2017

Realizing EDGAR: eliminating information asymmetries through artificial intelligence analysis of SEC filings

Ryan H. Giarusso
University of Northern Iowa

REALIZING EDGAR:

ELIMINATING INFORMATION ASYMMETRIES THROUGH ARTIFICIAL INTELLIGENCE ANALYSIS OF SEC FILINGS

A Thesis Submitted

in Partial Fulfillment

of the Requirements for the Designation

University Honors

Ryan H. Giarusso

University of Northern Iowa

May 1, 2017

This Study by: Ryan H. Giarusso

Entitled: Realizing Edgar: Eliminating Information Asymmetries through Artificial Intelligence Analysis of SEC Filings

has been approved as meeting the thesis or project requirement for the Designation

University Honors with Distinction or University Honors (select appropriate designation)

_____ _____

Date Dr. Paul Gray, Honors Thesis Advisor

_____ _____

Date Dr. Jessica Moon, Director, University Honors Program

Table of Contents

Problem and Purpose

The U.S. Securities and Exchange Commission (SEC) maintains a publicly-accessible database of all required filings of all publicly traded companies. Known as EDGAR (Electronic Data Gathering, Analysis, and Retrieval), this database contains documents ranging from annual reports of major companies to personal disclosures of senior managers. However, the common user and particularly the retail investor are overwhelmed by the deluge of information, not empowered. EDGAR as it currently functions entrenches the information asymmetry between these retail investors and the large financial institutions with which they often trade. With substantial research staffs and budgets coupled to an industry standard of "playing both sides" of a transaction, these investors "in the know" lead price fluctuations while others must follow.[1,2]

In general, this thesis applies recent technological advancements to the development of software tools that will derive valuable insights from EDGAR documents in an efficient time period. While numerous such commercial products currently exist, all come with significant price tags and many still rely on significant human involvement in deriving such insights. Recent years, however, have seen an explosion in the fields of Machine Learning (ML) and Natural Language Processing (NLP), which show promise in automating many of these functions with greater efficiency. ML aims to develop software which learns parameters from large datasets as opposed to traditional software which merely applies a programmer's logic. NLP aims to read, understand, and generate language naturally, an area where recent ML advancements have proven particularly adept.

---

[1] In securities markets, the "sell-side" are those that issue securities, whether this be debt (bonds) or equity (shares of stock). The "buy-side" are investors looking to buy these securities. A large financial institution typically has divisions on both sides and, while they *technically* do not collude, they do in practice.

[2] Tetlock, 2014, "Information Transmission in Finance."

Specifically, this thesis serves as an exploratory study in applying recent advancements in ML and NLP to the vast range of documents contained in the EDGAR database. While algorithms will likely never replace the hordes of research analysts that now saturate securities markets nor the advantages that accrue to large and diverse trading desks, they do hold the potential to provide small yet significant insights at little cost.

This study first examines methods for document acquisition from EDGAR with a focus on a baseline efficiency sufficient for the real-time trading needs of market participants. Next, it applies recent advancements in ML and NLP, specifically recurrent neural networks, to the task of standardizing financial statements across different filers. Finally, the conclusion contextualizes these findings in an environment of continued technological and commercial evolution.

Literature Review[3]

There exists very little literature specific to the focus of this study, which in part helps to justify its exploratory nature. This stems from three general factors. First, as the introduction would suggest, a system that could efficiently analyze financial reports would bestow a marked advantage on any trader—an advantage they would rightfully refuse to publicize. Second, many of the algorithms this study will employ or the digital infrastructure on which they depend are on the bleeding edge of technology. The first full version of software library on which a significant portion of this study depends became available only in February 2017 and one of the key methods employed was released publicly in March. Finally, large financial institutions have little incentive to lead change in this domain, because they currently enjoy the benefits of large human research staffs.

[3] This document is intended for all readers, including those without a background in securities markets or in machine learning. As such, the majority of the Literature Review section eschews detailed descriptions and the complicated terminology they necessitate. Technical readers may find this information in the Methodology section.

The remainder of this literature review contains four general sections. First, "Securities Filings and EDGAR" describes the various filings of interest to this study, their contents, and methods of accessing them. Second, "XBRL and Filing Content" describes a specific format of filing content—financial statements—and their required reporting language. Third, "Machine Learning and Natural Language Processing" describes various methods in general, notable applications outside the securities domain, and necessary modifications for the securities domain. Finally, "Automated EDGAR Analysis" describes previous attempts at the objectives of this study, primarily securities-specific machine learning applications.

Securities Filings and EDGAR

Publicly traded companies in the United States have significant financial reporting requirements, established through many different legislative acts over the past century. While these forms range from the critical to the mundane, this study will focus specifically around four types: the 4, 8-K, 10-Q, and 10-K.[4]

Form 4 – Disclosure of trading by a company insider or significant owner.

Form 8-K – Current report of events that may have a material impact on the financial performance of a company.

Form 10-Q – Quarterly report summarizing a company's financial performance.

Form 10-K – Annual report summarizing a company's financial performance.

This study will focus initially on current (as of Quarter 3, 2016) members of the S&P 500 and Russell 3000 Indices. While "filers" to the SEC may have a variety of meanings ranging from companies to mutual funds to individuals, it is synonymous with "publicly traded company" in this study. Filers prepare their documents internally, then upload them to the File Transfer Protocol

[4] "Index to Forms, EDGAR Filer Manual Vol. II," *U.S. Securities and Exchange Commission,* August, 2015.

(FTP) servers of EDGAR, from which the SEC copies them to web servers. Most investors choose to view and download documents from these web servers, but automated collection typically gathers unformatted text through FTP.

Attempts to analyze the usage and consequences of rapidly and freely available electronic filings diverge between a long-term and short-term focus. While the latter seeks to employ EDGAR as a tool to obtain and act on material information before other investors, the former views EDGAR as a historical collection useful for developing long-term strategies and largely disregards consideration of real-time document retrieval. One particularly vital product of this research was the public release of EDGAR server logs from February 14, 2003 to September 30, 2015, as well as processing methods that these papers employed.[5]

EDGAR as a historical long-term fundamental research tool has attracted attention in both accounting and finance literature. In a series of three papers, Drake, Roulstone, and Thornock examine the EDGAR server logs for collection patterns of both current and historical documents while also comparing these patterns to those of Google searches around earnings announcements.[6,7,8] They find that investors typically make few and infrequent requests for quarterly financial documents while a small number of investors—likely larger firms with established "big data" strategies—make many and frequent requests for a wide variety of forms.

In somewhat similar analysis, Loughran and McDonald observe similar behavior patterns while coming to a rather premature conclusion that investors largely ignore EDGAR information

---

[5] "EDGAR Log File Data Set," *U.S. Securities and Exchange Commission,* December 23, 2015.

[6] Drake, Roulstone, and Thornock, 2014, "Investor Information Demand: Evidence from Google Searches Around Earnings Announcements."

[7] Ibid., 2015, "The Determinants and Consequences of Information Acquisition via EDGAR."

[8] Ibid., 2015, "The Usefulness of Historical Accounting Reports."

for fundamental research.[9] The authors intentionally ignore traffic they arbitrarily label as automated but do provide exceptionally clear descriptions of their methods. This allows for useful modifications to their methods, namely a simple reversal to examine automated traffic relevant to short-term analysis. Furthermore, the authors admit that EDGAR logs fall short of a conclusive sample, as investors have a wide range of sources from which to gather information, even with the inclusion of Google search behavior. A complete analysis would require inclusion of free aggregator data, such as from Google Finance/Yahoo Finance/Wall Street Journal Finance, as well as proprietary aggregator data, such as Bloomberg, Street Smart Edge, or Interactive Brokers (all ™).

Contrary to these conclusions, Christensen, Heninger, and Stice find significant differences in price and analyst reactions to SEC filings before and after the widespread availability of EDGAR, suggesting that at least institutional investors may engage with the database in some way.[10] However, their observations could easily indicate a mere quickening of information flow in general rather than a broad and influential use of EDGAR information.

While the use of EDGAR information in long-term applications remains in question, there remains the potential use of EDGAR data in short-term strategies. The rise and prevalence of High Frequency Trading (HFT) demonstrates that what is essentially instantaneous to the human eye is appallingly slow to securities markets. An advantage of mere microseconds with material information can deliver impressive—even astonishing—returns. With this in mind, a number of researchers have explored the patterns of document requests and the delivery speed afforded to different requesters.

---

[9] Loughran and McDonald, 2016, "The Use of EDGAR Filings by Investors."

[10] Christensen, Heninger, and Stice, 2013, "Factors Associated with Price Reactions and Analysts' Forecast Revisions Around SEC Filings."

Two recent works highlight the modern realities of document retrieval, both focusing around an institutional advantage enjoyed by large firms. Jackson and Mitts observe considerable differences between posting times of documents on the EDGAR FTP servers and the SEC web servers, as well as significant but smaller lead times afforded institutional investors through the public dissemination service (PDS)—a commercial subscription allowing for the automated dissemination of EDGAR filings.[11] Rogers, Skinner, and Zechman critique this work, noting that mere appearance of a document on an FTP server does not allow for immediate retrieval by a potential trader.[12] More importantly, they describe an interview with SEC staff, who describe the process of document submission and seem to indicate that commercial PDS subscribers are intentionally afforded an information advantage over retail investors.

