# Assessment and Visual Analysis of Trends using Article Reviews


Jamshed Khan, Padmanabhan Rajendrakumar, Jaideep Misra

## Table of Contents

## 1) Problem Statement:

In the age of Big Data, an estimated 2.5 quintillion bytes of data is generated every day and a huge amount of this is of a textual nature. With scores of documents available on the web and more pouring in day after day, how can one make sense of a general summary without actually diving in and reading every word? Searching for insights from such an enormous amount of information can become very tedious and time-consuming.

Imagine that you have a massive amount of text and wanted to quickly understand the general trends, topics, or themes that it contains. This concept can take any form such as having to read a huge book (pdf document) or perhaps going through user feedback and comments at work. Fortunately, there exists a solution to this scenario called *topic modelling.*

Consider a real-world example: a book store vendor receives a set of new books for his store and as per policy, must segregate books according to their genre across the shop. For example: Fiction, Fantasy, Romance etc. In some cases, one could get an idea of the genre by looking at a summary or even the title itself. However, when this is not possible, the vendor would have to read at least some part of the book in order to understand what it is about. However, with topic modelling, a book could be summarized automatically and in a negligible amount of time thus enabling it to be sorted efficiently. The same scenario works in a digitized world too. For instance, the book store is replaced by a book selling website and the books themselves would be text documents such as pdfs.

*Topic modeling* is a type of unsupervised machine learning that makes use of clustering to find latent variables and structures in textual data. In short, it is an approach for finding topics or themes across large amounts of text. Topic modeling is great for document clustering, information retrieval from unstructured text, and feature selection. A topic model is a type of algorithm that scans a set of documents (also known as a corpus), examines how words and phrases co-occur in them, and automatically "learns" groups or clusters of words that best characterize those documents.

For our project, we perform topic modelling on news articles. There are several influences behind this choice of documents. For one, news articles are extremely abundant online. It is estimated there are approximately 2 million news stories being released onto the internet every day. Besides, there are a wealth of user comments and opinions on these sites that also allow us to perform sentiment analysis in order to better understand these news topics and analyze relations between them.

## 2) Objectives:

While doing our project, we have kept the following primary objectives in mind

- To analyze news articles and user comments on those articles from leading news sites in order to understand and examine the most prevalent or 'trending' topics at any given time.
- To perform text mining and discover the hidden semantic structures in a collection of procured documents.

- Extraction of meaningful information and classification of named 'entities' mentioned in unstructured text and organizing them into pre-defined categories such as person names, trending events etc.
- To examine people's feeling or sentiments about current situations and happenings such as politics, government, elections etc. from their comments and opinions.

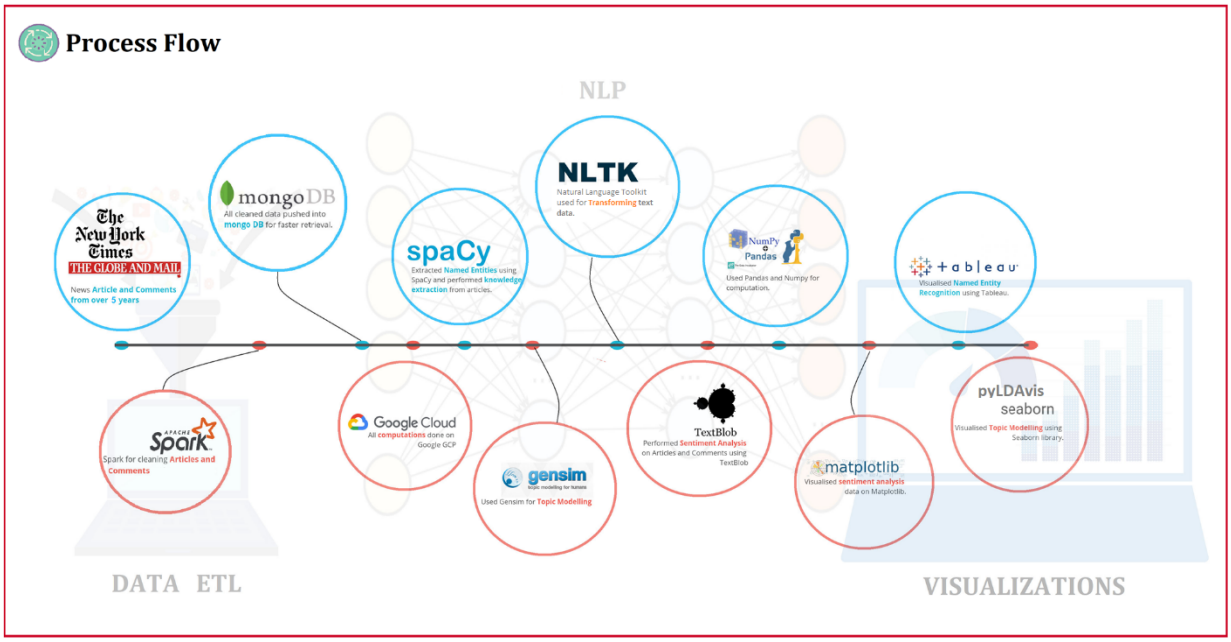
A few more insights about the various applications of topic modelling are as follows,

