Open Source Text Mining Tools and Libraries

Companion to the PASED 2011 tutorial on
"Information Retrieval Methods for Software Engineering"

created by Sonia Haiduc

Lucene

http://lucene.apache.org

- High-performance, full-featured text search engine library
- Written entirely in Java
- Based on the Vector Space Model and the Boolean Model in IR
- Comes with a set of basic applications, which can be used as-is or modified by users
- Contains functionality for:
  - **Document processing**: stop-words removal, stemming, tokenization, etc.
  - **Indexing**
  - **Searching**

Searching in Lucene

http://lucene.apache.org

- Supports the indexing and searching of several *fields* for each document (e.g., “title”, “contents”, etc.)
- Accepts several types of queries:
  - **Term query** (e.g., buffer edit)
  - **Phrase query** (e.g., “buffer edit”)
  - **Boolean query** (e.g., buffer AND edit OR modify)
  - **Wildcard query** (e.g., te?t, test*, te*t)
  - **Range query** (e.g., date: [20020101 TO 20030101)
  - **Fuzzy query** - uses the Levenshtein Distance between strings (e.g., roam~ searches for terms similar to roam, like “roam”, “foam”)
  - **Proximity query** – finds terms within a specific distance away (e.g., “jakarta apache”~10 searches for a “apache” and “jakarta” within 10 terms of each other in a document)

Other Lucene Implementations

http://lucene.apache.org

- There are also implementations of Lucene in many other programming languages:
  - **CLucene** - implementation in C++
  - **Lucene.Net** - implementation in .NET
  - **Lucene4c** - implementation in C
  - **Zend Search** - implementation in the Zend Framework for PHP 5
  - **Plucene** and **KinoSearch** - implementations in Perl
  - **PyLucene** - GCJ-compiled version of Java Lucene integrated with Python
  - **MUTIS** - implementation in Delphi
  - **Ferret** - implementation in Ruby
  - **Montezuma** - implementation in Common Lisp

jLSI

http://tcc.itc.it/research/textec/tools-resources/jlsi.html

- An open source Java tool for Latent Semantic Indexing
- Requires the following linguistic processing to be performed *before* its usage:
  - Tokenization
  - Sentence splitting
  - Part-of-speech tagging (optional)
  - Lemmatization (optional)

The Semantic Engine http://knowledgesearch.org/

- A C++ library that implements IR indexing and retrieval
- Uses mathematical algorithms based on graph theory to index the *latent semantic* content of documents
- A semantic graph of a text collection is created which can be used to find relevant documents that may not contain any keyword matches
- **Document processing** functionality: tokenization, POS tagging, stemming, stopwords removal

The Semantic Vectors Package

http://code.google.com/p/semanticvectors

- Support for indexing and retrieval of documents by applying a Random Projection algorithm to term-document matrices created using Apache Lucene
- The Random Projection algorithm is a form of automatic semantic analysis, similar to Latent Semantic Analysis (LSA) and Probabilistic Latent Semantic Analysis (PLSA)
- Very scalable, does not rely on the use of Singular Value Decomposition (SVD), even though it achieves a performance comparable with LSI

The Lemur Toolkit

http://www.lemurproject.org

- Supports the construction of basic text retrieval systems using language modeling, VSM, LSI, probabilistic model
- Interactive applications for Windows, Linux, and Web
- Cross-platform, fast and modular code written in C++
- APIs available for C++, Java, and C#
- Many sample applications, including information retrieval and document clustering applications
- In use for over 6 years by a large user community

Features of Lemur http://www.lemurproject.org

- **Document processing**
  - Tokenization
  - Porter and Krovetz word stemming
  - Stopwords removal
  - Acronym recognition
  - Token-level properties: part of speech, named entities
- **Indexing**
  - Incremental indexing
  - Out-of-the-box indexing support for plain text, HTML, XML, PDF, MBox, Microsoft Word, Microsoft PowerPoint, etc.