As part of the groundwork for future research, this thesis implemented open-source programs for the automated collection and storage of historical and current EDGAR filings.[13] Initial examination suggests FTP retrieval times could still obtain a reliable time advantage over web documents, contrary to Rogers, Skinner, and Zechman. If such methods paralleled or even exceeded the advantage afforded PDS subscribers, such programs could reduce a significant institutional advantage and improve market liquidity immediately after key announcements.[14]

Of significant importance to this thesis was a decision by the SEC to permanently shut down public FTP access, effective December 30, 2016.[15] Announced through a small note on the relevant

---

[11] Jackson and Mitts, 2014, "How the SEC Helps Speedy Traders."

[12] Rogers, Skinner, and Zechman, 2015, "Run EDGAR Run: SEC Dissemination in a High-Frequency World."

[13] The following works were referenced in the creation of the EDGAR data collection code, which is available at https://github.com/gioGats/EDGAR/crawling:

- García and Norli, 2012, "Crawling Edgar."
- Engelberg and Sankaraguruswamy, 2007, "How to Gather Data Using a Web Crawler."

[14] Tetlock, 2014, "Information Transmission in Finance."

[15] "Accessing EDGAR Data," *U.S. Securities and Exchange Commission,* March 17, 2017.

web page just five days prior, this decision came with no official explanation. Evaluating the specific implications of this infrastructure change lies outside the scope of this thesis, but general large-scale data collection and analysis is significantly more difficult, if not impossible. While other methods for large-scale historical data collection still exist, they likely violate the SEC's terms of service. A replacement feed for current documents, updated every 10 minutes, provides some of the previous functionality albeit with a delay that probably precludes any short-term trading strategy. Although the data collection portion of this thesis completed prior to this shutdown, many of the data collection methods in this study are no longer applicable. However, relevant historical data was successfully compressed and archived for future use.

XBRL and Filing Content

Beginning in 2009 the SEC required filings containing financial statements (primarily the 10-Q and 10-K) to report this information in eXtensible Business Reporting Language (XBRL).[16] This requirement aimed for transparency, because firms have many different names that may appear on statements for the same accounts. XBRL combined a term (account names) with a definition in the XBRL taxonomy, with the intent of some degree of standardization and a larger degree of transparency.

While laudable in intent, significant research indicates anything but the desired outcome. Financial ratio calculation, an important use of financial statement data, is no easier with XBRL, because tags lack a sufficient level of consistency.[17] Analysis of institutional behavior following widespread XBRL adoption notes little change at the institutional level. If institutions employ

[16] "Structured Disclosure at the SEC: History and Rulemaking," U.S. Securities and Exchange Commission. April 27, 2016.

[17] Williams, Wenger, and Elam, 2012, "Analysis of Actual Company Filings Using XBRL."

financial ratios in their strategies, they calculate these ratios independent of XBRL consistency.[18] Far afield from its objectives, some research suggests XBRL may have actually increased the complexity of financial statements, making such information even harder for retail investors to understand and apply.[19] Despite these negative impacts, numerous proposals exist to improve standardization of XBRL, and existing analysis of the influence of public financial news suggests such improvement could greatly benefit investors and overall market health.[20,21]

The mechanism for additions to the XBRL taxonomy is a key failure of the XBRL system. While companies may genuinely need new tags to capture nuance in their accounts, the SEC allows essentially unfettered additions to the taxonomy largely irrespective of necessity. Many of the above papers share a common theme bemoaning the complexity of the XBRL system, most notably due to this proliferation of essentially identical tags.

This degree of tag proliferation has shifted focus from forced standardization through regulatory changes to various forms of post-filing standardization. A number of market information aggregation sites—namely Yahoo Finance and Google Finance—deployed methods intended to add standardized financial statements to their offerings. These both employed proprietary methods similar to methods proposed in recent research, but were far from fully successful.[22, 23] Both commercial and academic attempts failed to produce a standardization scheme sufficient for practical use.

---

[18] Cho, Bhattacharya, and Kim, 2014, "XBRL Mandate and Access to Information."

[19] Hoitash and Hoitash, 2015, "Measuring Accounting Complexity with XBRL."

[20] Starr and Yount, 2013, "The State of SEC Reporting Using Structured Data."

[21] Tetlock, 2010, "Does Public Financial News Resolve Asymmetric Information?"

[22] O'Riain, Curry, and Harth, 2012, "XBRL and Open Data for Global Financial Ecosystems."

[23] Wang, et. al., 2013, "Semi-automatic Generation Model of Elements in XBRL Taxonomy."

Yahoo Finance suffered a notable failure when their methods failed to connect the account for "Technology and Design" in Amazon's financial statements with the conventional "Research and Development." For a number of quarters, Yahoo Finance reported $0 in Research and Development for Amazon, while the real figure was around $2bn/year and prefaced the rise of Amazon Web Services (now Amazon's most profitable division).[24]

Aside from standardization, XBRL presents a translation challenge in facilitating information exchange between international investors seeking data on American companies and international companies seeking American investors. By employing a greater attention to detail and nuance often required of legal document translation, attempts at XBRL translation have had greater success.[25]

The relative success of automated translation informs this study's intended approach of applying recent advancements in machine translation to the task of standardization, whereby each term is treated as if it were in a foreign language and the remaining XBRL terms are the target language. Each "foreign" term is then mapped to its nearest XBRL neighbor, and the link is preserved if the two are sufficiently close. This approach is discussed in greater detail in "Methodology."

Machine Learning (ML) and Natural Language Processing (NLP)

Recent years—and even months—have seen an explosion in methods for and applications of Machine Learning, or programs that "learn" in the sense that they make decisions based on data analysis rather than explicit program logic. Due in part to these advancements, Natural Language Processing, the semantic comprehension of text, is both faster and more reliable. Given its exploratory nature, this research focuses primarily on the most recent methods, namely recurrent

---

[24] Loughran and McDonald, 2016, "The Use of EDGAR Filings by Investors," 5-6.

[25] Thomas, et. al., 2014, "Semantically Assisted XBRL-Taxonomy Alignment Across Languages."

neural networks, convolutional neural networks, and other deep neural networks implemented through libraries that support parallel and graphics processing unit (GPU) processing.

A neural network is a machine learning algorithm modelled after the human brain. It contains a number of perceptrons—or simply nodes—organized into layers. In each layer, each perceptron takes a range of inputs from the preceding layer, weighted according to some parameters. An activation function then converts these weighted inputs to a single output, which is passed to the next layer. During training, a neural network varies weights to produce "correct" values. At the risk of a gross overgeneralization, the non-technical reader may think of a trivially simple neural network as a kind of multi-layer linear regression. Numerous variants of neural networks have proven tremendously powerful; Google has deployed deep neural networks in areas ranging from image captioning to translation to generating email replies.[26,27]

This research will depend heavily on a wide range of software libraries. Chief among these is TensorFlow, a software package released by Google in December, 2015. It allows for "large-scale machine learning on heterogeneous distributed systems" by providing a framework for defining a series of computations, which the software then optimizes for the hardware available.[28] By using TensorFlow, programs implemented in this research can be duplicated later on a wide range of hardware, including commercial deep learning clusters, thus allowing for reuse and improvement in later work.

TensorFlow allows for the creation of intricate neural networks which include many layers (deep) featuring various complications. This research will focus on two complications: recurrence and convolution. Recurrent Neural Networks (RNNs) consider an input within the context of

---

[26] Szegedy, et. al., 2014, "Going Deeper with Convolutions."

[27] Corrado, 2015, "Computer, Respond to this Email."

[28] Abadi, et. al., 2015, "TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems."

previous input, much like humans consider each word they read in the context of the previous word. Recent versions use the Long Short Term Memory (LSTM) model, in which special perceptrons randomly allow previous input to weigh on current input. LSTM models have proven particularly well suited to the comprehension and generation of text. Successful use in image captioning, machine translation, and even literary composition all demonstrate the tremendous potential of LSTM models.[29, 30, 31, 32]

Convolutional Neural Networks (CNNs) mimic the structure of the animal visual cortex by arranging neurons to respond to overlapping "tiles" in the visual field. This is the mechanism by which the human eye sees shapes, as opposed to a collection of individual pixels of color. CNNs have proven particularly effective in computer vision, but recent applications include NLP. These models use chunks of words in place of visual tiles, and have competitive performance in semantic parsing, search query retrieval, sentence modelling, classification, and prediction.[33, 34, 35, 36, 37] Recent research describes an effective use of CNNs at character-level machine translation, a significant leap from word-level machine translation of RNNs.[38]

Despite these advances, business-specific language presents a key issue for NLP within the securities domain. Many terms that have a certain meaning or sentiment within a general context

---

[29] Vinyals, et. al., 2015, "Show and Tell: A Neural Image Caption Generator."

