HarvestText

Release 0.8

Oct 08, 2020

Contents:

本文档目前记录了部分函数的参数含义，具体例子请见项目主页：https://github.com/blmoistawinde/HarvestText

CHAPTER 1

harvesttext package

1.1 Submodules

1.2 harvesttext.entity_discoverer module

1.3 harvesttext.harvesttext module

class harvesttext.harvesttext.**HarvestText**(*standard_name=False*, *language='zh_CN'*)

Bases: harvesttext.ent_network.EntNetworkMixin, harvesttext.ent_retrieve.EntRetrieveMixin, harvesttext.parsing.ParsingMixin, harvesttext.sentiment.SentimentMixin, harvesttext.summary.SummaryMixin, harvesttext.word_discover.WordDiscoverMixin

主模块：- 主要保留了与实体分词、分句，预处理相关的代码 - 还有存取、状态管理等基础代码 - 其他功能在各个 mixin 里面 - 主模块的功能是会被各个子模块最频繁调用的，也体现了本库以实体为核心，基于实体展开分析或改进算法的理念

add_entities(*entity_mention_dict=None*, *entity_type_dict=None*, *override=False*, *load_path=None*)

登录的实体信息到 ht，或者从 save_entities 保存的文件中读取（如果指定了 load_path）

Parameters

- **entity_mention_dict** – dict, {entity:[mentions]} 格式，
- **entity_type_dict** – dict, {entity:entity_type} 格式，

- **override** – bool, 是否覆盖已登录实体，默认 False
- **load_path** – str, 要读取的文件路径（默认不使用）

Returns None

add_new_entity(*entity0*, *mention0=None*, *type0='添加词'*)

add_new_mentions(*entity_mention_dict*)

add_new_words(*new_words*)

add_typed_words(*type_word_dict*)

build_trie(*new_word*, *entity*, *entity_type*)

check_prepared()

choose_from(*surface0*, *entity_types*)

choose_from_multi_mentions(*mention_cands*, *sent=''*)

clean_text(*text*, *remove_url=True*, *email=True*, *weibo_at=True*, *stop_terms=('转发微博',)*, *emoji=True*, *weibo_topic=False*, *deduplicate_space=True*, *norm_url=False*, *norm_html=False*, *to_url=False*, *remove_puncts=False*, *remove_tags=True*, *t2s=False*)

进行各种文本清洗操作，微博中的特殊格式，网址，email，html 代码，等等

Parameters

- **text** – 输入文本
- **remove_url** – （默认使用）是否去除网址
- **email** – （默认使用）是否去除 email
- **weibo_at** – （默认使用）是否去除微博的 @ 相关文本
- **stop_terms** – 去除文本中的一些特定词语，默认参数为（“转发微博”,）
- **emoji** – （默认使用）去除 [] 包围的文本，一般是表情符号
- **weibo_topic** – （默认不使用）去除 ## 包围的文本，一般是微博话题
- **deduplicate_space** – （默认使用）合并文本中间的多个空格为一个
- **norm_url** – （默认不使用）还原 URL 中的特殊字符为普通格式，如 (%20 转为空格)
- **norm_html** – （默认不使用）还原 HTML 中的特殊字符为普通格式，如 (转为空格)
- **to_url** – （默认不使用）将普通格式的字符转为还原 URL 中的特殊字符，用于请求，如 (空格转为%20)
- **remove_puncts** – （默认不使用）移除所有标点符号

- **remove_tags** – （默认使用）移除所有 html 块
- **t2s** – （默认不使用）繁体字转中文

Returns 清洗后的文本

clear()

cut_sentences(*para*, *drop_empty_line=True*, *strip=True*, *deduplicate=False*)

Parameters

- **para** – 输入文本
- **drop_empty_line** – 是否丢弃空行
- **strip** – 是否对每一句话做一次 strip
- **deduplicate** – 是否对连续标点去重，帮助对连续标点结尾的句子分句