1) **Information Retrieval** - Topic models, which model each document as a mixture of topics and each topic as a mixture of words, offer an interesting framework to model documents in information retrieval. Popular topic models such as probabilistic latent semantic analysis (PLSA) and latent Dirichlet allocation (LDA) have been both explored to improve document language models.
2) **Topic Labelling -** Topics can have their own labels such as "information technology" or "Art". These are convenient descriptors, but completely removed from the raw distribution over words. Thus, it is often useful to assign labels to topics within an interface. In contrast to the previous visualization approaches, labeling focuses on showing not the original words of a topic but rather a clearer label more akin to what a human summary of the data would provide.
3) **To gain scholarly insights –** Consider a scenario where we are training a topic model jointly on a combined corpus of original journal articles in English and a similar corpus of Wikipedia articles but in German. The model provides insight into the relative concentration of scholarly interests across the two language communities. The German-language journal articles contain more work on law and oratory, themes that are present in the English-language articles but less prevalent. The model also shows a large increase in interest in poetry in the German journal in the period following the second world war. In the English journals there is a large increase starting in the 1980s in cultural and economic studies along with critical theory, which does not appear in the German journals.
4) **In the Humanities -** While close reading is a foundational tool in the study of literature, it is necessarily limited by its scale. We value literature because it is one of the best ways to capture the spirit of an age, and the experiences of those who lived through it. But standard close reading methods require narrow focus and thorough interpretation. Topic models complement close reading in two ways, as a survey method and as a means for tracing and comparing large-scale patterns.
5) **Qualitative Analyses -** A common task in qualitative social science is to develop high-level theories that explain social processes based on low-level observations, such as field reports or ethnographic notes. One way to operationalize this process is the grounded theory method. Grounded theory describes a process for iteratively developing theories through repeated reading of source material. As such the topic model can best be thought of as a tool for applying grounded theory more efficiently and with greater insulation from human biases.
6) **Sentiment Analysis -** A useful way to think of the application of topic models in quantitative social science is as a means of deriving a numeric variable from text. Specific topics in a document can stand in for themes that may not be otherwise easily measurable. These inferred topic variables can then be added to standard statistical methods to find connections between topics and non-textual variables. As a motivating example, we first

consider sentiment analysis. Here, the goal is to determine the "sentiment"—e.g., positive or negative opinions—associated with a piece of text. For example, "Chipotle is great!" would be associated with positive sentiment, while "Chipotle made me sick" would be associated with negative sentiment.

7) **Social Networks and Media -** Sometimes we are interested in metadata that describe the relationships between documents: which users follow each other on Twitter, which scientific papers cite each other, or which webpages link to each other. There are intrinsic groups of documents and links that are analogous to the topics in topic models, except that the links are "shared" between documents.
8) **Machine Translation -** The most frequent application of multilingual topic models is in machine translation. Given a text input in one language (source language), statistical machine translation tries to find a similar sequence of words in another language (target language). Modern machine translation systems use millions of training examples to learn the translation rules and apply these rules on the test data. Topic models are useful in this application when they can help to inform word meaning and word choice in specific contexts. While the translation rules are learned in local context, these systems work best when the training corpus has a consistent domain, such as a genre (e.g., sports, business) or style (e.g., newswire, blog-posts).

## 3) Process Flow/Data Pipeline:




The process flow is as seen in the image above and can be summarized by the following steps:

- Collection of News articles and user comments from the developer API of New York Times website, The Globe and Mail as well as the SOCC corpus provided by the SFU Discourse Lab.
- Cleaning and filtering the data using Apache Spark before storing the data in mongoDB.
- Identification of the top 25 topics from data by performing Topic modelling with the two NLP methods of Latent Dirichlet Association (LDA) and Latent Semantic Indexing.

- Extraction of named entities such as people, organizations, countries, products etc. using the pretrained model 'en_core_web_lg' from spaCy with an accuracy of 86.25%.
- Identification of people's reactions by performing sentiment analysis on the data. Using NLTK and TextBlob sentiment analysis libraries we identify polarity and subjectivity of people's comments along with the general trend of responses.
- We report our findings by visualizing the results using Tableau, Seaborn, PyLDAvis and Matplotlib.

## 4) Technology Stack and Libraries Used:

| | |
|---|---|
| **Dev. Environment** | Linux, Google Cloud Platform |
| **Data Collection** | NY Times APIs, Globe and Mail Canada CSVs |
| **Data Preprocessing/Cleaning** | Apache Spark, NLTK Libraries, Numpy and Pandas |
| **Data Storage** | MongoDB |
| **Cloud Computing GPU** | Google Cloud Platform (Nvidia-Tesla-P100) |
| **NLP Libraries** | SpaCy, Gensim, TextBlob |
| **Visualization** | Tableau, Matplotlib, Seaborn, PyLDAvis |

## 5) Methodology

### 1.Topic Modelling

A topic model is a type of statistical model for discovering the abstract topics that occur in a collection of text documents. Topic modeling is a frequently used for text-mining and discovering hidden semantics in text documents. Topic modeling is the process of identifying topics in a set of documents. This can be useful for search engines, customer service automation, and any other instance where knowing the topics of documents is important.

Dataset: We used the Globe and Mail Canada and New York Times data set for topic modelling. Use used the NYT API for New York Times and Input CSVs for Globe and Mail Canada. The data span over 10 years of news articles and comments collected till March 2019.

Data Cleaning/ Preprocessing: The input data was cleaned using Apache Spark and other NLP tools like Gensim and NLTK. The text documents were stemmed, lemmatized and tokenized and stored in CSVs and in MongoDB documents.

Data Storage: The data was stored in MongoDB as collections for faster retrieval. This is much more efficient as we can parallelize the reading and preprocessing of the text data using PySpark. MongoDB is highly scalable and fault tolerant, so it can easily adapt as the data grows.
We perform topic modelling using two Genism models – LDA and LSI

LDA is a form of unsupervised learning that views documents as bags of words LDA works by first making a key assumption: the way a document was generated was by picking a set of topics and then for each topic picking a set of words. It builds a topic per document model and words per topic model, modeled as Dirichlet distributions. LDA or Latent Dirichlet Allocation is a "generative probabilistic model" of a collection of composites made up of parts.

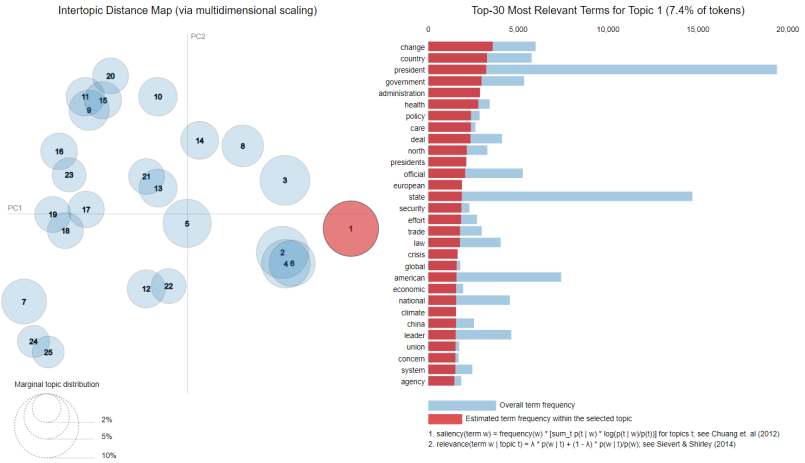
Training: We trained the input text on GENSIM's LDA model for 50 passes for the top 25 topics.