[30] Karpathy and Li, 2015, "Deep Visual-Semantic Alignments for Generating Image Descriptions."

[31] Sutskever, et. al., 2014, "Sequence to Sequence Learning with Neural Networks."

[32] Karpathy, 2015, "The Unreasonable Effectiveness of Recurrent Neural Networks."

[33] Grefenstette, et. al., 2014, "A Deep Architecture for Semantic Parsing."

[34] Shen, et. al., 2014, "Learning Semantic Representations Using Convolutional Neural Networks for Web Search."

[35] Kalchbrenner, et. al., 2014, "A Convolutional Neural Network for Modelling Sentences,"

[36] Kim, 2014, "Convolutional Neural Networks for Sentence Classification."

[37] Collobert, et. al., 2011, "Natural Language Processing (almost) from Scratch."

[38] Lee, et. al., 2016, "Fully Character-Level Neural Machine Translation without Explicit Segmentation."

may have opposite sentiment or a very different meaning within a business context. As part of similar research, Loughran and McDonald produced a reference dictionary which redefines such words within a business context, greatly improving many simple NLP approaches.[39] While it is unclear how much domain issues will plague deep neural networks, the availability of a specialized dictionary will aid in initial training stages.

Automated EDGAR Analysis

While strong financial incentives preclude the public release of previous similar work, a number of papers attempt steps towards automated processing of EDGAR filings. Most pre-neural network NLP relied on the bag of words method, by which each word in a phrase carries a weight irrespective of position. These methods bear some resemblance to early ML, although they warrant little comparison to modern ML. While somewhat effective in a number of cases, these early NLP methods often fail to capture detailed nuance, a common occurrence in financial disclosures. However, limited attempts have had some degree of success, even in extrapolating NLP into market predictions.

Previous papers fall into three general categories: narrow textual analysis, broader feature analysis, and market-level analysis. The first of these use NLP to analyze a specific aspect of a document, such as management's sentiment or financial constraint identification. The second adds other features in addition to raw text, such as search patterns, filing timing or frequency, or media reaction. The third comes closest to broad and artificially intelligent processing, combining many features including raw text into a broader analysis aimed at least in part at market prediction. For brevity, relevant research in these areas is not directly cited, but listed by category in a "Works Consulted" section following "Works Cited."

---

[39] Loughran and McDonald, 2015, "The Use of Word Lists in Textual Analysis."

A key resource of note, however, is CS 229: Machine Learning, a course at Stanford University that requires its students to publish a semester project via the course website. This provides a wealth of undergraduate and graduate student research in applying various recent ML methods to a wide range of problems. Whereas scholarly literature on applications of recent ML to financial markets is sparse, the CS 229 website lists 30 student papers on the topic in the preceding two years.[40] Few of the student attempts had even a minor degree of success, but the papers do provide the results of a wide range of methods to avoid.

Research Questions

Given its exploratory nature, this thesis focuses first around general questions that seek to understand general possibilities and then on a specific question that serves primarily as a technology demonstrator.[i]

Document Acquisition: Can individual users obtain filings in a sufficiently timely manner such that their automated analysis can inform trading decisions?

Performance Considerations: Given the overarching objective of widely applicable usage, what reductions in data size and complexity are necessary for reasonable time performance on consumer-grade hardware? Do these reductions alter results to such an extent that it invalidates the methodology?

Financial Statement Standardization: Can an algorithm learn to standardize the XBRL taxonomy to then generalize financial statements for the automated calculation of financial ratios?

---

[40] Ng and Duchi, "CS229: Machine Learning," *Stanford University*, October 12, 2016.

Methodology

This thesis incorporated three general phases, each addressing one of the three research questions. Due to their general and exploratory nature, the first and second sections report their findings at the end of their sections in Methodology, reserving the entirety of Results for the third research question. First, "Document Acquisition" discusses methods for the mass collection of historical SEC filings through the EDGAR database, the parsing of such text data efficiently, and storing large data for regular access and for archive. Second, "Performance Considerations" discusses the hardware requirements of large datasets and various software tools for their efficient processing. The general findings of this investigation have also been summarized in a blog post to help improve others' attempts at building deep learning capable machines with consumer hardware, included in Appendix C. Finally, "Financial Statement Standardization" discusses the methods, technical foundations, and assumptions underpinning the implemented software package that maps custom XBRL tags to standard XBRL tags.

Document Acquisition

Three document acquisition strategies were used or received serious consideration throughout the course of this thesis: file transfer protocol (FTP), web-based (HTTP), and pre-parsed data. Alternatives to the first were considered after the shutdown of FTP servers on December 30, 2016 left previous work unviable. Regardless of the data acquisition strategies utilized for later portions of this research, there currently exists no method for rapid and programmatic access to a large batches of historical SEC filings.

At the beginning of this thesis, FTP was the standard for programmatic collection of historical and current SEC filings. The protocol enables the reliable and efficient transfer of files

across remote computers.[41] By opening this protocol to consumers, the SEC allowed programmatic access to filings without the need for cumbersome webpage formatting while also dividing traffic between FTP clients (typically programs) and HTTP clients (typically humans).

Despite an efficient protocol, a complex structure housed EDGAR FTP files.[42] A large directory contained a sub-directory for each central identification key (CIK), a unique number assigned to every filer including every individual and corporation required to file documents with the SEC. These CIK directories contained all relevant filings, some in multiple formats. Separate index files allowed for accessing specific files by detailing information about the filer, form number, time of filing, and an access link for every filing in a quarter. Available only as a compressed download, document collection scripts must first download the relevant quarterly index, sift through for desired documents, and then initiate an FTP transfer.

The programs implemented as part of this thesis for document acquisition include three key adaptations to optimize time and network performance. First, a deeper and narrower directory structure helped improve access performance. Because CIK numbers have nine digits, there are theoretically one billion subdirectories in the EDGAR FTP structure.[43] By dividing CIK numbers into three, three-digit blocks, each directory has a theoretical maximum of one thousand sub-directories and practically has far fewer. Current CIK numbers do not comprise even 1% of the available address space.

Second, limiting transfers to only desired documents ensured storage and network requirements remained manageable. EDGAR contains a vast range of filings, many of them

---

[41] Postel and Reynolds, "File Transfer Protocol," *ISI*, October, 1985.

[42] "Accessing EDGAR Data," *Securities and Exchange Commission,* March 17, 2017.

[43] This structure probably masks an implementation detail of the EDGAR infrastructure and thus does not negatively impact EDGAR's internal performance in the same way.

irrelevant to most users. As such, implemented programs include settings files that limit document acquisition based on CIK, form type, and date range. These files allow even non-programmers to quickly adjust their document acquisition based on research requirements.

Limiting document acquisition required the use of multiple "target" lists. Acquisition programs compare each row of an index file against user-defined values. If the row's values all match the user's, the program queues the file for download. Repeated scanning of potentially lengthy lists necessitated the third and final performance adaptation: self-balancing binary search (or simply AVL, after its inventors) trees.[44] In an AVL tree, each node has 0, 1, or 2 child nodes (a property of all binary trees) while also maintaining balanced depth. If a parent node has two children, the heights (or number of levels beneath the child) of those children differ by at most one. While a somewhat trivial implementation detail, AVL trees improved time performance by almost a factor of two.

While deployed, programs limited access times to SEC-defined non-business hours, during which FTP servers allowed for unlimited download requests. Outside of these hours, too many requests in a given time interval would earn the offending program a temporary ban. Repeated violations, though unconfirmed, would likely result in a permanent ban of some kind. These access windows made FTP ideal for the mass download of historical documents and for nightly downloads of a previous day's filings.

Current documents, however, require a narrower access window for effective use in modern markets. With a 10-minute update interval, an existing Rich Site Summary (RSS) feed provided an upper-bound for FTP performance.[45,46] If FTP could deliver access times under 10 minutes, it could

---

[44] Adelson-Velsky and Landis, 1962, "An algorithm for the organization of information."

[45] Libby, "RSS," *Netscape Communications,* July 10, 1999.

[46] "Structured Disclosure RSS Feeds," *Securities and Exchange Commission,* February 23, 2017.

deliver an information advantage to the user. A narrower time window, as hypothesized in the literature review, was investigated carefully given the potential implications of loss of access. Two methods emerged, but neither showed promise.

First, identifying new filings of any filer required downloading and decompressing a quarterly index file. With total time including time to update the index file, download time, decompression time, and search time (finding new entries and determining if they are within the defined collection net), this method would probably not outperform the RSS feed. Searching for new filings of a specific CIK, however, required only listing the contents of the CIK-specific directory and scanning that list for new or updated files. Such an approach could allow for an information advantage for a specific, highly-anticipated filing such as a quarterly earnings report. However, the speed of this method was limited to the frequency of directory listing. Because this operation is an FTP request, increasing this frequency to the speed required of modern trading often resulted in a temporary ban.