Returns sentences: list of str

decoref(*sent*, *entities_info*)

deprepare()

dig_trie(*sent*, *l*)

entity_linking(*sent*, *pinyin_tolerance=None*, *char_tolerance=None*, *keep_all=False*, *with_ch_pos=False*)

Parameters

- **sent** – 句子/文本
- **pinyin_tolerance** – {None, 0, 1} 搜索拼音相同 (取 0 时) 或者差别只有一个 (取 1 时) 的候选词链接到现有实体，默认不使用 (None)
- **char_tolerance** – {None, 1} 搜索字符只差 1 个的候选词 (取 1 时) 链接到现有实体，默认不使用 (None)
- **keep_all** – if True, keep all the possibilities of linked entities
- **with_ch_pos** – if True, also returns ch_pos

Returns entities_info：依存弧, 列表中的列表。if not keep_all: [([l, r], (entity, type)) for each linked mention m] else: [([l, r], set((entity, type) for each possible entity of m)) for each linked mention m] ch_pos: 每个字符对应词语的词性标注（不考虑登录的实体，可用来过滤实体，比如去掉都由名词组成的实体，有可能是错误链接）

get_linking_mention_candidates(*sent*, *pinyin_tolerance=None*, *char_tolerance=None*)

get_pinyin_correct_candidates(*word*, *tolerance=1*)

hanlp_prepare()

load_entities(*load_path='./ht_entities.txt'*, *override=True*)

从 save_entities 保存的文件读取实体信息

Parameters

- **load_path** – str, 读取路径（默认：./ht_entities.txt）
- **override** – bool, 是否重写已登录实体，默认 True

Returns None, 实体已登录到 ht 中

mention2entity(*mention*)

找到单个指称对应的实体

Parameters **mention** – 指称

Returns 如果存在对应实体，则返回（实体, 类型），否则返回 None, None

posseg(*sent*, *standard_name=False*, *stopwords=None*)

prepare()

remove_entity(*entity*)

remove_mention(*mention*)

save_entity_info(*save_path='./ht_entities.txt'*, *entity_mention_dict=None*, *entity_type_dict=None*)

保存 ht 已经登录的实体信息，或者外部提供的相同格式的信息，目前保存的信息包括 entity,mention,type.

如果不提供两个 dict 参数，则默认使用模型自身已登录信息，否则使用提供的对应 dict

格式:

entity|| 类别 mention|| 类别 mention|| 类别

entity|| 类别 mention|| 类别

每行第一个是实体名，其后都是对应的 mention 名，用一个空格分隔，每个名称后面都对应了其类别。

保存这个信息的目的是为了便于手动编辑和导入:

- 比如将某个 mention 作为独立的新 entity，只需剪切到某一行的开头，并再复制一份再后面作为 mention

Parameters

- **save_path** – str, 要保存的文件路径（默认: ./ht_entities.txt）
- **entity_mention_dict** – dict, {entity:[mentions]} 格式，
- **entity_type_dict** – dict, {entity:entity_type} 格式，

Returns None

search_word_trie(*word*, *tolerance=1*)

Parameters

- **word** –
- **tolerance** –

Returns

seg(*sent*, *standard_name=False*, *stopwords=None*, *return_sent=False*)

set_linking_strategy(*strategy*, *lastest_mention=None*, *entity_freq=None*, *type_freq=None*)

为实体链接设定一些简单策略，目前可选的有：’None’,’freq’,’latest’,’latest&freq’

‘None’：默认选择候选实体字典序第一个

‘freq’：对于单个字面值，选择其候选实体中之前出现最频繁的一个。对于多个重叠字面值，选择其中候选实体出现最频繁的一个进行连接【每个字面值已经确定唯一映射】。

‘latest’：对于单个字面值，如果在最近有可以确定的映射，就使用最近的映射。

‘latest’- 对于职称等作为代称的情况可能会比较有用。

比如”经理”可能代指很多人，但是第一次提到的时候应该会包括姓氏。我们就可以记忆这次信息，在后面用来消歧。

‘freq’- 单字面值例：’市长’+{‘A 市长’:5,’B 市长’:3} -> ‘A 市长’

