Quantifying Change in Financial Reports
Goldman Sachs Capstone Project

Final report

Abstract

As part of federal securities laws, the Securities and Exchange Commision (SEC) requires all the publicly-listed companies to submit three quarterly (10-Q) and one annual filing (10-K) each year. These reports provide important information on the company's performance, risks and future plans, so it is crucial for investors to evaluate them to make better and timely investments decisions. However, given the large number of companies and the length of these reports, reading through all of them can be quite time consuming and costly. To overcome this problem, we have created a visual tool which measures the lexical and semantic changes between documents of consecutive years. The lexical changes are measured by computing the cosine distance of the TF-IDF representation of the documents, while semantic changes are computed as distance between the topic vectors generated by an LDA and a separate word embeddings model. With this tool, the analyst can simply lookup the CIK of a given company and assess the level of change on documents through the years via a number of visualizations. In addition, the analyst also has access to a heat map showing the level of change of all companies for all years. Using the heat map the analyst can detect which companies have performed the largest changes in their documents to prioritize them for analysis.

our team seeks to evaluate the changes in the quarterly reports as well as annual reports and develop a systematic algorithm to quantify these changes.

1.Introduction

As part of federal securities laws, the Securities and Exchange Commision (SEC) requires all the publicly-listed companies to disclose related information on a quarterly basis. The information includes essential financial statements, managerial discussion regarding the company's operations result, risk disclosure and sometimes forward-looking statements for future quarters. Each publicly-listed company must submit three quarterly filings (10-Q) and one annual filing (10-K) each year. The quarterly and annual reports have long served as an important official source for stakeholders and potential investors to gain insights on the financial position of companies. The market will respond to the information and one can always observe greater-than-usual fluctuation in a company's stock price and abnormal returns around its earnings announcements. Hence, it is crucial for investors to evaluate the information in the financial reports correctly in order to make better investments.

In this project, our team seeks to evaluate the changes in the quarterly reports as well as annual reports and develop a systematic algorithm to quantify these changes. Particularly, we focus our scope on the identifying the changes in the text corpora, for that the industry has already done significant work analyzing the changes in the financial statements. Information related to numbers in financial statements can be quickly processed by financial analysts at Goldman Sachs and there are well-established accounting models to evaluate it. However, it is relatively more difficult to process textual information and reach a conclusion in a limited time. Therefore, we only focus on text because this is what is most needed by the investors and it also reduces our scope.

The project can be divided into two phases. First, we will build web parsers to extract all 10-K and 10-Q filings data (2011-2015) from SEC EDGAR system, and warehouse the data in a structured way. Next, we will utilize existing methodology and researches in machine learning, natural language processing to perform the analysis and to develop the scoring algorithm.

This first report explains the initial steps that we have implemented for building parsers, exploratory analysis on the financial reports, the preliminary algorithm we defined and discussion of next steps.

Extensive prior researches have been done in the past decade on the topic of processing financial news, earnings calls and public sentiment around earnings announcements. Bulter and Keselj (2009) used N-gram technique to decompose the annual filings and developed a readability score. They created a SVM classifier to predict stock returns around earnings and the experimental portfolio outperformed benchmark from 2003 to 2007. Kogan et al (2009) analyzed the market risk section of annual reports and used TF-IDF to create feature matrix. A regression model was formulated on the stock price volatility around earnings and TF-IDF matrix. Ahmad and Zinzalian (2010) also predicted stock price volatility using similar TF-IDF matrix of earnings call transcripts.

One may observe that word frequencies (or N-grams) and TF-IDF matrices are widely used in analyzing corpora of the same topic. Hence, we will start exploring the data by transforming corpora into TF or TF-IDF matrices.

2.The Data Set

There are about 180,000 filings (10-K and 10-Q) in total from 2011 to 2015. A typical quarterly report (10-Q) is composed of two parts. The first part includes four sections: Financial Statements, Management's Discussion and Analysis of Financial Condition and Results of Operations, Quantitative and Qualitative Disclosures About Market Risk, Controls and Procedures. The second part includes Legal Proceedings, Risk Factors, Unregistered Sales of Equity Securities and Use of Proceeds, Mine Safety Disclosures and Other Information.

A company files a more comprehensive annual report (10-K) each year after its fiscal year ends. In addition to the sections included in the quarterly reports, the annual reports explain in detail its business models, product lines and its marketing strategies. Also, it disclose Executive Compensation, Security Ownership of Certain Beneficial Owners and Management and Related Stockholder Matters, Certain Relationships and Related Transactions and Director Independence, and Principal Accounting Fees and Services.

As a result of compliance and sensitivity of the market, companies are often cautious in composing the financial reports and reluctant to change expressions. One may find most of the paragraphs are similar in terms of sentences and vocabulary. Sometimes only numbers and percentages are updated in a report compared on a yearly basis. Hence, wherever a new sentence emerges in a paragraph or there is an additional paragraph appears in the section, we assume that reflects significant and important changes happened in the quarter.