Features of Lemur (2) http://www.lemurproject.org

- **Retrieval**
  - Various retrieval models: language modeling approaches, VSM, LSI, tf-idf, Okapi and InQuery
  - Support for relevance feedback
  - Accepts term, phrase, and wildcard queries, as well as queries specified in a structured query language
  - Supports arbitrary document priors (e.g., Page Rank, URL depth)
- **Summarization**
  - Basic applications for the summarization of documents

Features of Lemur (3) http://www.lemurproject.org

- **Document Clustering**
  - Cosine similarity in the VSM as similarity measure for most clustering algorithms
  - Agglomerative and centroid clustering
  - Several clustering algorithms, including K-means and PLSA
- **Evaluation**
  - Applications for evaluating various IR techniques
  - The documents need to be in TREC format

Terrier IR Platform http://terrier.org

- Open source search engine
- Deployable on large-scale collections of documents
- Implemented in Java
- Available as a desktop application, JSP web interface, and API
- Large user-base over 6 years of public release

Features of Terrier

http://terrier.org

- **Document processing**
  - Tokenizer
  - Various stemmers, including the Snowball and Porter stemmers
  - Stopwords remover
  - Acronym expander
- **Indexing**
  - Several indexing strategies
  - Indexing support for text, HTML, PDF, Microsoft Word, Excel, PowerPoint, and TREC collections
  - Indexing of field information (e.g., frequency of terms in the field TITLE)
  - Indexing of position information on a word, or a block
  - Support for fetching files to index by HTTP

Features of Terrier (2)

http://terrier.org

- **Retrieval**
  - Desktop, command-line and Web based querying interfaces
  - Many document weighting models, including 126 Divergence From Randomness (DFR) ranking models, Okapi BM25, language modeling, and TF-IDF
  - Query expansion facilities by pseudo-relevance feedback
  - Advanced query language that supports boolean operators, +/- operators, phrase and proximity search, and search on fields
- **Evaluation**
  - Application for evaluating results of TREC tasks

The Dragon Toolkit

http://dragon.ischool.drexel.edu

- Java development package for Text Mining
- Includes tools for text retrieval, classification, clustering, summarization, and topic modeling
- Integrates a set of NLP tools
- Various document representations including words, phrases, ontology-based concepts and relationships
- Very scalable, especially designed for large-scale application

Features of Dragon

http://dragon.ischool.drexel.edu

- **Document processing**
  - Tokenizers and phrase extractors
  - Part-of-speech tagger
  - Porter Stemmer
  - English lemmatizer
  - Named entity recognizer
  - Various taggers
  - Support for ontology extraction and building
- **Indexing**
  - Supports indexing at the sentence level and sequence level

Features of Dragon (2)

http://dragon.ischool.drexel.edu

- **Retrieval**
  - Supports retrieval based on language modeling methods as well as traditional probabilistic and vector space models
  - Various relevance feedback approaches: Minimum divergence feedback, Rocchio feedback, etc.
- **Classification**
  - Various classifiers: Naïve Bayes, Semantic Naïve Bayes, Nigam active learning, SVM
- **Clustering**
  - Various clustering algorithms: Hierarchical clustering, K-means, and Link-based K-Means

Features of Dragon (3)

http://dragon.ischool.drexel.edu

- **Summarization**
  - Supports generic multi-document summarization
  - Summarizer based on graph-based lexical centrality
- **Topic modeling**
  - Implements three state-of-the-art topic models: the aspect model, the LDA model, and the simple mixture model
- **Evaluation**
  - Provides an evaluation program for each text mining tasks including text retrieval, classification, clustering and summarization

Xapian http://xapian.org/

- An open source search engine library
- Written in C++
- Bindings to allow use from PHP, Perl, Python, C#, Ruby.
- Supports the Probabilistic Information Retrieval model (Okapi BM25) and also a rich set of boolean query operators
- Besides the library, there are also a number of small example programs, and a larger application for indexing and search (Omega)

Features of Xapian http://xapian.org/

- **Document processing**
  - Tokenizer
  - Stemmers
  - Stopwords removal
- **Indexing**
  - Can index plain text, HTML, PHP, PDF, PostScript, OpenOffice, OpenDocument, Microsoft Word/Excel/Powerpoint/Works, etc.
- **Retrieval**
  - Support for relevance feedback and query expansion
  - Types of queries: boolean, term, wildcard, phrase, and proximity
  - Spelling corrector for queries
  - Support for the use of synonyms in queries (“~term”)

Unstructured Information Management Architecture (UIMA)

http://uima.apache.org

- An open, scalable and extensible platform for building analytic solutions that process and search unstructured information to find latent meaning, relationships and relevant facts
- Enables the creation and aggregation of single NLP tools (called Analysis Engines (AEs)) into pipelines (aggregate AEs)
- Developed by IBM, now part of Apache
- Available for Java and C++, but supports also components in Perl, Python, and TCL