With the shutdown of EDGAR FTP, these methods fell into obsolescence. Index files, while still available, do not rival RSS performance. With directory structures now opaque to the user, programs cannot wait for an anticipated filing. Access URLs now depend on a filings' ascension number, a unique identifier given at the time of filing. Of greater consequence to this thesis, however, document acquisition programs lost their data source, necessitating a reexamination of sources and methods.

EDGAR still allows for programmatic access to filings through Hypertext Transfer Protocol (HTTP), the protocol on which much of the modern internet relies.[47] This approach is viable at least in theory. Index files still list access URLs, from which HTTP programs such as *wget* can access

---

[47] Fielding, et. al., "Hypertext Transfer Protocol," *ISI,* June, 1999.

and download historical documents.[48] At 10 minute intervals, such programs can download current documents from URLs provided via the RSS feed. While many of these files now come in a web format (.html), more complicated programs can still parse these files for relevant data in a timely manner.

However, two core limitations to this approach precluded serious investigation in this thesis and therefore the adaptation of FTP programs. First, the mass download of millions of documents falls clearly within the intended use of FTP, not HTTP. Placing such a burden on SEC servers often results in a temporary ban, as the SEC very openly prioritizes human users. Second, current document collection with HTTP is firmly limited to the speed of the RSS feed. For automated trading strategies, this 10-minute delay is simply not viable.

As part of the early 2017 upgrade to its historical data offerings that shuttered FTP, the SEC compiled all financial statement data dating to 2009 into quarterly archives. While the loss of FTP significantly reduced the applications of document acquisition programs, the availability of neatly structured, pre-parsed data proved a tremendous benefit for later attempts at financial statement standardization. This data makes processing financial statements far easier, but two limitations make it a poor replacement to FTP. Its quarterly update cycle leaves this data only valuable for historical research and largely useless for current documents or for trading strategies. Furthermore, the removal of text that accompanies such financial statements precludes broader text mining and analysis, a previously stated objective of this thesis.

With the loss of FTP access, much of the early work in document acquisition fell to the evolution of technology. Easily available and pre-formatted XBRL data made code revision or new document acquisition strategies immaterial to completing the initial research questions. HTTP

---

[48] Scrivano and Nikšić, "GNU Wget," *Free Software Foundation*, February 13, 2017.

presents new challenges, but if true market advantages exist through the collection and processing of historical EDGAR data, future research should certainly focus on this.

Performance Considerations

The average person has little comprehension of the vast complexity that underpins financial reporting. In 2016 alone, filers used 285,102 unique tags to describe their accounts. As such, analyzing a large number of similar financial statements to standardize some of this large number of tags presents serious performance concerns. Problems of such a scale usually remain in the limited purview of large technology firms and cloud providers. Yet for software developed in this thesis to remain widely accessible—a stated goal of the work—they must also remain within the capabilities of consumer grade hardware. Two factors dominated performance considerations: the parallel nature of many methods and the size and accessibility requirements of data storage.

Sequential computation has defined much of human and algorithmic history; when we do mathematics, we execute a series of steps. Parallel computation stands in marked and counterintuitive contrast; in such problems, thousands or millions of computations do not depend on the results of others and thus can occur simultaneously. A master program can spread a sufficiently parallel computation across a wide range of computation hardware, greatly increasing the overall computational performance. First commercialized and popularized through MapReduce, parallel computation underpins much of the modern revolution in Machine Learning.[49,50]

Like humans, programs default to sequential processing and require lengthy and deliberate consideration for effective parallelization. Programs implemented in this thesis make use of parallel computation wherever possible as the single largest step to ensure acceptable performance. This

---

[49] Dean and Ghemawat, 2008, "MapReduce."

[50] Brown, "Deep Learning with GPUs," *NVIDIA,* June, 2015.

required numerous supporting software libraries, detailed in Appendix B. Although parallelization can achieve suitable performance for end users, the experimentation required in this research necessitated hardware specialized for parallel computation.

Because end users rarely need worry about parallelization, most consumer hardware does not perform well on such tasks. Rather than rely on cloud computing to perform the bulk of calculations and violate the consumer-grade intent of this thesis, the author constructed a specialized computer for the relevant computation. While generally outside the scope of this thesis, a guide to constructing a computer for massively parallel computations with consumer grade hardware is included for interested readers in Appendix C.

Working with large and complex data posed significant storage requirements in both experimental and deployment environments. Storage considerations typically pit size requirements against access speed. Access speed was prioritized in the experimental environment, while size was prioritized in deployment. Generally, specialized hardware downloaded large raw datasets, parsed raw data for specifically relevant information, saved this information in a highly accessible format, and heavily compressed the remaining raw data.

Throughout the project, approximately 3 Terabytes of data were downloaded, collected, or processed into three datasets: historical filings (1.1TB), SEC server logs (1.5 TB), and XBRL data (100 GB). By employing complementary compression algorithms, Long Range Zip and ZPAQ, the first two compressed to just 100GB and 150GB respectively for archival storage.[51, 52] This extreme compression ratio came at the cost of time, with each compression lasting just over 48 hours. With

[51] Kolivas, "Long Range Zip," November 13, 2016.

[52] Mahoney, "ZPAQ," August 17, 2016.

the closure of EDGAR FTP, however, an archived copy of a significant portion of its dataset may play an important role in future research.

As for accessible data, the Hierarchical Data Format (HDF5) was selected for its efficient use of memory and a simple and forgiving interface.[53] While the size of data for computations is not a concern on experimentation hardware, few practical users of an XBRL mapping tool would appreciate a need to close all other applications prior to its use. HDF5, with efficiently designed programs, allows for a regular file system to support near memory speeds by anticipating data needs during iterative operations. This multi-tiered storage structure allowed maximal usage of parallel computation capabilities, combining to allow for a much wider range of experimentation than otherwise possible.

Financial Statement Standardization

As the primary experiment in this thesis, the algorithmic simplification of the XBRL taxonomy comprises the most substantive portion of the work. This section details an overview of the software implemented for this experiment and a cursory overview of the theory underpinning supporting methods. At a general level, the system must convert text to a mathematically meaningful representation, perform the actual mapping of custom XBRL terms to standard XBRL terms, and finally some form of simplification to keep the problem manageable. These three tasks require vector embedding, vector space mapping, and unsupervised clustering, respectively.

[53] "Hierarchical Data Format," *The HDF Group,* 2017.

While the text on a computer screen is a software abstraction of a series of mathematical operations performed on ones and zeros, these representations have little mathematical meaning. For words to have a mathematical meaning implies the feasibility of a kind of word math. For example, a proper representation should allow for the following equation:

KING – MAN + WOMAN = QUEEN

In plain English, we might say that we take away from the idea of "king" the idea of "man". To what remains, we add the idea of "woman" and arrive at the idea of "queen". In a mathematical sense, the system must convert the words to a numerical representation that, when the same mathematical operations are applied, produces a numerical representation sufficiently close to that generated by converting the right half of the equation. Generally, this process is called vector embedding.

In 2013, a team of Google researchers proposed a word level embedding model, word2vec that showed remarkable promise.[54] Subsequent research found that multi-layer long short-term memory (LSTM) cell neural networks could embed complete sequences by incorporating a degree of memory in computations.[55, 56] In essence, the network learns that the meaning of the current work depends in part on the previous words. While much of this research now focuses on sequence to sequence machine translation, these methods also allow for effective and efficient vector embedding in other applications.

Two serious concerns drove the design of the vector embedding approach for this system. First, to allow for a wider range of follow-on methods, embedding would be to a fixed length. Choosing this length, however, required a balance of performance and theoretical considerations. A

---

[54] Mikolov, et. al., 2013, "Distributed Representations of Words and Phrases and their Compositionality."

[55] Sutskever, et. al., 2014, "Sequence to Sequence Learning with Neural Networks."

[56] Wu, et. al., 2016, "Google's Neural Machine Translation System."

shorter vector may not sufficiently embed the full meaning of a sequence, but an overly long vector will slow computations behind the realm of practicality. Based on initial performance testing, vector lengths of 50, 100, 150, and 200 were selected for testing as maximum values that still provided adequate performance. A freely available library, seq2vec, provided a wrapper class over a TensorFlow implementation of an LSTM network, effectively duplicating the method currently employed in Google Neural Machine Translation.

Once embedded as fixed length vectors, a wide range of algorithms are available. However, data limitations significantly narrowed the range of available methods, with three noteworthy characteristics. First, XBRL data is unlabeled, meaning that it does not include the "right" answer (it would be frivolous to define a custom tag and then state that it merely duplicates another). This stands in stark contrast to many of the greatest recent advances in ML, which rely on labeled data for supervised learning, or simply a training procedure in which the program makes a guess, checks its answer, and adjusts itself accordingly to produce the correct answer next time.

The second most common training procedure, reinforcement learning, requires a reward function. This method uses repeated trial and error in a scenario to learn tendencies that maximize a specified goal. Yet similarly, no objective goal function could reward an algorithm for correctly mapping a tag. With the two most common methods unavailable, only the very frontier of ML remained: unsupervised learning.