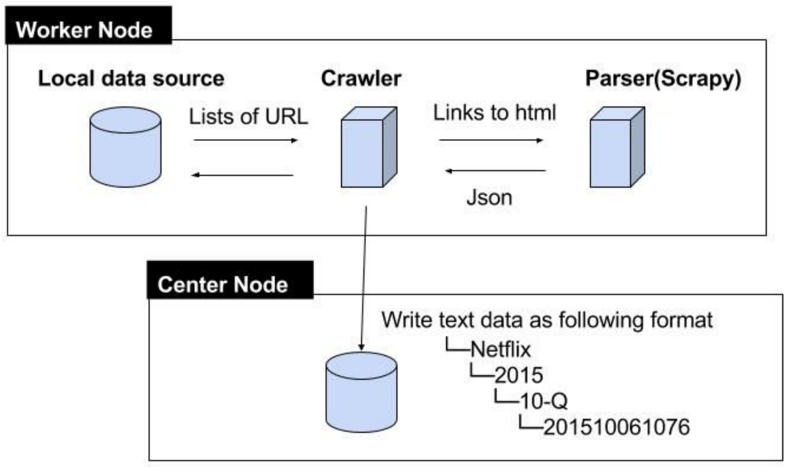
Crawling and parsing

Methodology

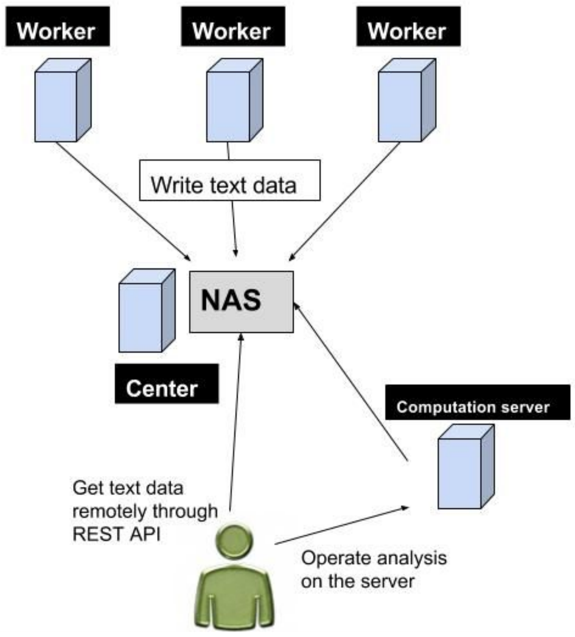
Here, steps for overall crawling and parsing operation will be explained. We have three steps for this operation.

Step 1: Creating lists of URL that are direct link to 10-Q and 10-K. These lists are gotten from crawler.idx at Edgar. At each year, and at each quarter, they have indexes for each company. This is a sample link ftp://ftp.sec.gov/edgar/full-index/2015/QTR1/crawler.idx. The link on crawler.idx is not direct link to each report, so we also parse the HTML to get the direct URL for the report. Our target is to analyze 10-K and 10-Q, so these indexes are parsed, and we will get lists of URL.
Step 2: Storing lists of URL gotten at STEP 1 at mongodb. We use these lists for monitoring crawling operation.
Step 3: Executing crawling based on data stored at STEP2. During crawling, when we need to parse HTML, we use a parser. The results of parsing will be stored as text files at central server so that team members can work on analyzing.

Here, system architecture is explained. We prepare "Worker Node" and "Center Node" for crawling and parsing operation. In addition, we have "Computation server" which has large memory and high spec CPU. Worker Node" is in charge of getting lists of target URL, parsing HTML, and saving parsed text data at "Center Node". Parsed text data is saved as the format shown at the following image.



We have multiple worker nodes so that we can parse large text data. At each worker node will be assigned for certain lists of URL and each node proceeds crawling and parsing individually. Center node has Network Attached Storage(NAS) and worker nodes access to the NAS environment and write text data there. Computation server also reads text data from central server through NAS. REST API for getting text data is also prepared for analysts.


Worker
Worker
Worker
Write text data
NAS
Center
Computation server
Get text data remotely through REST API
Operate analysis on the server

We also have re-crawling and parsing functionality to expand the target years of reports. Given CIK, we let analyst to crawl and parse data again. Here are steps for re-crawling and parsing.

Step 1: Analyst inputs CIK. The system checks whether there are any missing reports for the CIK internally.
Step 2: For given CIK, we use following link to gather information for each report.
https://www.sec.gov/cgi-bin/browse-edgar?action=getcompany&CIK=[CIK]&type=[report type]...
For example, when we will get Amazon(CIK=1018724), we call following URL.
https://www.sec.gov/cgi-bin/browse-edgar?action=getcompany&CIK=1018724&type=10-K
Then, we will parse data gotten from the above URL, and get the direct URL to each report.
Step 3: Parse the each report using scraper, and output the text data at Central server.

Scraper Design

Developing a web scraper that collects and organizes the data is a challenging task, as there is not a standardized HTML format across companies and no *id* or special *classes* are used. On our first attempt we tried to use the documents' *Table of Contents* as reference, taking the hyperlinks (@hrefs) as anchor points that define the beginning and ending of each section. However after revising several other 10-K and 10-Qs we noticed that not all the reports contain this section, so the idea was discarded.

On a second attempt we search directly for the section names within the document (i.e. “Item 1A.”). Once we got them we try to disambiguate among the results to identify which of the references is to the actual beginning of the related section. For these a number of custom rules were applied, to make sure the selection logic works on the largest number of documents. Unfortunately it is not possible to derive a ruleset that behaves as desired on all reports, as sometimes documents from different companies have opposing formats for the same anchor. For example:

- Company A defines the “Item X” header as a ‘font’ element with style ‘font-weight:bold’. All other mentions of “Item X” are not headers.
- Company B defines the “Item X” header as a ‘font’ element with no style, while other mentions of “Item X” with style ‘font-weight:bold’ are not headers.

To account for this, the scraper performs a check when parsing documents and raises an exception if any of the reports could not be processed. After this one manually needs to check the formatting of the affected document, and incorporate a modified rule to handle it if possible. Additionally, only paragraphs with more than 100 characters are extracted. This is to skip all elements that belong to tables, such as financial reports, as well as junk elements that appear throughout the text and provide no information (symbols, horizontal lines, links to the table of contents, etc.).

System information on Google Cloud

- Worker Node: 2CPU, 6GB memory, 100GB SSD disk, 5 instances
    - Python(including crawler and scraper), MongoDB
- Central Node: 4CPU, 8GB memory, 1200GB SSD disk, 1 instance
    - Network Attached Storage(NAS) server
- Computation Node: 16CPU, 60GB memory, 100GB SSD disk, 1 instance
    - Python(including Jupyter Ipython Notebook)
    - Web server/Application server for REST API

Parse and crawl result.

