Text Analytics in the Legal Field

X. Chen, Y. Mountacif

December 20, 2016

1 Text Analytics

In this section, we present a succinct overview of current technologies for text data mining and analysis. Text analytics is the process of deriving high-quality information from (massive) text data through the use of linguistic, statistical, machine learning techniques. There can be a variety of applications in business intelligence, investigation, exploratory data analysis, or any other research activity requiring the need to model and structure information content of textual sources. First the data must be preprocessed, often in a manual and empirical way: samples of the text data are examined to see what cleaning operation must be done in order to make the data set ready for analysis. Then the aim is to structure the input text, identify patterns in it and interpret these patterns. Various techniques are currently used to tackle this:

- **Language modeling**. Probabilistic language model is a technique that takes a sentence as input, and assigns a probability to it. Language modeling is used in speech recognition, machine translation, part-of-speech tagging, parsing, handwriting recognition, information retrieval and other applications.

- **A topic model** is a type of statistical model for discovering the abstract *topics* that occur in a collection of documents. Topic modeling is a frequently used text-mining tool for discovery of hidden semantic structures in a text body.

- **Tagging**, also referred to as annotation, is one rapidly evolving technology that classifies and clusters data for analysis. Tagging works in conjunction with predictive analytics tools to grow and refine a knowledge base for mining unstructured text.

- **Summarization**, the goal of automatic summarization is to reduce a text document with a computer program in order to create a summary that retains the most important points of the original document. The main idea of summarization is to find a representative subset of the data, which can have the information of the whole initial set. Automatic data summarization technology belongs to the fields of machine learning and data mining.

- **The noisy channel model**, is a probabilistic framework used in spell checks, question answering, speech recognition and machine translation. This following flow explains how the noisy channel model works :

We observe a distorted message R, foreign language : f. We have a model on how the message is distorted (translation model t(f/e)) and also a model on which original messages are probable (language model p(e)). In order to figure out the message e, we can apply following formula.

Derivation of "noisy channel model" in a probabilistic framework using the Bayes rule:

$$\begin{aligned}\hat{e} &= \arg max_e p(e/f) \\ &= \arg max_e \frac{p(e/f)p(e)}{p(f)} \\ &= \arg max_e p(e/f)p(e)\end{aligned}$$

- **Text clustering and classification**. The goal is to gather objects into sets, such that the objects in a set will have some similarities or will be related to one another. We can use clustering for many application such as improving *search recall* (i.e. getting results that are related to a request) and *search precision* (i.e. getting accurate results for a request). The use of document clustering can enhance the efficiency of human browsing of a document collection when we can not formulate the search query. Text clustering can be achieved with various kinds of algorithms, each differing in how a cluster is defined and the way to find them:

  - k-means
  - bisecting k-means
  - hierarchical clustering
  - suffix tree clustering
  - relationship data, i.e. hyperlinks

  Clustering algorithms in text analysis aims at creating internally coherent sets of documents that are distinct from one another. In the other hand Classification is a form of supervised learning where the features of the documents are used to predict the content of documents and the type. k-means clustering is a method to partition n observation into k clusters where each observation belongs to the cluster with the nearest mean. This means represent the prototype of the cluster. The problem with this method consist on computational difficulty. The prototype of each cluster is defined as the minimum of the cost function: With a set of observations $(x_1, x_2, \ldots, x_n)$ where each observation is a multidimensional vector. The goal of k-means clustering is to partition the n observation into k sets $S = \{S_1, S_2, \ldots, S_k\}$ so as to minimize the sum of distance functions of each point in the cluster to the centroid.

$$\underset{S}{\operatorname{argmin}} \sum_{i=1}^{k} \sum_{\mathbf{x} \in S_i} \|\mathbf{x} - \boldsymbol{\mu}_i\|^2$$

  Where $\boldsymbol{\mu}_i$ is the mean of points in S_i. The algorithm proceeds by alternating between two steps:

  - the *assignment step*, at which each observation is placed in the cluster with the nearest mean,
  - the *update step*, at which new means are calculated to be the centroids of the observation in the new clusters.

  The algorithm converges when the assignments no longer change. However, there is no guarantee that it will converge to the global optimum, and the result may depend on the initial clusters. The complexity of this algorithm in d dimensions is NP-hard and take $O()$ for k clusters and n observations. There exist many software that implement this algorithm, for instance SciPy, Scikit-Learn, Torch, OpenCV. . .

- **Sentiment analysis**. Generally speaking, sentiment analysis, sometimes referred to as opinion mining, aims at determining the attitude of a speaker or a writer. It may be her judgment or evaluation, an effective state or an emotional communication. Sentiment is a feeling, an attitude, an emotion or opinion, it is a subjective impression and not a fact. For sentiment analysis, Natural Language Processing and statistics are used to extract, identify and characterize the sentiment content of a token. The major aim is classifying the polarity of a given text and decide if it is positive, negative, or neutral. One step forward is to go beyond binarity and instead look for subtler emotional states such as 'anger', 'sadness', and 'happiness' . . . In addition, sentiment analysis can be used to determine whether a text is objective or subjective. This application can be more difficult than polarity classification because the subjectivity of words and phrases will depend on the context of the situation. Moreover, results are dependent on the definition of subjectivity used when annotating texts, performance can be improved by removing objective sentences from a document before analyzing its polarity.

