https://github.com/NLP-lecture/part-2

教程二 关键词提取

东北大学自然语言处理实验室

关键词提取概述

- 关键词是快速获取文档主题的重要方式，在信息检索和自然语言处理等领域均有重要应用。关键词提取，顾名思义，是从文档内容中寻找并推荐关键词。这个任务是文本挖掘领域的一个分支，是文本检索、文档比较、摘要生成、文档分类和聚类等文本挖掘研究的基础性工作。
  - 传统的方法主要依靠词汇统计信息进行关键字提取，不需要人工标注的语料，该类方法是先抽取出候选词，然后对各个候选词进行打分，然后输出top-K个分值最高的候选词作为关键词。

二氧化锰是两性氧化物，它是一种常温下非常稳定的黑色粉末状固体，可作为干电池的去极化剂。

二氧化锰是一种无机化合物，化学式为MnO_2，为黑色无定形粉末或黑色斜方晶体，难溶于水、弱酸、弱碱、硝酸、冷硫酸，加热情况下溶于浓盐酸而产生氯气。

→

关键词	分数
两性氧化物	20
二氧化锰	100
黑色	90
…	…

一个关键词提取的样例

[1] 本文多处参考清华大学刘知远老师论文(http://nlp.csai.tsinghua.edu.cn/~lzy/publications/phd_thesis.pdf) 和技术博客(https://blog.csdn.net/asialee_bird/article/details/96454544)

关键词提取算法

- 关键字提取算法有很多，在这里主要介绍一种是基于统计特征的关键词提取算法（TF-IDF），这个算法的思想是利用文档中词语的统计信息抽取文档的关键词。

- TF-IDF算法中TF表示**词频**（Term Frequency）和IDF表示**逆文档频率**（Inverse Document Frequency）是一种用于信息检索与文本挖掘的常用加权技术。
  - 现在，我们有很多的文档，词频越大表示某个单词在文档中越重要；同时，如果这个单词在越多的文档中都出现过，它也越不重要。换句话说，单词的重要性与它在文档中出现的次数成正比，与它在语料库中出现的频率成反比

“二氧化锰“基本只会出现在化学文档中；”企鹅”基本只会出现在生物文档中。而“是”很多文档中都会出现。所以“二氧化锰”和“企鹅”比“是”更重要，更应该是关键词。

化学文档：二氧化锰是两性氧化物…可作为干电池的去极化剂。…二氧化锰是一种无机化合物，化学式为MnO_2 …氯气。二氧化锰是…

生物文档：企鹅是鸟纲、企鹅科所有物种的通称…是一种最古老的游禽。…体型最大的物种是帝企鹅，平均约1.1米高，…最小的企鹅物种是…

TF-IDF算法

- 令w表示单词， d表示文档

TF：w在d中出现的次数
即 w在d中的重要性

$$\mathrm{tf}_{w,d} = c(w,d)$$

或

$$\mathrm{tf}_{w,d} = \log(c(w,d)+1)^1$$

IDF：w出现的文档数的倒数
即 w出现的频繁度（或不频繁度）

$$\mathrm{idf}_w = \log(\frac{N}{\mathrm{df}_w}+1)^1$$

其中，N表示所有文档数， df_w表示包含w的文档数

- 最后，将两项相乘得到最后的结果为：

$$\mathrm{tf}-\mathrm{idf}_{w,d} = \mathrm{tf}_{w,d} \times \mathrm{idf}_w$$

[1] $\mathrm{tf}_{w,d} = \log(c(w,d)+1)$ 中加入log()函数可以弱化$c(w,d)$线性增长带来的影响，+1避免结果为负。

教程三 机器翻译

东北大学自然语言处理实验室

安装工具

- 创建python3虚拟环境

  Python3 –m venv mt

- 安装pytorch

  Pip install torch

- 安装fairseq

https://github.com/pytorch/fairseq

- PyTorch version >= 1.5.0
- Python version >= 3.6
- For training new models, you'll also need an NVIDIA GPU and NCCL
- **To install fairseq** and develop locally:

```
git clone https://github.com/pytorch/fairseq
cd fairseq
pip install --editable ./

# on MacOS:
# CFLAGS="-stdlib=libc++" pip install --editable ./

# to install the latest stable release (0.10.x)
# pip install fairseq
```

获得数据

- 下载数据

https://github.com/NLP-lecture/part-3

- 安装分词工具

https://github.com/rsennrich/subword-nmt

INSTALLATION

install via pip (from PyPI):

```
pip install subword-nmt
```

- bpe.sh

```
cat train_origin.en train_origin.fr > train_origin.enfr

subword-nmt learn-bpe -s 10000 < train_origin.enfr > code
subword-nmt apply-bpe -c code < train_origin.en > train.en
subword-nmt apply-bpe -c code < train_origin.fr > train.fr
subword-nmt apply-bpe -c code < valid_origin.en > valid.en
subword-nmt apply-bpe -c code < valid_origin.fr > valid.fr
subword-nmt apply-bpe -c code < test_origin.en > test.en
subword-nmt apply-bpe -c code < test_origin.fr > test.fr
```

开始训练

- 加载数据

```
src=en
tgt=fr
TEXT=../translation_data/multi30k
fairseq-preprocess --source-lang $src --target-lang $tgt \
  --trainpref $TEXT/train \
  --validpref $TEXT/valid \
  --testpref $TEXT/test \
  --destdir data-bin/multi30k \
  --workers 8 \
  --joined-dictionary
  #--thresholdtgt 4 \
  #--thresholdsrc 4 \
  #--nwordstgt 20000 \
  #--nwordssrc 20000
```

- 开始训练

```
DATA='data-bin/multi30k'  # input data
ARCH='transformer_tiny'  # model structure
SAVE='checkpoints/transformer.en-fr.tiny'  # save dir

tgt='fr'
src='en'

CUDA_VISIBLE_DEVICES=0,1,2,3,4,5,6,7
fairseq-train $DATA --task translation \
      --arch $ARCH --share-all-embeddings --dropout 0.35 \
      --warmup-updates 150 --lr 0.006 \
      --max-tokens 8196 \
      --max-update 1000 \
      --source-lang $src \
      --target-lang $tgt \
      --save-dir $SAVE \
      --keep-last-epochs 6 \
      --find-unused-parameters --patience 5 \
      --optimizer adam
```

模型评价

- 模型评价

```
set -e
model_dir=checkpoints/transformer.en-fr.tiny
# set device
gpu=7
# data set
who=test
ensemble=5
batch_size=128
beam=10
src_lang=en
tgt_lang=fr
length_penalty=1.3
data_dir=multi30k
checkpoint=checkpoint_best.pt

if [ -n "$ensemble" ]; then
        if [ ! -e "$model_dir/last$ensemble.ensemble.pt" ]; then
                PYTHONPATH=`pwd` python3 scripts/average_checkpoints.py --inputs $model_dir --output $model_dir/last$ensemble.ensemble.pt --num-epoch
checkpoints $ensemble
        fi
        checkpoint=last$ensemble.ensemble.pt
fi

output=$model_dir/translation.log
export CUDA_VISIBLE_DEVICES=$gpu
fairseq-generate data-bin/$data_dir \
  -s $src_lang -t $tgt_lang \
  --path $model_dir/$checkpoint \
  --gen-subset $who \
  --batch-size $batch_size \
  --beam $beam \
  --lenpen $length_penalty \
  --remove-bpe > translate.txt
```

谢谢！