There are 9855 companies that we crawled and parsed. Each company may contain 10-K and 10-Q reports from 2007 to 2015, and the total reports would be over 180,000 reports. To parse each report, we needed around several minutes, although it varies on the content of text. To crawl and parse 6000 reports, we needed 1 hour for a single worker node. The crawling and parsing could be done around 1 day.

Data Processing

Before doing any further analysis, we performed the following data preprocessing to better fit our models.

- Remove Special Characters and Items

We removed the following terms using regular expression:

1. Numbering items using parenthesis like (a), (1)
2. Punctuations

3. Non-ASCII characters
4. English stop words
5. Special stop words
    - Generating Special Stop Words

10K and 10Q documents are used for government regulation on companies' financial performances. For this reason, they use a very special vocabulary related to financial terminologies. Therefore, besides normal English stop words, we also included the list of most frequent words in those documents and include them as stop words as well.

- Stemming

To prevent multiple forms of the same word, we also stem the words. We use the WordNet lemmatizer and stem each word as a noun and a verb and set the shorter one as the correctly stemmed word.

3.The Modeling Approach

Lexical Distance Measures

In the present section we evaluate the effectiveness of the following document representations and distance measures:

1. We represent each document as a binary vector, where a 1 indicates if a given word is present or not on the document. Then used the Jaccard distance to compute the dissimilarity between two documents. The jaccard distance measures the dissimilarity between two sets A and B:

   $$\frac{|A \cup B| - |A \cap B|}{|A \cup B|}$$

   On the graphs below, this distance is represented by the yellow bar (Jaccard Distance).

2. Same as point 1. but we first remove words that are present in all the documents of a given company. Hence, when we evaluated the forms belonging to Ford, words such as "Ford", "cars" and "SUV" were removed before computing the Jaccard distance. On the graphs below, this is represented by the green bar.

3. We used the TF-IDF representation of documents and then used the cosine distance. First, the TF-IDF, term frequency inverse document frequency is a numerical representation of how important is a given word to a document in a collection of documents. For term *t* in document *d,* we assign a weight in the following way:

   $tf - idf_{t,d} = tf_{t,d} * idf_t$

   where,
   $tf_{t,d} =$ $(Number\ of\ times\ term\ t\ appears\ in\ document\ d)\ /\ (Total\ number\ of\ terms\ in\ document\ d)$
   and
   $idf_{t,d} =$ $log\_e(Total\ number\ of\ documents\ /\ Number\ of\ documents\ with\ term\ t\ in\ it)$

   Secondly, the cosine distance measure the cosine of the angle between two vectors. It is a very popular measure used in text mining and natural language processing when comparing two documents. For a pair of vectors A and B, the cosine distance is defined as:

   $$1 - \frac{A.B}{||A||.||B||}$$

On the graphs below, it is represented by the blue bar.

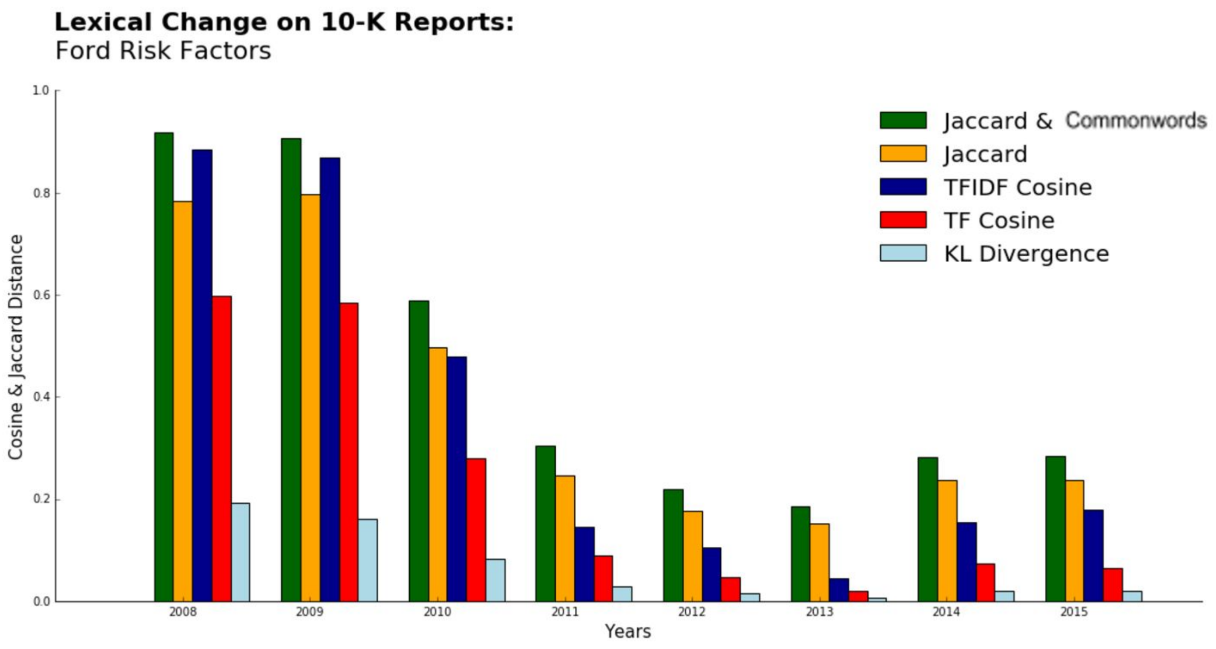
4. We used the TF, term frequency, representation of documents and then used the cosine distance. The TF representation is simply the count of each word in each document. This is represented by the red bar in the graphs below.