UIMA Structure

http://uima.apache.org

- UIMA has three main parts:
  - *Frameworks*, which support configuring and running pipelines of *Annotator* components; frameworks available for Java and C++
  - *Components*, i.e, *Annotators,* which do the actual work of analyzing the unstructured information
  - *Infrastructure*, includes a server that can receive requests and return annotation results, for use by other web services

UIMA Components

http://uima.apache.org

- Current annotators available for UIMA include:
  - Tokenizers
  - Sentence Splitter
  - Stemmers
  - Acronym Annotator
  - Named Entity Tagger
  - Lucene Indexer
  - Concept Mapper
  - Feature Extractor, etc.
- Besides the ones already included in UIMA, annotators can be found at:
  - http://uima.lti.cs.cmu.edu
  - http://www.julielab.de/Resources/Software/NLP+Tools.html

GATE http://gate.ac.uk/

- A comprehensive open source infrastructure for developing language processing applications
- Written in Java
- Mature and actively supported
- Leverages also other projects like:
  - **Information Retrieval:** Lucene, Google and Yahoo search APIs
  - **Machine Learning:** Weka, MaxEnt, SVMLight, etc.
  - **Ontology Support:** Sesame and OWLIM
  - **Parsing:** RASP, Minipar, and SUPPLE
  - **Other:** UIMA, Wordnet, Snowball, etc.

Tasks Covered by GATE

http://gate.ac.uk/

- Provides a baseline set of customizable Language Engineering components that can be extended and/or replaced by users, for the following NLP tasks:
  - Tokenization
  - POS tagging
  - Sentence splitting
  - Named entity recognition
  - Co-reference resolution
  - Information Extraction
  - Machine learning, etc.

The GATE Family

http://gate.ac.uk/

- GATE includes:
  - an IDE, *GATE Developer* for NLP components bundled with an information extraction system and a set of other plugins
  - a web app, *GATE Teamware*: a collaborative annotation environment for semantic annotation projects
  - a framework, *GATE Embedded*: an object library optimized for inclusion in diverse applications giving access to all the services used by GATE Developer and more
  - an architecture: a high-level organizational picture of language processing software composition
  - a process for the creation of robust and maintainable services

LanguageWare

http://www.alphaworks.ibm.com/tech/lrw

- An NLP technology developed by IBM, that allows the processing of natural language text
- It comprises a set of Java libraries which provide a range of NLP functions:
  - Dictionary lookup
  - Language identification
  - Text segmentation/tokenization
  - Parsing
  - Lexical and morphological analysis
  - Entity and relationship extraction
  - Semantic analysis and disambiguation
  - POS tagging

LanguageWare Resources

http://www.alphaworks.ibm.com/tech/lrw

- Contains a set of configurable lexico-semantic resources which describe the characteristics and domain of the processed language
- LanguageWare Resource Workbench is a set of Eclipse-based customization tools and allows domain knowledge to be compiled into resources and incorporated into the analysis process
- LanguageWare can be deployed as a set of UIMA-compliant annotators, Eclipse plug-ins or Web Services

The Natural Language Toolkit

http://www.nltk.org/

- A suite of Python modules for natural language processing
- Includes modules for:
  - Classification
  - Parsing
  - Tokenization
  - Stemming
  - Tagging
  - Discourse checking
  - Information Extraction
  - Theorem proving, etc.

LingPipe http://alias-i.com/lingpipe/

- A toolkit for processing text using computational linguistics, used for tasks like:
  - Named entity recognition
  - Topic classification
  - Clustering
  - POS tagging
  - Sentence detection
  - Spelling correction
  - Language Identification
  - Word Sense Disambiguation
  - Information Retrieval (LSI), etc.
- Java API with source code and unit tests available

Stanford NLP Software

http://nlp.stanford.edu/software/index.shtml

- The Stanford NLP Research group offers a series of open-source NLP tools for text manipulation, implemented in Java:
  - The **Stanford Parser:** probabilistic natural language parsers
  - The **Stanford POS Tagger**: a maximum-entropy POS tagger
  - The **Stanford Named Entity Recognizer:** features for Named Entity Recognition
  - The **Stanford Classifier:** conditional loglinear classifier
  - **Topic Modeling Toolbox:** a suite of topic modeling tools