重叠字面值例，’xx 市长江 yy’+{‘xx 市长’:5,’长江 yy’:3}+{‘市长’:’xx 市长’}+{‘长江’:’长江 yy’} -> ‘xx 市长’

Parameters

- **strategy** – 可选‘None’,’freq’,’latest’,’latest&freq’中的一个
- **lastest_mention** – dict, 用于’latest’, 预设
- **entity_freq** – dict, 用于’freq’, 预设某实体的优先级（词频）
- **type_freq** – dict, 用于’freq’, 预设类别所有实体的优先级（词频）

:return None

1.4 harvesttext.match_patterns module

1.5 harvesttext.resources module

harvesttext.resources.get_baidu_stopwords()

获得百度停用词列表来源，网上流传的版本：https://wenku.baidu.com/view/98c46383e53a580216fcfed9.html 包含了中英文常见词及部分标点符号

Returns stopwords: set of string

harvesttext.resources.**get_english_senti_lexicon**(*type='LH'*)

获得英语情感词汇表

目前默认为来自这里的词汇表 https://www.cs.uic.edu/~liub/FBS/sentiment-analysis.html#lexicon

If you use this list, please cite the following paper:

Minqing Hu and Bing Liu. "Mining and Summarizing Customer Reviews." Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD-2004), Aug 22-25, 2004, Seattle, Washington, USA,

Returns sent_dict = { "pos" :[words]," neg" :[words]}

harvesttext.resources.**get_jieba_dict**(*min_freq=0*, *max_freq=inf*, *with_pos=False*, *use_proxy=False*, *proxies=None*)

获得 jieba 自带的中文词语词频词典

Params min_freq 选取词语需要的最小词频

Params max_freq 选取词语允许的最大词频

Params with_pos 返回结果是否包括词性信息

Return if not with_pos, dict of {wd freq}, else, dict of {(wd, pos): freq}

harvesttext.resources.**get_nltk_en_stopwords**()

来自 nltk 的英语停用词

Returns stopwords: set of string

harvesttext.resources.**get_qh_sent_dict**()

获得参考褒贬义词典：褒贬义词典清华大学李军

此资源被用于以下论文中: Jun Li and Maosong Sun, Experimental Study on Sentiment Classification of Chinese Review using Machine Learning Techniques, in Proceding of IEEE NLPKE 2007 李军中文评论的褒贬义分类实验研究硕士论文清华大学 2008

Returns qh_sent_dict = { "pos" :[words]," neg" :[words]}

harvesttext.resources.**get_qh_typed_words**(*used_types=['IT', '动物', '医药', '历史人名', '地名', '成语', '法律', '财经', '食物']*)

THUOCL：清华大学开放中文词库 http://thuocl.thunlp.org/ IT 财经成语地名历史名人诗词医学饮食法律汽车动物

Parameters used_types –

Returns typed_words: 字典，键为类型，值为该类的词语组成的 set

harvesttext.resources.**get_sanguo**()

获得三国演义原文

Returns ["章节 1 文本" ," 章节 2 文本" ,···]

harvesttext.resources.**get_sanguo_entity_dict**()

获得三国演义中的人名、地名、势力名的知识库。自行搭建的简单版，一定有遗漏和错误，仅供参考使用

Returns entity_mention_dict,entity_type_dict

1.6 harvesttext.sent_dict module

1.7 harvesttext.texttile module

1.8 harvesttext.utils module

1.9 harvesttext.word_discoverer module

1.10 Module contents

harvesttext.**loadHT**(*filename*)

harvesttext.**saveHT**(*htModel*, *filename*)

CHAPTER 2

Indices and tables

- genindex
- modindex
- search

Python Module Index

h

Index

H

L

M

P

R

S