In unsupervised learning, programs generally do not compute correct or incorrect answers, but instead identify generalized patterns or associations within a large dataset. While by far the most challenging learning approach, the fundamental objectives of this thesis required working with unlabeled data. To invest significant time and resources in correctly labelling data by hand amounts to little more than a high-tech take on existing investment research firms and poses an insurmountable challenge to the resources of a single researcher. This decision to work with

unlabeled data narrowed the range of available methods, forcing a focus on methods with a mathematical basis for results absent pre-ordained correct answers. This led to vector distances as the metric for mapping decisions.

In this design, a program takes a novel custom tag and embeds it into a vector space that already contains the embeddings of standard tags. It then maps the custom tag to the nearest standard tag, based on a distance metric. This potentially high dimensional space allows for numerous distance calculation methods. Yet because these computations will run many millions—if not billions—of times even the smallest inefficiencies can quickly compound. With this in mind, only distance functions with exceptionally efficient implementations were considered because low-level implementation improvements fell outside the scope of this research. Experiments tested the performance of two distance metrics, Euclidean and Cosine. In the former, the magnitude of a vector plays a key role; in the latter, all vectors are scaled to the unit vector and only direction is considered.

The considerable number of terms and the computational limits to the dimensionality of the vector space and to the problem set in general necessitated a certain degree of simplification. A number of dataset simplifications and unsupervised clustering were used to this end. First, while data dating to 2009 was available, only 2016 data was considered. The SEC updates its taxonomy every year with small tweaks, meaning a simple mapping of terms from year to year could produce an excellent 7-fold decrease in tag volume—while actually having little impact at all. Moreover, there are likely few significant changes in custom tags between years. Massaging accounts may be general practice, but inventive accounting is the post-Enron kiss of death.

Second, XBRL tags come with a number of data fields that describe their general nature, such as if the account is a credit or debt or if the value is numeric of not. To maintain the focus on NLP, tags were presorted into categories based on these objective properties. While this, to some

extent, does the algorithm's job, it makes little sense to make a difficult problem even more so. This effectively shrinks the problem from a single one of 285,000 tags, to a series of nine problems with between 600 and 70,000 tags.

Beyond these dataset simplifications, unsupervised clustering played an important role in shrinking the scope of required computations. The selected algorithm, KMeans, randomly selects k (a specified parameter) points at random as cluster centroids.[57] It then assigns each point in the dataset to the nearest cluster, and moves the centroids to the center of these new clusters. The process repeats until the cluster inertia, or the sum of the distances between all points and the nearest centroid, ceases to improve. This algorithm is prone to falling into local minima and producing a sub-optimal result, so each trial is run 10 times and the best result reported. Selecting the number of clusters implies some general structure of the data, so a wide range of k values were tested and inertia at each reported. Broader computations, however, were run at 1000 clusters only for performance concerns. This implies that 1000 tags can adequately convey the necessary variation in each category.

In addition to theoretical considerations, the nature of the real-world environment represented in XBRL plays a key role in understanding the limitations of the model. XBRL data may seem mundane, but the ideas it expresses can have serious implications in financial markets. In current market conditions, firms that miss quarterly earnings targets can expect significant losses in their stock price, sometimes exceeding 10% in a single day. Firms desperately try to make their earnings-per-share (EPS) appear as high as possible and few refuse to employ a certain level of maneuvering within their accounts to do so. Given such realities, XBRL data probably contains some degree of deception.

---

[57] Lloyd, 1982, "Least Squares Quantization in PCM's."

While entirely within the bounds of regulation, attempts to massage the balance sheet of a large firm probably motivate a large degree of the expansion of XBRL custom tags. To a large extent, such largely unnecessary expansion motivates this research. However, it also presents a unique challenge and limitation when compared to other NLP problems. Most NLP involves generally honest and forthcoming data. By applying NLP methods to potentially deceptive data, this thesis stretches the application of these methods to new domains both potentially expanding the field and delivering less than ideal results.

Finally, the narrow and specific nature of XBRL tags—a subset of corporate accounting, a subset of general accounting, and a subset of the business world in general—means the XBRL text that programs will encounter shares little with the general English text in large training databases used for most NLP problems. This will require algorithms to generalize to an entirely new set of information, including words that may mean something very different in an XBRL tag than they do in general English. These limitations will test the limits of sequence to vector embedding tools.

With these data and performance limitations in mind, a complete system for standardizing XBRL tags was designed, supporting libraries were selected, and experimental variables and their ranges were identified. As shown in *figure 1*, the system first sorts XBRL tags into categories. It then uses a sequence encoder to transform text into a fixed-length vector. To reduce the complexity of the problem, it is assumed that any one category requires no more than 1,000 standard tags or, if there are fewer than 1,000 tags in the initial data, 50% of the original number. To reduce standard tags to these benchmarks, a KMeans algorithm is used. Finally, each custom tag is tested against the standard clusters of its category, and assigned to the closest cluster. All but the last step can be processed in advance and saved, allowing distributed software to quickly process a mapping of a user-defined custom tag. A number of freely-available software libraries played important roles in

the implementation of this system. A full list is detailed in the appendices with details of their use and reference links.

