11/03/2021

APACHE OPEN NLP

1. OVERVIEW

Apache OpenNLP is an open source java-based machine learning library for performing common NLP related tasks. The library also includes maximum entropy and perceptron-based machine learning. The library provides set of APIs' to perform common NLP tasks such as the following:

- Language Detection
- Sentence Detection
- Tokenization
- Name Finder
- POS tagging
- Lemmatization
- Chunking

For full features, please refer to
https://opennlp.apache.org/docs/1.9.3/manual/opennlp.html

2. FEATURES

Apache Open NLP provides a component-based architecture which can be used in building an NLP pipeline. Each component performs an NLP task. Some of the components include Sentence Detector, Tokenizer, Name Finder, Document Categorizer, Chunker, Parser and Coreference resolution. These components can also be combined to form an NLP pipeline.

The component in general provides an API to perform the following:

- Perform the NLP task
- Train the model for an NLP task
- Evaluation

The toolset also comes with predefined models for some of the tasks and languages. These models can be used as it is or new models can be built by providing the appropriate training parameters and the examples.

The tool set provides two modes to perform an NLP task, an API mode and a command line CLI mode.

- API mode:

  The API mode is simple and straightforward to use. Each API in general expects a model and an input to perform an NLP task.

  Here is an example of how to perform an "Tokenization task"

```java
try (InputStream modelIn = new FileInputStream("en-token.bin")) {
  TokenizerModel model = new TokenizerModel(modelIn);
}

Tokenizer tokenizer = new TokenizerME(model);

String tokens[] = tokenizer.tokenize("An input sample sentence.");

// output : "An", "input", "sample", "sentence", "."
```

Credit: https://opennlp.apache.org/docs/1.9.3/manual/opennlp.html#tools.tokenizer.api

Here are some details on the components and its associated API :

Component	Description	Java API
Language Detector	Component to predict language for a given document. Supports 103 languages. Provides the language output in ISO-639-3 format (eg: eng, tam)	**LanguageDetector**
Sentence Detector	Component to detect and extract sentences from a given document	**SetenceDetector**
Tokenizer	Component to extract character sequences into tokens such as words, numbers, etc..	**Tokenizer**
Name Finder	Component to extract named entities (NER) such as a person, place, number etc… This. component uses Maximum Entropy model	**NameFinderME**
Document Categorizer	Component for document classification to classify the document into predefined categories. This. component uses Maximum Entropy model	**DocumentCategorizerME**
POS Tagger	Component to tag the tokens in a given corpus with particular Part of Speech (POS). This. component uses Maximum Entropy model	**POSTaggerME**
Lemmatizer	Component to perform lemmatization/normalization of a token. This. component uses Maximum Entropy model	**LemmatizerME**
Chunker	Component to chunk sentences to word groups. It works on top of POStagging. This. component uses Maximum Entropy model	**ChunkerME**
Parser	Component to provide the parse tree(internal structure based on the grammar) for a given raw sentence.	**Parser**

- Command CLI mode:

The command line script (opennlp.bat or opennlp) is another alternative to perform the NLP tasks, training, evaluation. The CLI mode is normally useful when manual activities are needed (Eg: researching, exploring)

Here is an example of invoking the Tokenizer tool using CLI

For full documentation, please refer the following link:
https://opennlp.apache.org/docs/1.9.3/manual/opennlp.html#intro.cli

OPENNLP VS NLTK (NATURAL LANGUAGE TOOLKIT)

NLTK is another machine learning based NLP toolkit. I have listed down quick comparison of features between OpenNLP and NLTK.

Feature	openNLP	NLTK
Open Source	Yes	Yes
Language	Java	Python
POS Tagging	Yes	Yes
Chunker	Yes	Yes
Name Entity Finder	Yes	Yes
Tokenization	Yes	Yes
Document Classification	Yes	Yes
Sentence Detection	Yes	Yes
Lemmatization	Yes	Yes
Stemming	Yes	Yes
API availability	Yes	Yes
Command Line Interface	Yes	Yes
Predefined Models	Yes	Yes
Support for Entropy	Yes	Yes
Support for Perceptron	Yes	Yes
Concordance	No	Yes

CONCLUSION:

Apache OpenNLP is a solid option for performing Enterprise level NLP needs. I have listed some of the pros and cons of the library

- Pros :
  - Provides most of the basic NLP tasks.
  - Open Source hence easy to extend or modify according to one's need.
  - Standard API interface for easier integration
  - Pre-trained models provide a good starting point.
- Cons :
  - No super active development. Last stable release was around 2019.
  - Only java based.