Latent semantic analysis (LSA/LSI) is a technique in natural language processing, in particular distributional semantics, of analyzing relationships between a set of documents and the terms they contain by producing a set of concepts related to the documents and terms. LSA assumes that words that are close in meaning will occur in similar pieces of text (the distributional hypothesis).LSI (also known as Latent Semantic Analysis, LSA) learns latent topics by performing a matrix decomposition (SVD) on the term-document matrix.

Training: We trained the input text on GENSIM's LSI model with a decay of 0.5.
The accuracy we got for LDA was higher than that of LSI's. We got a perplexity score of approximately -8 and a Coherence Score of 0.6.

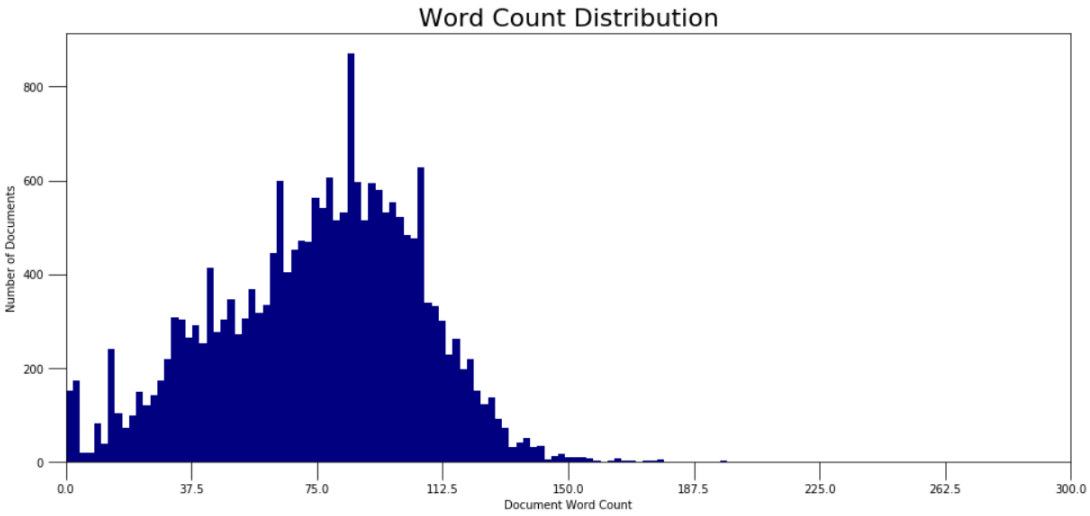
EDA and Visualization:

1) The LDA model was visualized using PyLDAvis library. pyLDAvis is designed to help users interpret the topics in a topic model that has been fit to a corpus of text data. The package extracts information from a fitted LDA topic model to inform an interactive web-based visualization. It displays the Intertopic distance and the Top 30 salient words for each topic.



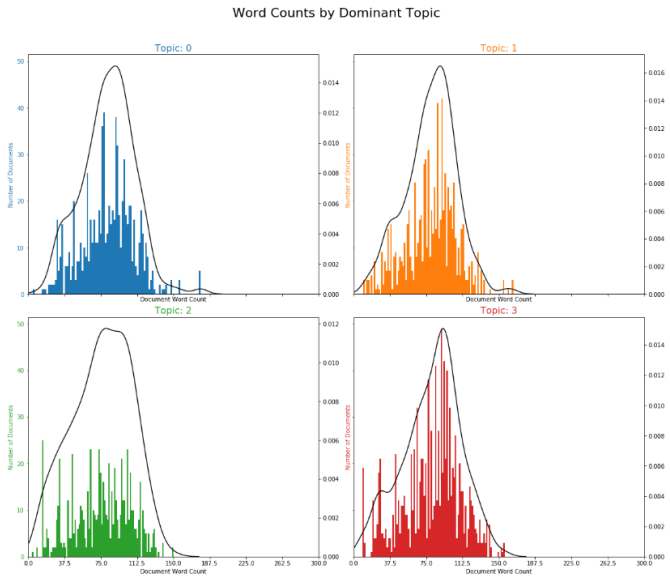

2) Word Count Distribution

We then visualized the frequency of the words by plotting a Word Count distribution.



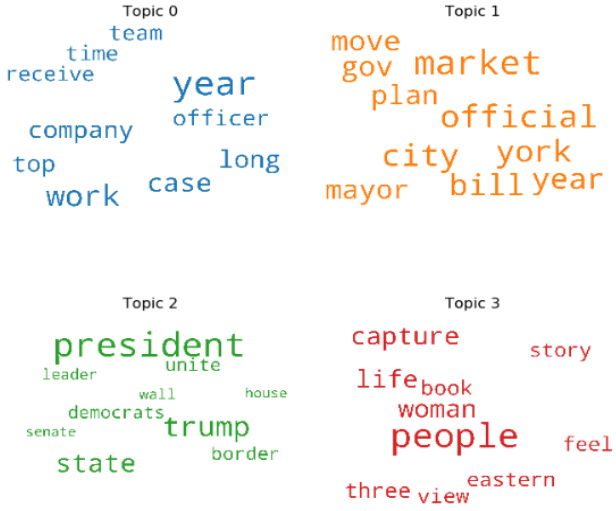

3) Word Counts by Dominant Topic

The below figure displays the Word Count Distribution by dominant topics. We can see how the dominant topics varied from one model to another



4) Word Cloud

The below figure displays the word cloud with the most prevalent words in the top 4 topics.