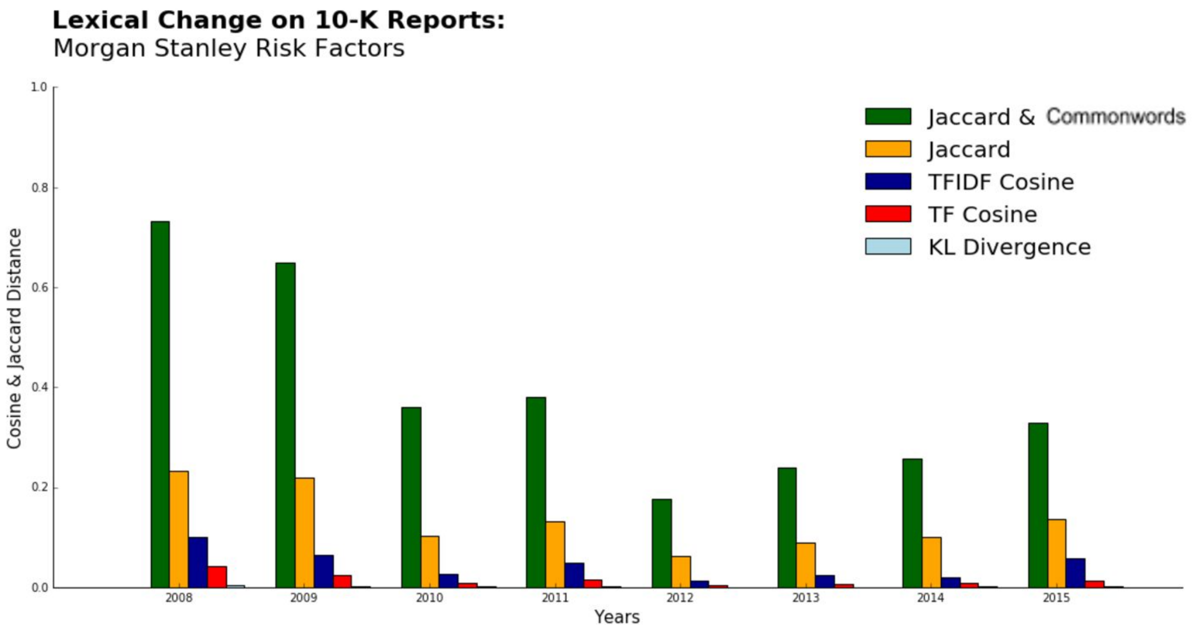
5. We represent each document as a vector, where each element of the vector represents the probability of a given word (count of the word/ total number of words). Hence, we are representing each document as a probability distribution over the words of the vocabulary . We then used the KL divergence or relative entropy, to measure the difference between two documents. The KL divergence between a distribution A and B is computed in the following way:

$$D_{KL}(A\|B) = \sum_i A(i)\, log\frac{A(i)}{B(i)}$$

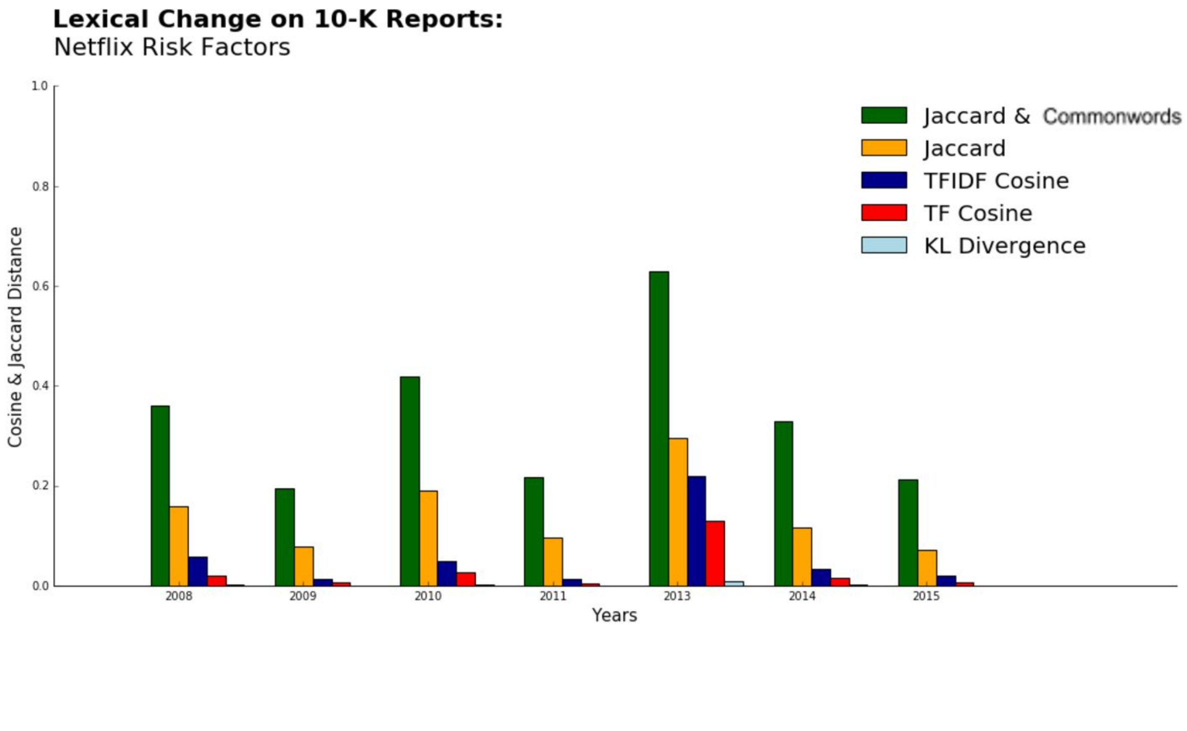
In the graphs below, this is represented by the light blue bar.