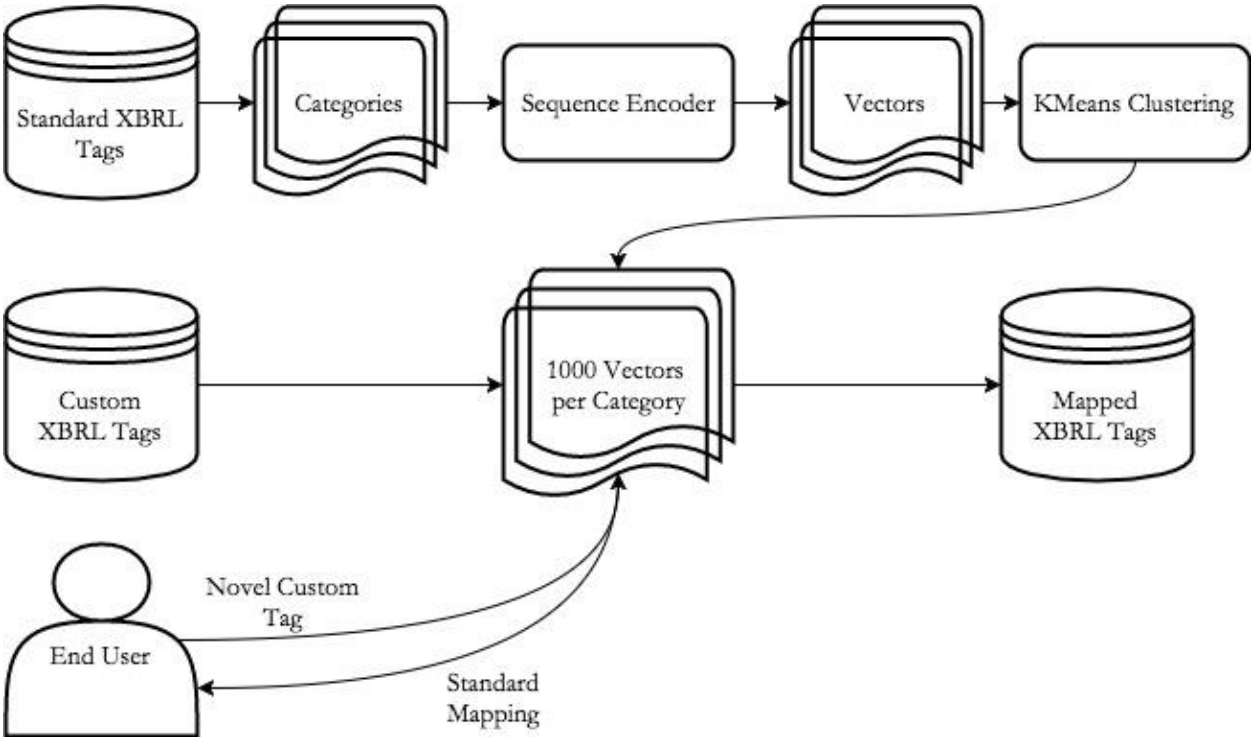


FIGURE 1 – SYSTEM OVERVIEW

Results

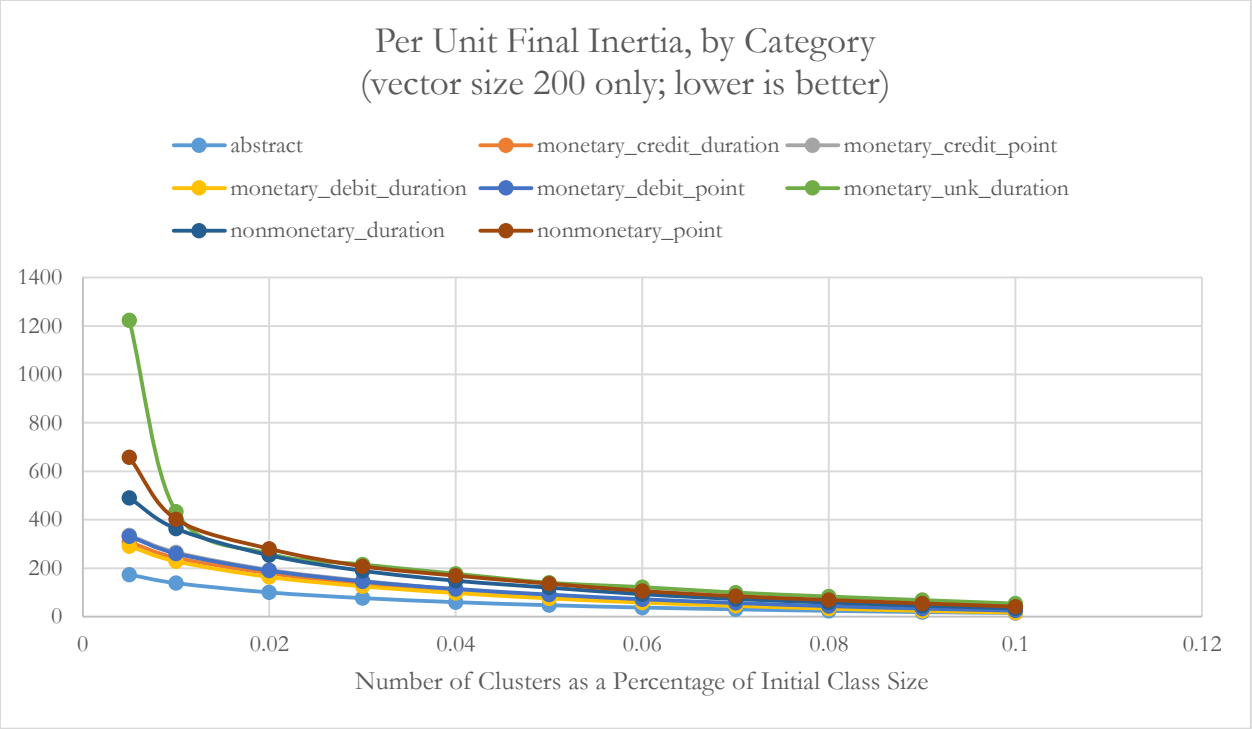
Reports from experiments performed with this data indicate a two-part story. In direct reference to the original research questions, the experimental model did not show sufficient performance with any mixture of values that would satisfy the information needs of a professional investor. However, several indications suggest the approach is valid. As is often the case in Machine Learning, the model concept is easy; the fine-tuning is perilous. This section first discusses experimental results that validate assumptions of broader system design and finishes with select best- and worst-case examples of actual term mapping.

Per Unit Final Inertia

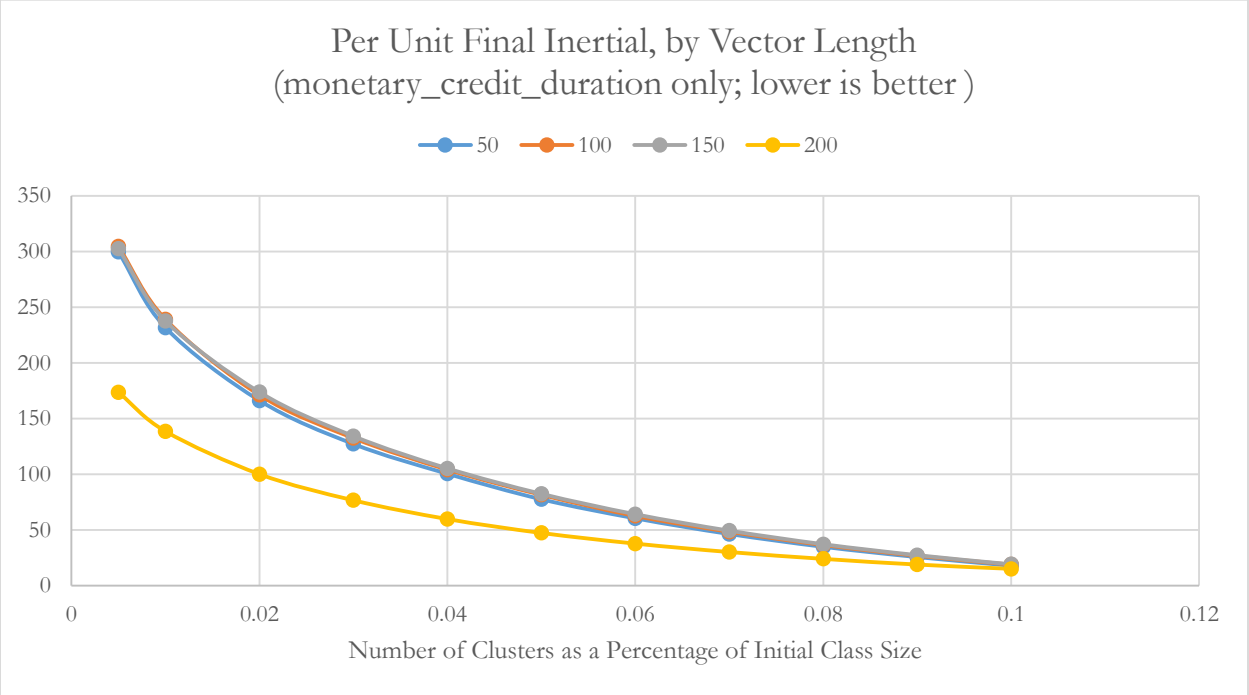
Clustering algorithms seek to minimize per unit final inertia, or the average distance from vectors to the center of their assigned cluster. To validate the use of clustering in reducing the set of standard terms, per unit final inertia is compared across various cluster sizes. Fewer clusters should almost always result in higher per unit final inertia. Yet if the data actually describes a set of various (but unknown) categories, per unit final inertia should increase at a relatively slow rate when number of clusters is more than actual categories and at a relatively high rate when number of clusters is less than actual categories. Summarized in the language of calculus, the per unit final inertia curve should have an inflection point.

The per unit final inertia graphs *(Graphs 1 A and B)* show this pattern emerging, with an inflection approximately when allowing 1% of the data size as clusters. If all XBRL tags were truly unique and therefore necessary, they should occupy a unique portion of the vector space. Clustering to 50% should leave significant inertia, much less 5%. This behavior, common to datasets with some form of underlying structure, supports the idea that even standard XBRL tags have some degree of redundancy. *Graph 1B* shows a second promising trend, a significant improvement in performance by increasing vector length to 200. A second series of vector length tests, probably requiring

increased parallelization to handle the computational load, could help to identify a more optimal length.



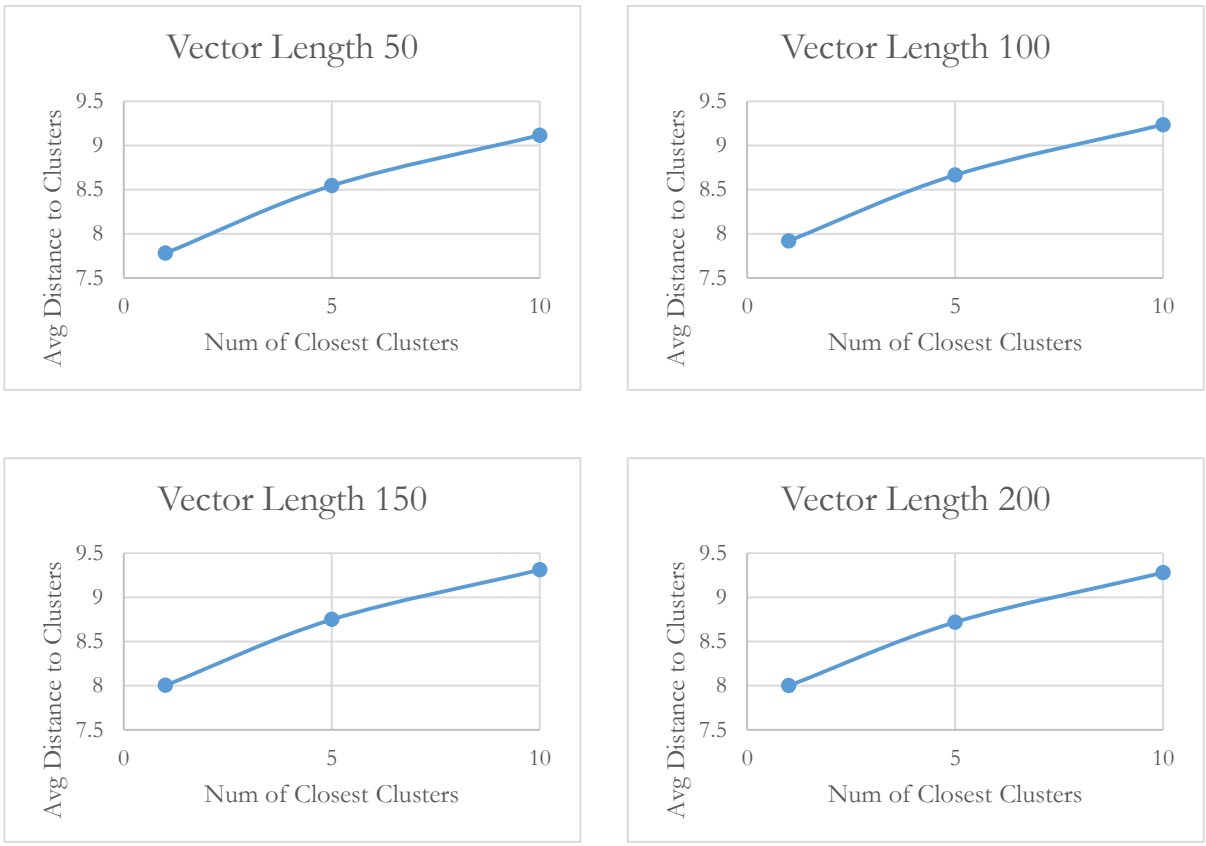
GRAPH 1(A) – PER UNIT FINAL INERTIA BY CATEGORY