## 2.Named Entity Recognition

Named Entity Recognition (NER) is a process where an algorithm takes as input a string of sentence or paragraph and extracts relevant entities which can be a place, a person or an organisation. It is not limited to just few categories as mentioned above but the possibilities are many by using “Parts of Speech” tags which we will see later.
We have taken the task of using this process on News Articles from web
and the comments on them to analyse which people were trending these articles and what were the people talking about. Apart from extracting persons we also extracted the below entities:

| TYPE | DESCRIPTION |
|---|---|
| PERSON | People, including fictional. |
| NORP | Nationalities or religious or political groups. |
| FAC | Buildings, airports, highways, bridges, etc. |
| ORG | Companies, agencies, institutions, etc. |
| GPE | Countries, cities, states. |
| LOC | Non-GPE locations, mountain ranges, bodies of water. |
| PRODUCT | Objects, vehicles, foods, etc. (Not services.) |
| EVENT | Named hurricanes, battles, wars, sports events, etc. |
| WORK_OF_ART | Titles of books, songs, etc. |
| LAW | Named documents made into laws. |
| LANGUAGE | Any named language. |
| DATE | Absolute or relative dates or periods. |
| TIME | Times smaller than a day. |
| PERCENT | Percentage, including "%". |
| MONEY | Monetary values, including unit. |
| QUANTITY | Measurements, as of weight or distance. |
| ORDINAL | "first", "second", etc. |
| CARDINAL | Numerals that do not fall under another type. |

We implemented our model using spaCy which is a free, open-source library for advanced Natural Language Processing (NLP) in Python. spaCy is designed specifically for production use and helps you build applications that process and "understand" large volumes of text. It can be used to build information extraction or natural language understanding systems, or to pre-process text for deep learning.

The spaCy statistical model we used is "en_core_web_lg" where:

- en: The model has been trained on English language.
- Core:For general-purpose model with vocabulary, syntax, entities and word vectors.
- Web: Type of text the model is trained on.
- lg: Model size indicator(large).

## Collection and Cleaning:

We used SOCC data for Named Entity Recognition:

### 1) socc_articles.csv

This CSV contains information about The Globe and Mail articles in our dataset. This has approximately 10,339 opinion articles (editorials, columns, and op-eds) for a five-year period (from January 2012 to December 2016).

### 2) socc_comments.csv

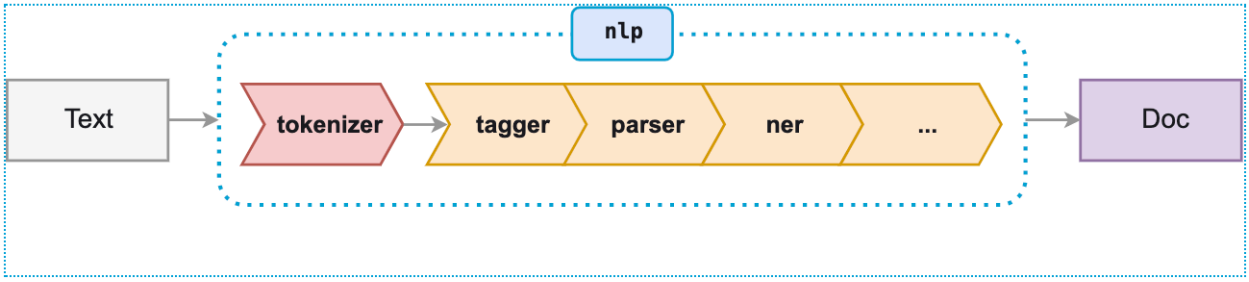
We have 663,173 unique comments from 303,665 comment threads, from the main Canadian daily in English, The Globe and Mail.

For cleaning the data, we used spark and removed some HTML tags from the articles.
Since spaCy has a built-in tokenizer we do not need to tokenize and lemmatize the sentences.

**Named Entity Detection Using spaCy:**

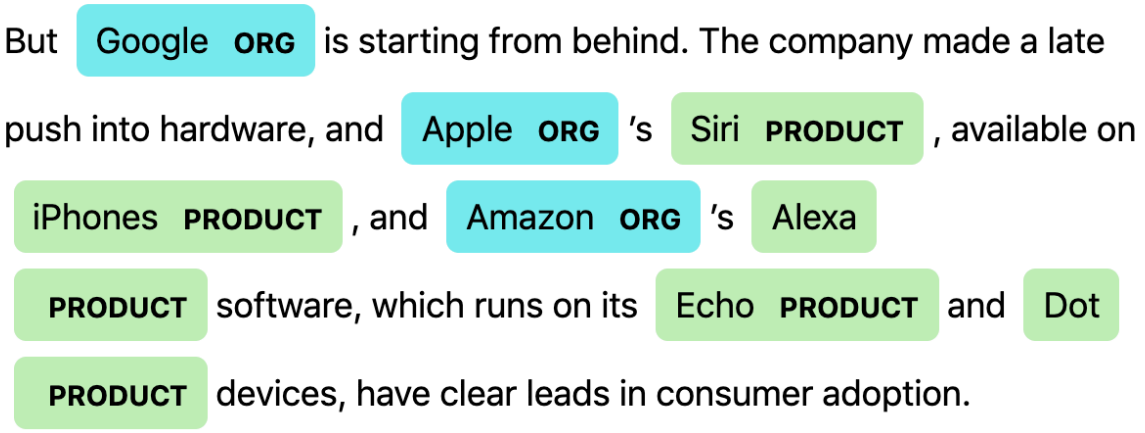
**1) spaCy pipeline:**

When you call nlp on a text, spaCy first tokenizes the text to produce a Doc object. The Doc is then processed in several different steps – this is also referred to as the processing pipeline. The pipeline used by the default models consists of a tagger, a parser and an entity recognizer. Each pipeline component returns the processed Doc, which is then passed on to the next component.




Running spaCy's model on a text would result the following. This has been done using the spaCy's displacy visualisation which shows the tagged entities:

text = """But Google is starting from behind. The company made a late push into hardware, and Apple's Siri, available on iPhones, and Amazon's Alexa software, which runs on its Echo and Dot devices, have clear leads in consumer adoption."""