Our goal was to evaluate the behaviour of these measures and how effective they are representing lexical change. To perform our analysis, we looked at the risk section of several companies from different sectors between several years. Below we present some of our results:





Lexical Change on 10-K Reports:
Morgan Stanley Risk Factors
Jaccard & Commonwords
Jaccard
TFIDF Cosine
TF Cosine
KL Divergence
Cosine & Jaccard Distance
Years
2008
2009
2010
2011
2012
2013
2014
2015



Lexical Change on 10-K Reports:
Netflix Risk Factors
Jaccard & Commonwords
Jaccard
TFIDF Cosine
TF Cosine
KL Divergence
Cosine & Jaccard Distance
Years
2008
2009
2010
2011
2013
2014
2015

From the three graphs above we observe the following:

1. KL divergence is almost insignificant in most of the cases.
2. Jaccard distance when removing common words among documents is much higher than jaccard including all words and is high even when the rest of the distance measures are low (i.e. Morgan Stanley graph). Hence, it does not seem very reliable.
3. All distance measures are quite correlated and present similar behaviour. This is actually a good result because all of the measures seem to be useful and indicate the same change.

Given our above analysis, we consider appropriate to use the Jaccard (not excluding normal words) and the TF-IDF representation with cosine distance. The remaining distance measures show a similar behaviour, but usually have a too small or large value.

Topic Modeling

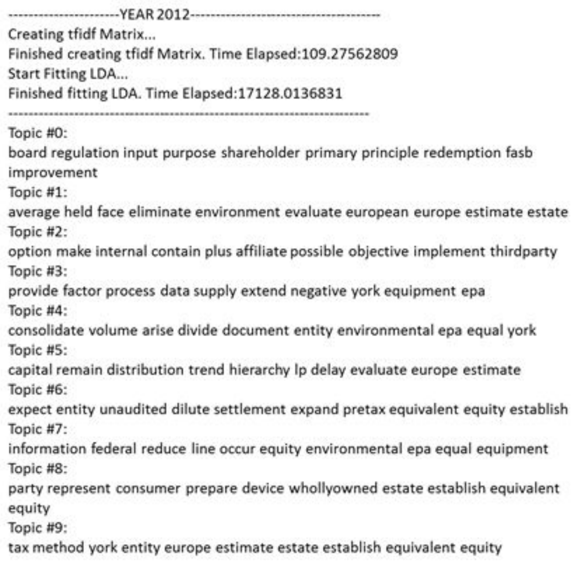
In this section, we use Latent Dirichlet allocation (LDA) to model the topic changes of company filings. A major advantage of LDA is that it is an unsupervised learning method, so no further labels are needed cluster the topics. The LDA outputs the given number of topics and the corresponding probability distribution of words in each topic. Also, one can estimate posterior distribution of topics for each document given the output matrix. The distribution of topics can thus be used to calculate the similarities between different filings.

At the current stage, we randomly select a sample of 1000 companies and use all their filings between 2012 and 2015 as training data set and the number of topics is set to 100 so far. The modeling is divided into two parts. In the first part, we cluster the filings by year and get 100 topics each year from 2012 to 2015. Compare sets of topics to see if there are significant shift of topics from year to year. In the second part, we pooled all the training filings together and use the trained model to compute cosine similarity between documents.

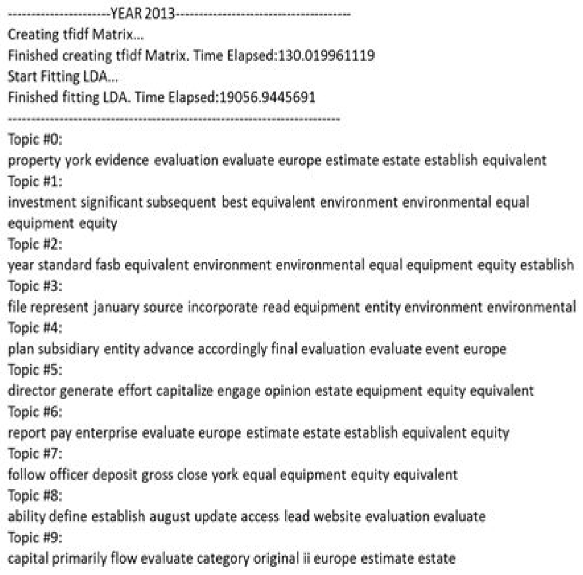
In the first part, we find that most of the topics overlap by years, while in each year there are still some "unique" topics of which the keywords cannot be found in other years. One reason that lots of topics share common keywords is due to the repetitive occurrence of financial and legal terms used across filings. For example, the keyword set [evaluate europe estimate estate] occurs in multiple topics in each year (e.g. 7 times in 2012 and 27 times in 2013). This could be caused by the risk sections where companies are obliged to discuss the overseas risks in each year's filing.

In the second part, we trained the model based on a sample of 1000 companies 10-K filings from 2012 to 2015. The model obtained can be used to transform any additional filings, whether section-level or document-level, and thus compute the probability matrix and cosine similarity scores. The scores are incorporated into our final output in addition to the lexical changes discussed in the previous sections.

The major drawdown for LDA modeling is that it is computationally intensive. It takes about 5 hours to train a tokenized data set of 1000 companies, given parameters set to 100 topics and 10 words each topic.