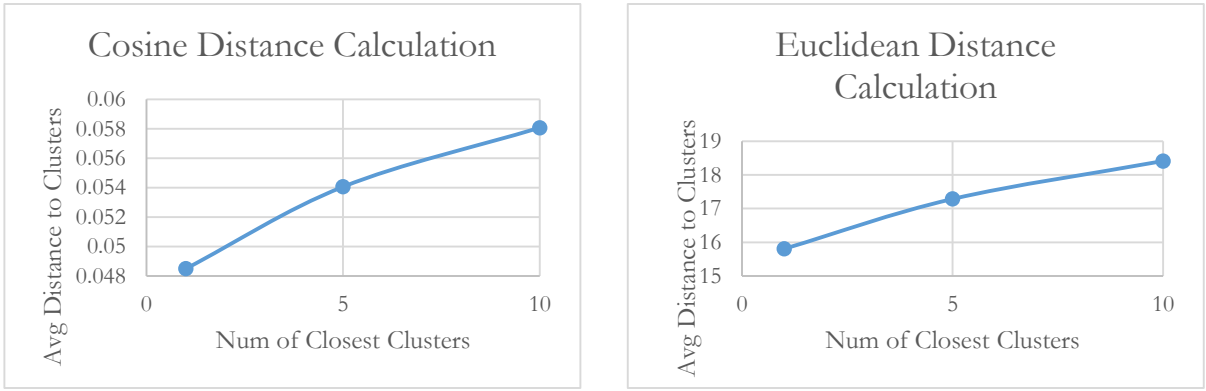
GRAPH 1(B) – PER UNIT FINAL INERTIA BY VECTOR LENGTH

N-Best Performance Analysis

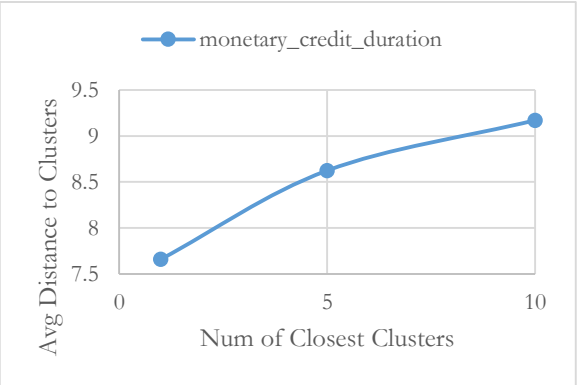
Without a unit of measurement, distances in a high-dimensional vector space have little concrete meaning, particularly when compared across multiple different high-dimensional vector spaces. In the absence of a ground truth performance metric, a novel measurement is used to verify expected behavior within the vector space. Each custom term is mapped to the nearest n clusters for n in $\{1, 5, 10\}$, and the average distance to the n clusters is recorded. If a custom term vector is arbitrarily close to many clusters, average distance will remain constant as n increases. If instead the custom vector is mapped to a unique cluster, average distance will increase with n. This behavior is confirmed across all variations of the data, once the Monetary-Unknown-Duration category is removed (due to distortions from small sample size). The one notable exception is the Monetary-Debit-Point category, which similarly suffered from small sample size.

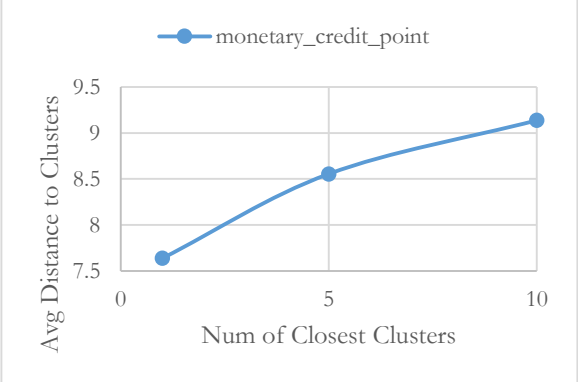


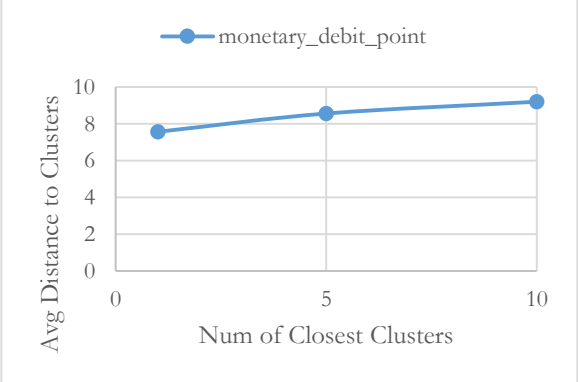
GRAPH 2(A) – N-BEST ANALYSIS OF VECTOR LENGTH

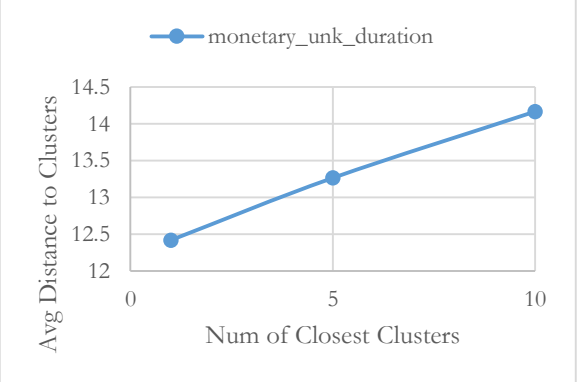


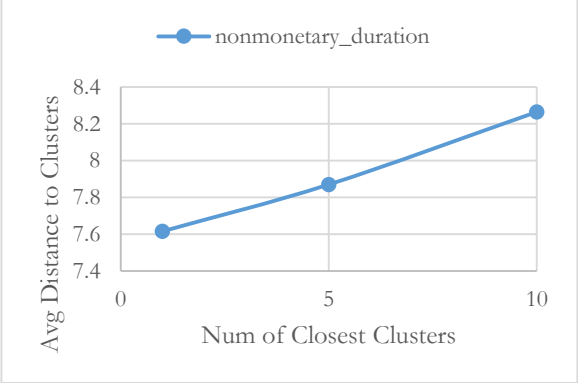
GRAPH 2(B) – N-BEST ANALYSIS OF DISTANCE FUNCTION

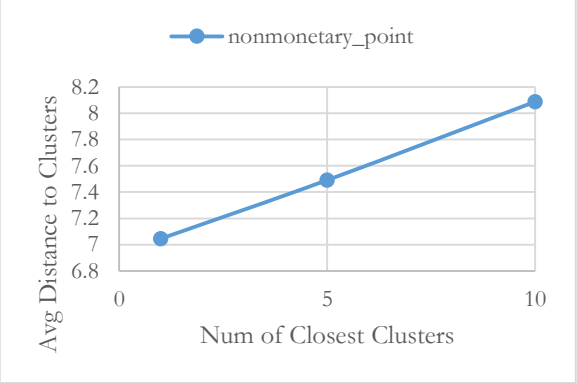












GRAPH 2(C) – N-BEST ANALYSIS OF CATEGORY

Select Term Mapping Examples

Finally, select instances of nearest and farthest vector mappings show examples of system success and failure. While cherry-picked, searching all mapped terms would require some form of ground truth accuracy metric. Unfortunately, constructing such a metric with unlabeled data would require manual labelling—an undertaking far beyond this scope. Examples are presented with their original term and any portion of their definition that serves important clarification.

Issuance of shares for cash	Stock Issued During Period
Series A Convertible preferred stock, shares available to be issued	Preferred Stock, Shares Issued, Total number of nonredeemable preferred shares (or preferred stock redeemable solely at the option of the issuer) issued to shareholders.