We passed our text to the nlp pipeline of spaCy and used model "en_core_web_lg" to tokenize and tag each token with its part of speech recognition.

## 2) Part-of-Speech Tagging

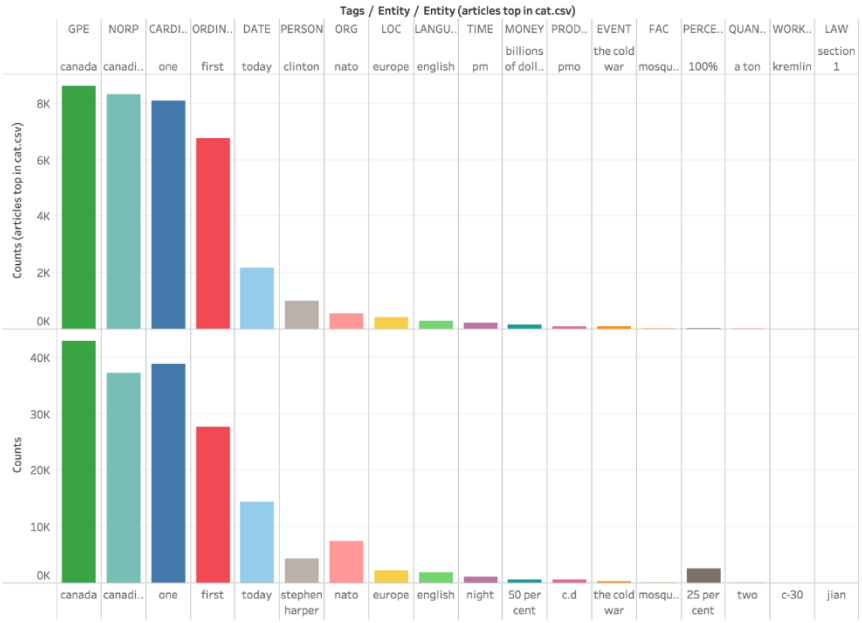
After tokenization, spaCy can parse and tag a given Doc. This is where the statistical model comes in, which enables spaCy to make a prediction of which tag or label most likely applies in this context. A model consists of binary data and is produced by showing a system enough examples for it to make predictions that generalize across the language – for example, a word following "the" in English is most likely a noun.

```
import spacy

nlp = spacy.load('en_core_web_sm')
doc = nlp(u'Apple is looking at buying U.K. startup for $1 billion')

for token in doc:
    print(token.text, token.lemma_, token.pos_, token.tag_, token.dep_,
          token.shape_, token.is_alpha, token.is_stop)
```

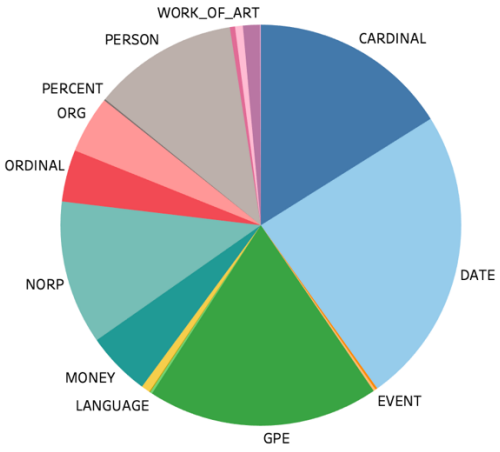
```
Output: Apple Apple PROPN NNP nsubj Xxxxx True False
is be VERB VBZ aux xx True True
looking look VERB VBG ROOT xxxx True False
at at ADP IN prep xx True True
buying buy VERB VBG pcomp xxxx True False
U.K. U.K. PROPN NNP compound X.X. False False
startup startup NOUN NN dobj xxxx True False
for for ADP IN prep xxx True True
$ $ SYM $ quantmod $ False False
1 1 NUM CD compound d False False
billion billion NUM CD pobj xxxx True False
```

## Results and Explanation:

## 1) Tableau Visualisations:




This shows the comparison of the named entities observed in Articles(top) and Comments(bottom). Apart from showing the count it also shows the top named entities in each category.

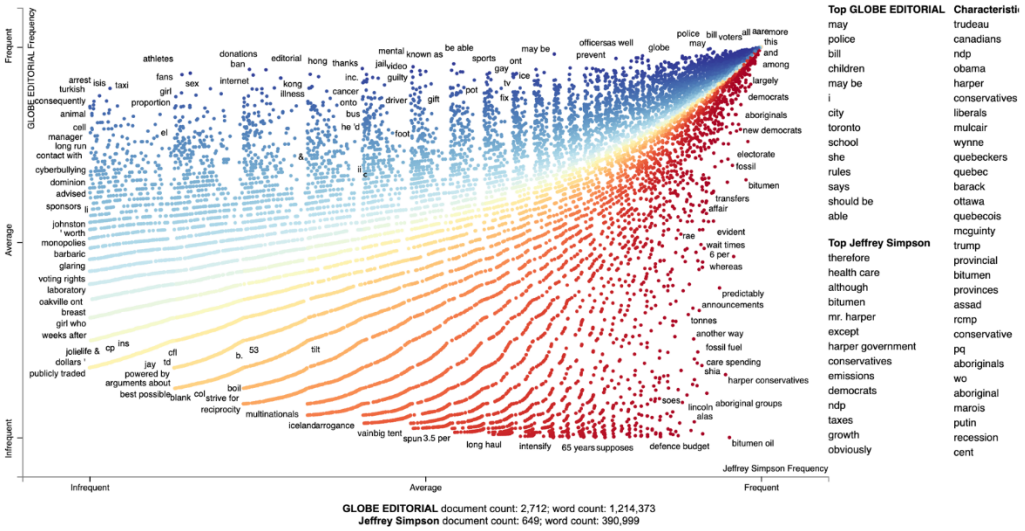



The above pie chart shows the distribution of the amount of named entities in both articles and comments combined.