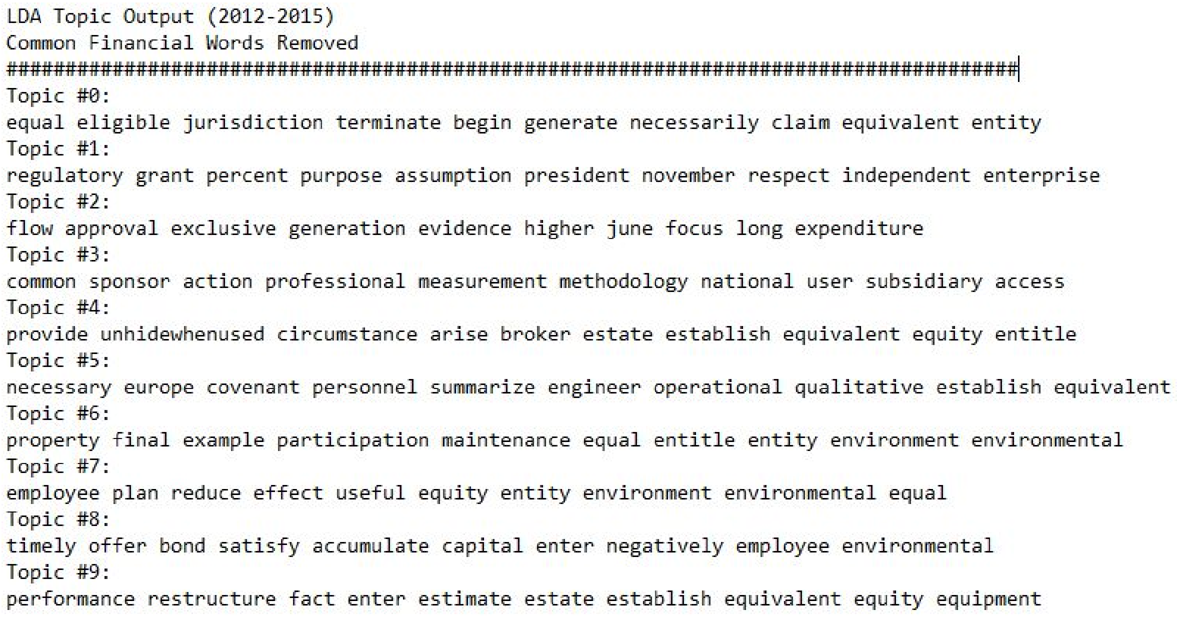


```
----------------------YEAR 2012--------------------------------------
Creating tfidf Matrix...
Finished creating tfidf Matrix. Time Elapsed:109.27562809
Start Fitting LDA...
Finished fitting LDA. Time Elapsed:17128.0136831
-----------------------------------------------------------------------
Topic #0:
board regulation input purpose shareholder primary principle redemption fasb
improvement
Topic #1:
average held face eliminate environment evaluate european europe estimate estate
Topic #2:
option make internal contain plus affiliate possible objective implement thirdparty
Topic #3:
provide factor process data supply extend negative york equipment epa
Topic #4:
consolidate volume arise divide document entity environmental epa equal york
Topic #5:
capital remain distribution trend hierarchy lp delay evaluate europe estimate
Topic #6:
expect entity unaudited dilute settlement expand pretax equivalent equity establish
Topic #7:
information federal reduce line occur equity environmental epa equal equipment
Topic #8:
party represent consumer prepare device whollyowned estate establish equivalent
equity
Topic #9:
tax method york entity europe estimate estate establish equivalent equity
```



```
---------------------YEAR 2013------------------------------------
Creating tfidf Matrix...
Finished creating tfidf Matrix. Time Elapsed:130.019961119
Start Fitting LDA...
Finished fitting LDA. Time Elapsed:19056.9445691
----------------------------------------------------------------------
Topic #0:
property york evidence evaluation evaluate europe estimate estate establish equivalent
Topic #1:
investment significant subsequent best equivalent environment environmental equal
equipment equity
Topic #2:
year standard fasb equivalent environment environmental equal equipment equity establish
Topic #3:
file represent january source incorporate read equipment entity environment environmental
Topic #4:
plan subsidiary entity advance accordingly final evaluation evaluate event europe
Topic #5:
director generate effort capitalize engage opinion estate equipment equity equivalent
Topic #6:
report pay enterprise evaluate europe estimate estate establish equivalent equity
Topic #7:
follow officer deposit gross close york equal equipment equity equivalent
Topic #8:
ability define establish august update access lead website evaluation evaluate
Topic #9:
capital primarily flow evaluate category original ii europe estimate estate
```

LDA Trial 1: A snippet of LDA topic output by each year (10 out of 100) based on filings from 2012-2015



```
LDA Topic Output (2012-2015)
Common Financial Words Removed
#####################################################################################
Topic #0:
equal eligible jurisdiction terminate begin generate necessarily claim equivalent entity
Topic #1:
regulatory grant percent purpose assumption president november respect independent enterprise
Topic #2:
flow approval exclusive generation evidence higher june focus long expenditure
Topic #3:
common sponsor action professional measurement methodology national user subsidiary access
Topic #4:
provide unhidewhenused circumstance arise broker estate establish equivalent equity entitle
Topic #5:
necessary europe covenant personnel summarize engineer operational qualitative establish equivalent
Topic #6:
property final example participation maintenance equal entitle entity environment environmental
Topic #7:
employee plan reduce effect useful equity entity environment environmental equal
Topic #8:
timely offer bond satisfy accumulate capital enter negatively employee environmental
Topic #9:
performance restructure fact enter estimate estate establish equivalent equity equipment
```

LDA Trial 2: A snippet of LDA topic output (10 out of 100) based on all filings from 2012-2015, common financial vocabulary removed

Word2Vec Similarity

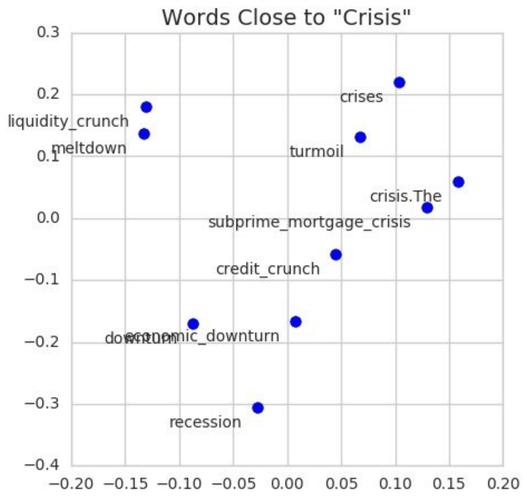
As a final measure of distance, we compare how semantically similar two documents are. To do this we will use their word embeddings representation, which corresponds to vectors in a *k*-dimensional vector space. The vectors are generated in a probabilistic model, where we try to maximize:

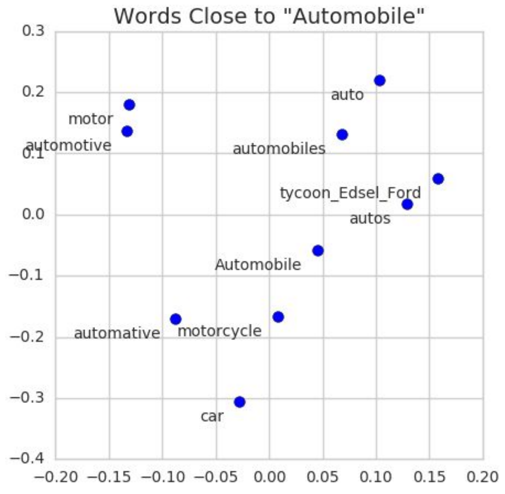
$$P(w_i \mid h) = \frac{\exp\{w_t \cdot h\}}{\sum_{j=1}^{n} \exp\{w_j \cdot h\}}$$

That is, the probability of the word (w_i) given its context (h), where context is defined as the set of words that commonly occur around it. For example, given the context '*I have a ____ that barks and bites*', we want to find the vector representation of '*dog*' and of the words in the context that maximizes the probability of seeing this complete sentence, as opposed to more uncommon alternatives like '*I have a pertaining that barks and bites'.*

Training a model that learns these word representations is an expensive task that requires a significantly large input corpus, so for practical purposes we have decided to use a pre-trained model from Google. This model covers a vocabulary of 3 million words and was trained on roughly 100 billion words from a Google News dataset, with a vector length of 300 features.

To understand better how these vector representations capture semantic meaning, let's look at a couple of examples. For visualization purposes the words embedding have been collapsed to two dimensions using TSNE.

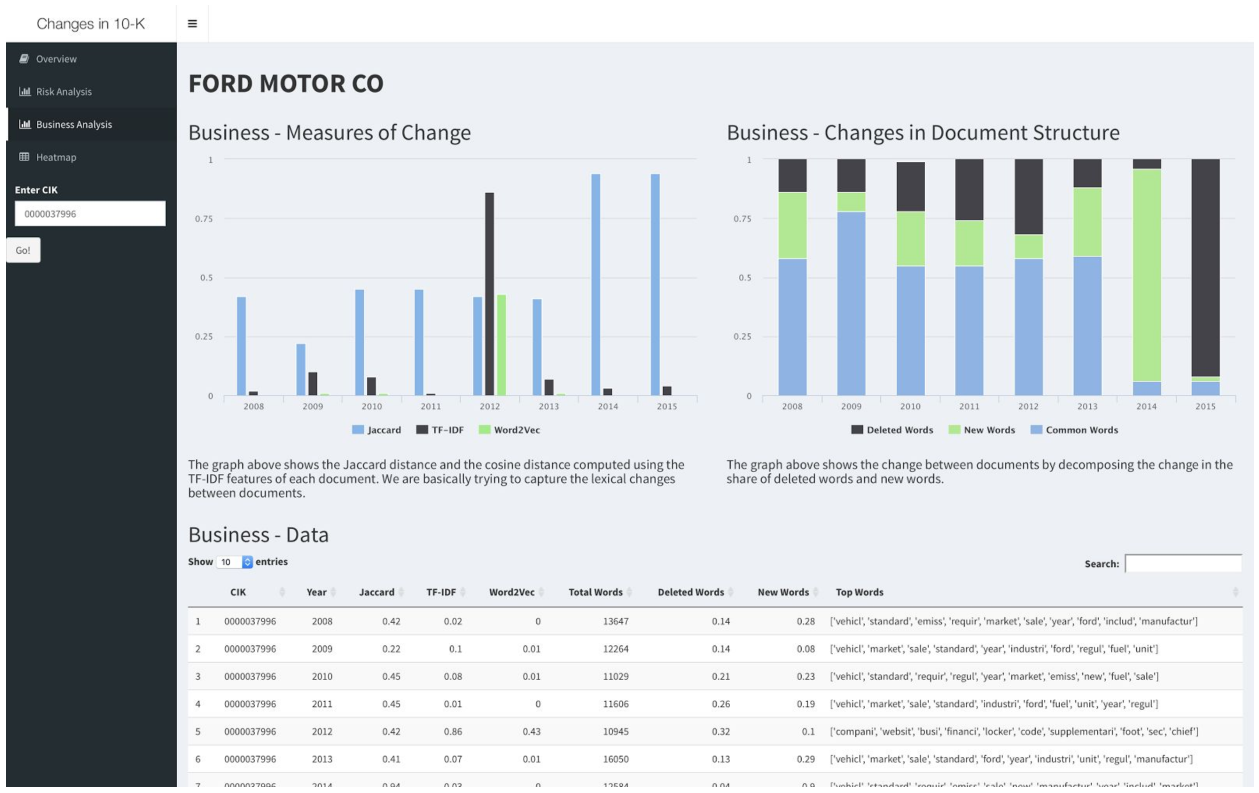




The charts above imply that *recession*, *turmoil* and *crisis* have a similar vector representation, while *car, automotive* and *Ford* are also similar to each other. For our purposes, we want to compare similarity across documents, so we can calculate the mean vector of each of them, and then compute the cosine similarity. After some experimentation we have confirmed that this similarity measure produces similar results (at least in terms of ranking) to the lexical measures, indicating that they are consistent among each other.

4.Conclusions and Future Work

As a result of this work we have created a Shiny Application (link) where users can track a number of companies, like those making up his portfolio, to track different metrics about the change in their 10-K forms. The user can either explore the data on a company level, or get a snapshot of all of them via a heatmap visualization.. Below we show a screencapture of the tool.



The graph above shows the Jaccard distance and the cosine distance computed using the TF-IDF features of each document. We are basically trying to capture the lexical changes between documents.

