行最后的分类。比较常用的比如max-pooling，average-pooling，或者利用Attention机制计算每个词的不同权重，进行加权求和，得到句向量。

本次实验集中在第一部分，测试在不同的词向量的文本分类的效果。

- 本次实验中已经实现了ELMo词向量

  ELMo：使用AllenNLP提供的预训练的Medium的上下文词向量，LSTM的隐层与输出层维度为2048/256。

- 实现了四种Encoder的方法

  Mean：不Encoder，直接对所有的词向量取平均作为句向量。
  CNN：使用卷积神经网络抽取特征。
  RNN：使用GRU网络抽取特征。
  Transformer：结合Position Embedding+Transformer Encoder来抽取特征，使用Pytorch提供的nn.Transformer 实现，其中num_head=1，num_layer=1。

- 池化

  由于本实验不考虑池化，因此统一使用mean 将所有词的特征取平均作为句向量。
  其余参数均保持一致，可以在config.py 里面查看，比如所有输出层都为64。需要注意的是，本次实验并没有刻意去调超参数，只有所有对比模型保持一致的原则。

  使用--gpu_id=1来指定使用的GPU

- **资源下载**

下载代码，成功复现，可以解释代码（6-7分）

- 实验实现

运行main.py
修改参数

```
if __name__ == "__main__":
 kwargs = {              'emb_method': 'elmo',
                  'enc_method': 'cnn',
                 # 其他参数根据需要添加    }
```

- **完善代码**

  补充其他词向量模型（每个模型1分）

  GloVe、Bert

- **完善代码**

  考虑不同池化层的影响，加入不同的pooling method（每个池化方法1分）

  - Max Pooling:在每个时间步中，选择特征值最大的元素。
  - Average Pooling:在每个时间步中，计算所有特征值的平均值。
  - Global Max Pooling:在整个特征图上选择最大值。
  - Global Average Pooling:在整个特征图上计算平均值。
  - K-max Pooling:在每个时间步中，选择K个最大值。
  - Attention Pooling:根据注意力机制加权平均各时间步的特征。
  - Dynamic Max Pooling:在每个时间步中，动态选择最大值，位置随输入变化。

- 完善代码

  对比分析同一词向量方法不同Encoder方法或者池化方法的模型结果（分析1分）

Embedding	Encoder	Acc	Second
ELMo	Mean	0.7572	25.05s
ELMo	CNN	0.8172	25.53s
ELMo	RNN	0.8219	27.18s
ELMo	Transformer	0.8051	26.209
Bert	MEAN		
Bert	CNN		
Bert	RNN		
Bert	Transformer		

谢谢

FOR YOUR LISTENING

陈雅妮.