## 2) Scattertext Visualisation:

Visualizing term associations for top 2 article authors

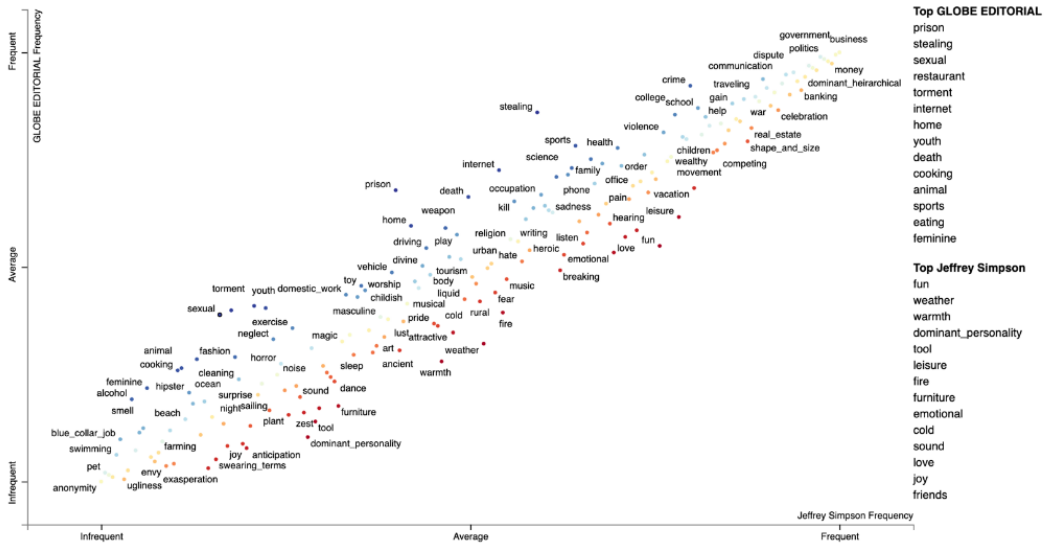
---




**GLOBE EDITORIAL** document count: 2,712; word count: 1,214,373
**Jeffrey Simpson** document count: 649; word count: 390,999

This chart shows the words used by the top 2 authors and outputs some notable term associations.

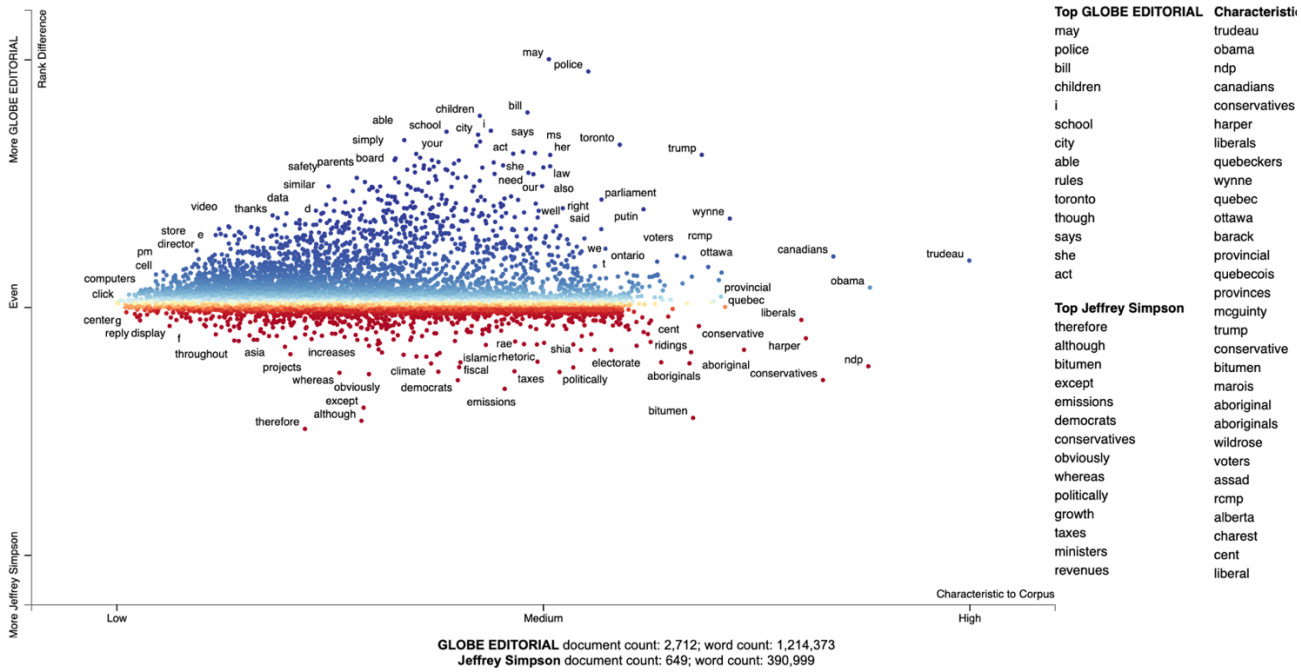
Visualizing Empath topics and categories from top 2 article authors




**GLOBE EDITORIAL** document count: 2,712; word count: 1,214,373
**Jeffrey Simpson** document count: 649; word count: 390,999

We created a Corpus of extracted topics and categories rather than unigrams and bigrams. For which we used the FeatsOnlyFromEmpath feature extractor.(Top_2_Authors-Empath.html).