The graph above shows the change between documents by decomposing the change in the share of deleted words and new words.

Business - Data

Show 10 entries

Search:

	CIK	Year	Jaccard	TF-IDF	Word2Vec	Total Words	Deleted Words	New Words	Top Words
1	0000037996	2008	0.42	0.02	0	13647	0.14	0.28	['vehicl', 'standard', 'emiss', 'requir', 'market', 'sale', 'year', 'ford', 'includ', 'manufactur']
2	0000037996	2009	0.22	0.1	0.01	12264	0.14	0.08	['vehicl', 'market', 'sale', 'standard', 'year', 'industri', 'ford', 'regul', 'fuel', 'unit']
3	0000037996	2010	0.45	0.08	0.01	11029	0.21	0.23	['vehicl', 'standard', 'requir', 'regul', 'year', 'market', 'emiss', 'new', 'fuel', 'sale']
4	0000037996	2011	0.45	0.01	0	11606	0.26	0.19	['vehicl', 'market', 'sale', 'standard', 'industri', 'ford', 'fuel', 'unit', 'year', 'regul']
5	0000037996	2012	0.42	0.86	0.43	10945	0.32	0.1	['compani', 'websit', 'busi', 'financi', 'locker', 'code', 'supplementari', 'foot', 'sec', 'chief']
6	0000037996	2013	0.41	0.07	0.01	16050	0.13	0.29	['vehicl', 'market', 'sale', 'standard', 'ford', 'year', 'industri', 'unit', 'regul', 'manufactur']
7	0000037996	2014	0.94	0.03	0	12584	0.04	0.9	['vehicl', 'standard', 'requir', 'emiss', 'sale', 'new', 'manufactur', 'year', 'includ', 'market']

The first graphs contains three distance measures: cosine using TF-IDF, Jaccard and cosine using Word2Vec. We compute the difference between documents of consecutive years using these three measures. The second graph on the left contains the change in terms of the composition of words. Each bar contains three colors representing the share of common, new and deleted unique words. Hence the user can better understand how the change is composed, if it's because words were deleted or added. Finally, we also provide a summary table which contains the same information displayed on the graphs, but numerically, and also the number of words by document and the top ten words in terms of TF-IDF (basically these are the most relevant words).

Additionally, we are working on adding a second tab which displays the same information as the summary table present above, but for all companies. Then, the user will be able to first filter by year (to see most recent form changes only) and then sort the results by the amount of change to prioritize which documents to view.

Our methodology can discover lexical and topic changes and assign the importance as well. However there are still several things that can be done to make this project better.

Lexical Changes across Sections

Our model can detect the year to year sectional lexical changes but to combine all changes across different sections to get a final score for changes, we may need some better way to assign weights to each section.

Integration of Lexical and Topic Changes

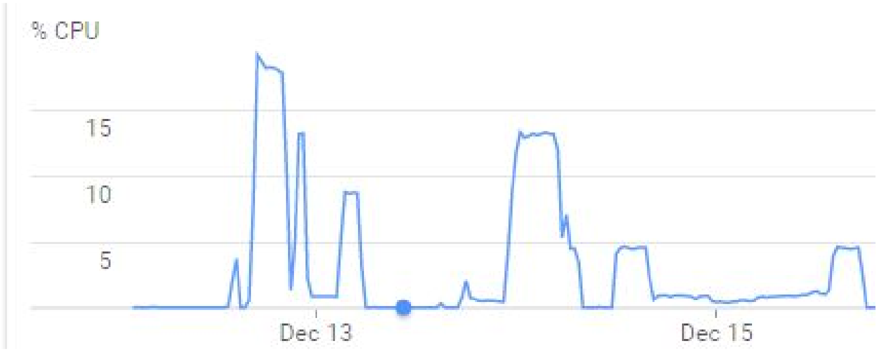
We are currently measuring lexical and topic changes separately and use a weighted average to get the final score. However, the lexical and topic changes are correlated since in most cases the change in topics will mostly result in a lexical change, which indicates that an weighted average may double count the changes.

More Granular Topic Model

We are fitting the topic model on all corpora of 10K and 10Q documents, however the companies from different industries may have different stop words and other unique features. In order to be more sensitive and accurate to the changes, we believe fitting topic models according to industries should be able to generate better result. Even though the companies are not labelled with an industry from the SEC filing system, given that most of the companies submitting 10K and 10Q files are public traded companies, we can consult Yahoo finance or Google Finance for the specific industries of a given CIK number and company name.

Modifying Computation Environment and Methodology

We tried to apply LDA for all companies, however, we encountered high CPU usage while we executed this operation. Below is the CPU utilization rate of our computation server. The system had 64GB memory and 16 CPU



As we can see above, %CPU spiked up to 16. At this time, we could not even log in to the server. We checked the states of processes using TOP or other commands, and we found that python processes consumed more than 100GB memory and used over 10 CPU constantly, and the system could not do the computation. We used Python (Jupyter ipython notebook with remote access), but it seems that it cannot do this operation for the current computation environment and methodology. As a future work, we should think of using other environment like spark which may be efficient for dealing with massive data. In addition, working on improving runtime for the computation may be useful.

5.Citations

[1] Butler, Matthew, and Vlado Kešelj. "Financial forecasting using character n-gram analysis and readability scores of annual reports." *Canadian Conference on Artificial Intelligence*. Springer Berlin Heidelberg, 2009.

[2] Kogan, Shimon, et al. "Predicting risk from financial reports with regression." *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*. Association for Computational Linguistics, 2009.

[3] Ahmad, Naveed, and Aram Zinzalian. "Predicting Stock Volatility from Quarterly Earnings Calls and Transcript Summaries using Text Regression." *CS224N Final Report* (2010).

[4] Le, Quoc V., and Tomas Mikolov. "Distributed Representations of Sentences and Documents." ICML. Vol. 14. 2014.