2 Toolboxes, Libraries and Other Software

- **General Architecture for Text Engineering**, GATE, has become one of the standard tools in text engineering and can assist lawyers to review legal documents. Given an input text, it can help extract words, sentences, paragraphs, people's names . . . Naively, extracting people's names is easily done: words starting with an uppercase letter, followed by a blank space and subsequently followed by one or more other word starting with an uppercase is very likely to refer to a person be it natural (human being) or legal (corporate entity, governmental body, organization . . .). Dictionaries of names or common organization types (court, board . . .) can help further distinguish between natural and legal persons. Given a series of emails and available metadata (e.g. sender's name, receiver's name . . .) one can extract other names in the email and repeat this operation for a series of emails. Visualizing data may provide insights on potential links between individuals, for instance. Placing a vertex (of a specific color, size and/or shape) for each email, and linking it to vertices (of different color, size and/or shape) representing the extracted names in the given email can help building the graph mapping the potential relationships between names.

- **Torch7** provides a Matlab-like environment for state-of-the-art machine learning algorithms. It is written in Lua and an underlying C/OpenMP/CUDA implementation. It includes lots of packages or neural networks, optimization, graphical models, image processing. It is mainly used for speech, image, video applications and large - scale machine learning applications. Torch 7 is a cross-platform framework, as it is written in LuaJIT, it is much easier to manipulate as language than python.

  For text analysis, Torch7 provides following techniques:

  - Part of speech
  - Chunking
  - Name entity recognition
  - Semantic role labeling
  - Parsing

  Torch7 is mainly used in Facebook for text understanding. However, if we do not need a very powerful tool, we can just use a python library in NLP, as we have to learn this language from scratch.

- **Word2Vec** is not a deep neural network, it is a group of related models which turns text into a numerical form that deep nets can understand. The purpose and usefulness of Word2vec is to group the vectors of similar words together in vectorspace. That is, it detects similarities mathematically. Word2vec creates vectors that are distributed numerical representations of word features, features such as the context of individual words. Given enough data, usage and contexts, Word2vec can make highly accurate guesses about a word's meaning based on past appearances. Those guesses can be used to establish a word's association with other words (e.g. "man" is to "boy" what "woman" is to "girl"), or cluster documents and classify them by topic. Those clusters can form the basis of search, sentiment analysis and recommendations in such diverse fields as scientific research, legal discovery, e-commerce and customer relationship management. The output of the Word2vec neural net is a vocabulary in which each item has a vector attached to it, which can be fed into a deep-learning net or simply queried to detect relationships between words.

  For the text analysis, Word2Vec provides following techniques:

  - Tokenization
  - Text classification
  - Test clustering

- **Natural Language Toolkit, NLTK** If your language of choice is Python, then look no further than NLTK for many of your NLP needs. It includes capabilities for tokenizing, parsing, and identifying named entities as well as many more features.

  For the text analysis, NLTK provides following techniques:

  - Tokenization

  - Tagging
  - Parsing
  - Text classification
  - Named entity recognition

- **Caffe** is an open-source library, public reference models, and working examples for deep learning. It is mainly used for vision models and beyond to detection, vision + language, and segmentation models. It is written in C++. Cafe library contains several algorithms such as :

  - Automation differentiation
  - Recurrent Neural Network
  - Convolutional Neural Network
  - RBM/DBNs

  Caffe can be a good tool if we need treat images however, there are couple libraries or frameworks much better for the NLP use

- **MALLET Package:** MALLET is a Java-based package used for statistical natural language processing, it includes different tools for document classification, such as, efficient routines for converting text to features, and also code for evaluating classifier performance. The package has many applications such as document classification, clustering, information extraction and topic modeling. It include also tools for sequence tagging for applications such as named-entity extraction from text. In addition, MALLET contains topic modeling toolkit useful for analysing large collections of unlabeled text.

- **Apache Lucene and Solr** While not technically targeted at solving NLP problems, Lucene and Solr contain a powerful number of tools for working with text ranging from advanced string manipulation utilities to powerful and flexible tokenization libraries to blazing fast libraries for working with finite state automatons. On top of it all, you get a search engine for free! Python / serveur : XML/http/ JSON/Ruby APIs

3 VIP and Downraid application

- **VIP** aims to build a network between the clients and lawyers based on their interests, events that they are part of, their jobs and so on. For this application, we can either use Word2Vec or NLTK as tools, written in Python. We do not need a powerful deep learning network to have relevant data. Once we managed to have enough data, we just need display it under a GUI (website view or console view), both work.

- **Downraid** aims to analyse professional emails of lawyers or clients and try to understand the underlying of words behind. For this project, we can begin using word2vec which is written in Python and it provides some techniques such sentiment analysis. However if we would like to build a complete software, we could use Torch7, which provides a complete cross-platform framework and couple interesting techniques in text analytics.