## 3) Visualising Ordering of Terms by Corpus Characteristics:

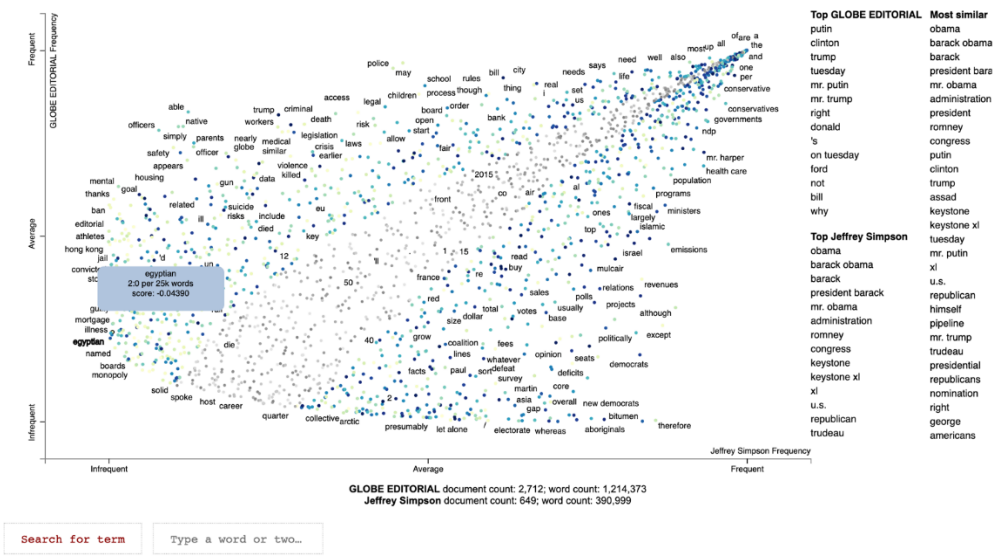



Often the terms of most interest are ones that are characteristic to the corpus as a whole. These are terms which occur frequently in all sets of documents being studied, but relatively infrequent compared to general term frequencies.
Corpus characteristics is the difference in dense term ranks between the words in all of the documents in the study and a general English-language frequency list.
(characteristic_chart.html)

## 4) Visualising genism similarity:

This graph shows how the two authors talked about Obama. Instead of "Obama" we can replace it with any topic one would like to see.
In this configuration of Scattertext, words are colored by their similarity to a query phrase.
This is done using spaCy-provided GloVe word vectors (trained on the Common Crawl corpus).
The cosine distance between vectors is used, with mean vectors used for phrases.

The calculation of the most similar terms associated with each category is a simple heuristic. First, sets of terms closely associated with a category are found. Second, these terms are ranked based on their similarity to the query, and the top rank terms are displayed to the right of the scatterplot.




To view the interactive graphs please see the HTML files in the output folder.

## 3. Sentiment Analysis

Sentiment Analysis also known as *Opinion Mining* is a field within Natural Language Processing (NLP) that builds systems that try to identify and extract opinions within text. Usually, besides identifying the opinion, these systems extract attributes of the expression e.g.:

- *Polarity*: if the speaker expresses a *positive* or *negative* opinion,
- *Subject*: the thing that is being talked about,
- *Opinion holder*: the person, or entity that expresses the opinion.

Text information can be broadly categorized into two main types: *facts* and *opinions*. Facts are *objective* expressions about something. Opinions are usually *subjective* expressions that describe people's sentiments, appraisals, and feelings toward a subject or topic.

Sentiment analysis, just as many other NLP problems, can be modeled as a classification problem where two sub-problems must be resolved:

- Classifying a sentence as *subjective* or *objective*, known as **subjectivity classification**.
- Classifying a sentence as expressing a *positive*, *negative* or *neutral* opinion, known as **polarity classification**.

In an opinion, the entity the text talks about can be an object, its components, its aspects, its attributes, or its features. It could also be a product, a service, an individual, an organization, an event, or a topic. For example: *"The battery life of this camera is too short."*

A negative opinion is expressed about a feature (battery life) of an entity (camera).

Sentiment analysis can be applied at different levels of scope:

- **Document level** sentiment analysis obtains the sentiment of a complete document or paragraph.

- **Sentence level** sentiment analysis obtains the sentiment of a single sentence.
- **Sub-sentence level** sentiment analysis obtains the sentiment of sub-expressions within a sentence.

There are many types and flavors of sentiment analysis and SA tools range from systems that focus on polarity (positive, negative, neutral) to systems that detect feelings and emotions (*angry*, *happy*, *sad*, etc) or identify intentions (e.g. *interested* v. *not interested*). Following are some of the types:

- **Fine-grained Sentiment Analysis -** Sometimes you may be also interested in being more precise about the level of polarity of the opinion, so instead of just talking about *positive*, *neutral*, or *negative*opinions you could consider the following categories:
  Very positive
  Positive
  Neutral
  Negative
  Very negative

This is usually referred to as fine-grained sentiment analysis. This could be, for example, mapped onto a 5-star rating in a review, e.g.: Very Positive = 5 stars and Very Negative = 1 star.Some systems also provide different flavors of polarity by identifying if the positive or negative sentiment is associated with a particular feeling, such as, anger, sadness, or worries (i.e. negative feelings) or happiness, love, or enthusiasm (i.e. positive feelings).

Emotion detection - Emotion detection aims at detecting emotions like, happiness, frustration, anger, sadness, and the like. Many emotion detection systems resort to lexicons (i.e. lists of words and the emotions they convey) or complex machine learning algorithms. One of the downsides of resorting to lexicons is that the way people express their emotions varies a lot and so do the lexical items they use. Some words that would typically express anger like shit or kill (e.g. in your product is a piece of shit or your customer support is killing me) might also express happiness (e.g. in texts like This is the shit or You are killing it).

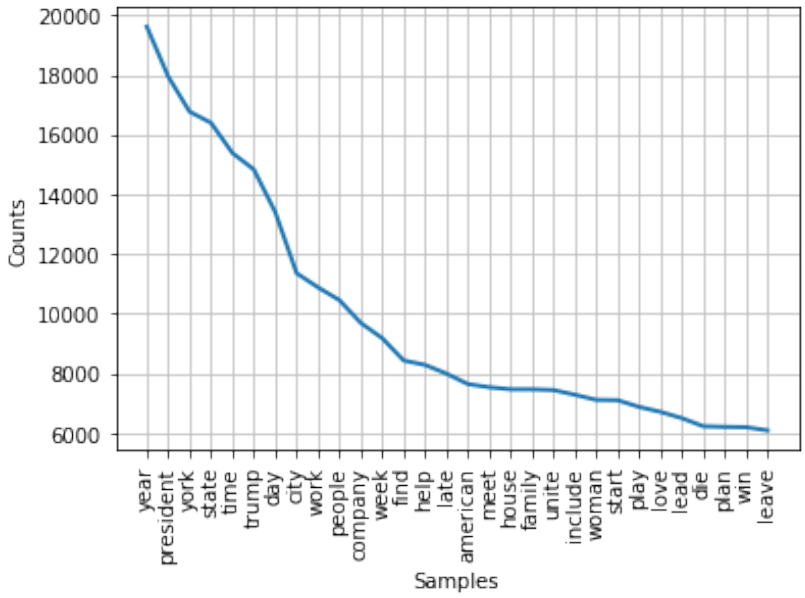
Aspect-based Sentiment Analysis - Usually, when analyzing the sentiment in subjects, for example products, you might be interested in not only whether people are talking with a positive, neutral, or negative polarity about the product, but also which particular aspects or features of the product people talk about. That's what aspect-based sentiment analysis is about. For example:
"The battery life of this camera is too short."
The sentence is expressing a negative opinion about the camera, but more precisely, about the battery life, which is a particular feature of the camera.

Intent analysis - Intent analysis basically detects what people want to do with a text rather than what people say with that text. Look at the following examples:
“Your customer support is a disaster. I’ve been on hold for 20 minutes”.
“I would like to know how to replace the cartridge”.
“Can you help me fill out this form?”
A human being has no problems detecting the complaint in the first text, the question in the second text, and the request in the third text. However, machines can have some problems to

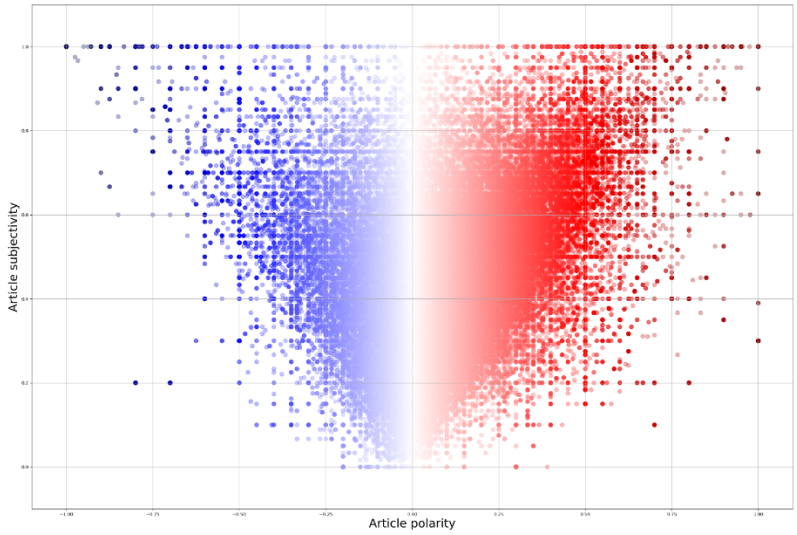
identify those. Sometimes, the intended action can be inferred from the text, but sometimes, inferring it requires some contextual knowledge.

Multilingual sentiment analysis - Multilingual sentiment analysis can be a difficult task. Usually, a lot of preprocessing is needed and that preprocessing makes use of a number of resources. Most of these resources are available online (e.g. sentiment lexicons), but many others have to be created (e.g. translated corpora or noise detection algorithms). The use of the resources available requires a lot of coding experience and can take long to implement. An alternative to that would be detecting language in texts automatically, then train a custom model for the language of your choice (if texts are not written in English), and finally, perform the analysis.

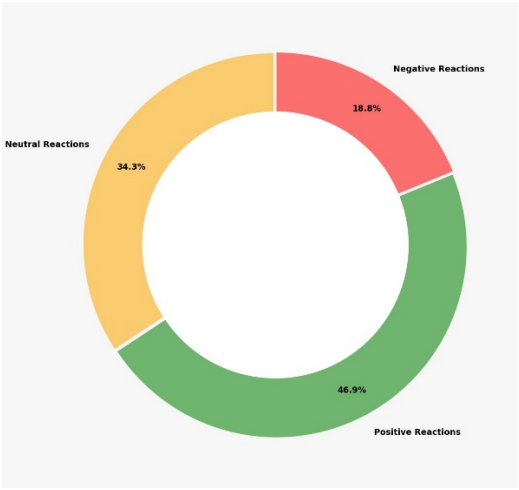
1) Frequency of words present in the comments




2) Sentiment Analysis of User comments on news articles. The given figure is a scatterplot that shows the index of polarity on the x-axis and subjectivity of the comments on y-axis.
The polarity takes values -1.00 which indicates most negative comments to +1.00 which stands for most positive comments. 0.00 stands for neutral comments. Subjectivity indicates how ‘subjective’ is the manner of the comment. For eg. ‘I really love this phone’ is a subjective comment while ‘This is a phone’ is an objective comment.




3) The given donut chart summarizes the nature of people's comments (or reactions) giving a percentage of positive, negative or neutral reactions.




## 6) Lessons Learnt:

1. LDA is a better model compared to LSI. It gives better accuracy and results. However, training an LDA model takes much longer than the LSI model.
2. The topics get closer to the point with the number of passes through the entire text data. In other words, topics get narrower when iterating through the document's multiple times.
3. Using NER and POS (Parts of Speech) tags in SpaCy model, we can extract more information other than just Named Entities. For example, extracting skills from Resume.
4. Learnt how to use visualizations using ScatterText such as term associations, corpus characteristics and visualizing query based categorical differences.

5. Most of the comments made by people are subjective rather than objective. The number of positive and neutral far exceeded the number of negative comments

## 7) Summary:

This project involves building visualizations which would help us demonstrate what is being talked about in News articles and comments and how it is being perceived. We gathered data from different sources from over 5 years and performed Topic Modelling, Sentiment Analysis and Named Entity Recognition on it. This helped us cover each aspect of Natural Language Processing which is necessary to understand huge data in just a quick glance from the visualisations. We learnt about the most common topics, sentiments of people on it, which entities were used the most while describing a topic, to name a few.