TABLE 1(A) –BEST EXAMPLES

Accounts Payable settled in Common Stock	Research and Development Expense
Depreciation, Depletion and Amortization, Including Other Property, Plant, And Equipment	Exploration Expense

TABLE 1(B) – WORST EXAMPLES

Conclusions

Without a ground truth accuracy metric, gauging the overall success of the system is perilous at best. Many of the metrics report expected behavior, and a number of example mappings make intuitive sense. For a high-level overview of financial statements, "Issuance of shares for cash" and "Stock issues during period" are effectively equivalent. However, many more bear no resemblance at all, suggesting this model remains far from its original—if lofty—objectives.

However, developments likely to come in the next six to twelve months promise marked improvements in the underlying tools that power this analysis. First among these is the open-source release of Google sequence-to-sequence (seq-2-seq) models and TensorFlow APIs to train custom embedders and encoders, which occurred on April 11, 2017 (between the conclusion of research and the final submission of this work). This would allow for much more accurate vector space embeddings by applying accounting-specific understandings of English language to the sequences rather than simply general English.

Second is a series of expected updates to libraries that will allow for more parallelized and heterogeneously distributed operations. A disconcerting portion of the code in this research had no parallelization at all, as many of the relevant libraries require community updates to accommodate recent updates to TensorFlow. Parallelization will in turn allow for faster analysis, longer vector embeddings, and broader experimentation. Within the next twelve months, the scope of practical analysis could very well expand by a factor of 100 while overall speed improves by a factor of 200.

In direct response to the research questions, this thesis has demonstrated the following:

Document Acquisition: Can individual users obtain filings in a sufficiently timely manner such that their automated analysis can inform trading decisions?

EDGAR filings are not a suitable information source for High Frequency Trading given various sources of information delay. Furthermore, it is perilously difficult to even rival the access times of PDS subscribed institutions. However, ML-enabled document processing certainly has the potential to review filings faster and more completely than a human reader. While EDGAR has little place in an HFT strategy, it has a very real place in a fundamental strategy, performing a broad range of advanced valuation methodologies far faster than an individual human could.

Performance Considerations: Given the overarching objective of widely applicable usage, what reductions in data size and complexity are necessary for reasonable time performance on consumer-grade hardware? Do these reductions alter results to such an extent that it invalidates the methodology?

Hardware is not the limitation; software is the limitation. Running massive vector operations sequentially on consumer hardware will require dataset reductions that negatively impact results, as indicated in comparisons of per-unit-final-cluster-inertia across various vector lengths. By running the same operations in parallel, however, the same hardware would require significantly less problem reduction—certainly not enough to invalidate methodology.

Financial Statement Standardization: Can an algorithm learn to standardize the XBRL taxonomy to then generalize financial statements for the automated calculation of financial ratios?

An effective commercial deployment remains a distant prospect. The problem reductions required for the scope of this research effectively traded nuance for performance, but if expected advances in supporting software do occur, both performance and nuance are attainable.

Works Cited

Abadi, Martin, et. al. "TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems." *Google Research*. November 9, 2015. http://download.tensorflow.org/paper/whitepaper2015.pdf.

Adelson-Velsky, Georgy and Evgenii Landis. Myron J. Ricci, trans. "An algorithm for the organization of information." *Soviet Mathematics Doklady*, 3 (1962):1259-1263.

Brown, Larry. "Deep Learning with GPUs." *NVIDIA*. June, 2015. http://www.nvidia.com/content/events/geoInt2015/LBrown_DL.pdf.

Cho, Young Jun, Nilabhra Bhattacharya, and Jae Bum Kim. 2014. "XBRL Mandate and Access to Information: Evidence from Reactions of Financial Analysts and Institutional Investors."

Christensen, Theodore E, William G Heninger, and Earl K Stice. 2013. "Factors Associated with Price Reactions and Analysts' Forecast Revisions Around SEC Filings." Research in Accounting Regulation 25 (2):133-148.

Collobert, Ronan, Jason Weston, Leon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. "Natural Language Processing (almost) from Scratch." March 2, 2011. arXiv:1103.0398 [cs.LG].

Corrado, Greg. "Computer, Respond to this Email." *Google Research*. November 3, 2015. https://research.googleblog.com/2015/11/computer-respond-to-this-email.html.

Dean, Jeffrey and Sanjay Ghemawat. "MapReduce: simplified data processing on large clusters." *Communications of the ACM,* 51, no. 1 (2008): 107-113. doi: 10.1145/1327452.1327492.

Drake, Michael S, Darren T Roulstone, and Jacob R Thornock. 2014. "Investor Information Demand: Evidence from Google Searches Around Earnings Announcements." Available at SSRN 1991455.

———. 2015. "The Determinants and Consequences of Information Acquisition via EDGAR." Contemporary Accounting Research 32 (3):1128-1161.

———. 2015. "The Usefulness of Historical Accounting Reports." Available at SSRN 2083812.

Engelberg, Joseph, and Srinivasan Sankaraguruswamy. 2007. "How to gather data using a web crawler: An application using SAS to search EDGAR." Available at SSRN 1015021.

Fielding, R., et. al. "Hypertext Transfer Protocol." *ISI*. June, 1999. https://tools.ietf.org/html/rfc2616.

García, Diego, and Øyvind Norli. 2012. "Crawling Edgar." The Spanish Review of Financial Economics 10 (1):1-10.

Grefenstette, Edward, Phil Blunsom, Nando de Freitas, and Karl Moritz Hermann. "A Deep Architecture for Semantic Parsing." April 29, 2014. arXiv:1404.7296 [cs.CL].

Harris, Trevor Samuel, and Suzanne G Morsfield. 2012. "An evaluation of the current state and future of XBRL and interactive data for investors and analysts."

Hoitash, Rani, and Udi Hoitash. 2015. "Measuring Accounting Complexity with XBRL." Available at SSRN 2433677.

Jackson, Robert J, and Joshua Mitts. 2014. "How the SEC helps speedy traders." Columbia Law and Economics Working Paper (501).

Kalchbrenner, Nal, Edward Grefenstette, and Phil Blunsom. "A Convolutional Neural Network for Modelling Sentences." April 8, 2014. arXiv:1404.2188 [cs.CL].

Karpathy, Andrej and Li Fei-Fei. "Deep Visual-Semantic Alignments for Generating Image Descriptions." April 14, 2015. arXiv:1412.2306v2 [cs.CV].

Karpathy, Andrej. "The Unreasonable Effectiveness of Recurrent Neural Networks." May 21, 2015. http://karpathy.github.io/2015/05/21/rnn-effectiveness/.

Kim, Yoon. "Convolutional Neural Networks for Sentence Classification." August 25, 2014. arXiv:1408.5882 [cs.CL].

Kolivas, Con. "Long Range Zip." November 13, 2016. https://github.com/ckolivas/lrzip.

Lee, Jason, Kyunghyun Cho, and Thomas Hofmann. "Fully Character-Level Neural Machine Translation without Explicit Segmentation." October 10, 2016. arXiv:1610.03017 [cs.CL].

Libby, Dan. "RSS." *Netscape Communications*. July 10, 1999. https://web.archive.org/web/20001204093600/http://my.netscape.com/publish/formats/rss-spec-0.91.html.

Lloyd, Stuart P. "Least Squares Quantization in PCM's." *IEEE Transactions on Information Theory,* 28, no. 2 (1982):129-136. doi: 10.1109/TIT.1982.1056489

Loughran, Tim and Bill McDonald. "The Use of Word Lists in Textual Analysis." February 1, 2015. Forthcoming in the Journal of Behavioral Finance. doi: 10.2139/ssrn.2467519.

———. August 2, 2016. "The Use of EDGAR Filings by Investors." Available at SSRN 2493941.

Mahoney, Matt. "ZPAQ." August 17, 2016. https://github.com/zpaq/zpaq.

Mikolov, Thomas, et. al. "Distributed Representations of Words and Phrases and their Compositionality." October 16, 2013. arXiv:1310.4546v1 [cs.CL].

Ng, Andrew and John Duchi. "CS229: Machine Learning." *Stanford University*. October 12, 2016. cs229.stanford.edu.

O'Riain, Seán, Edward Curry, and Andreas Harth. 2012. "XBRL and open data for global financial ecosystems: A linked data approach." International Journal of Accounting Information Systems 13 (2):141-162.

Postel, J. and J. Reynolds. "File Transfer Protocol." *ISI*. October, 1985. https://tools.ietf.org/html/rfc959.

Rogers, Jonathan L, Douglas J Skinner, and Sarah LC Zechman. 2015. "Run EDGAR Run: SEC dissemination in a high-frequency world." Chicago Booth Research Paper (14-36).

Scrivano, Giuseppe and Hrvoje Nikšić. "GNU Wget." *Free Software Foundation*. February 13, 2017. https://www.gnu.org/software/wget/.

Shen, Yelong, Xiaodong He, Jianfeng Gao, Li Deng, and Gregoire Mesnil. "Learning Semantic Representations Using Convolutional Neural Networks for Web Search." *Microsoft Research*. April 1, 2014.

Starr, Mike, and Susan Yount. 2013. "The state of SEC reporting using structured data: a view of the current state of financial reporting by public companies filing structured data, including the recommended changes necessary for the use of XBRL to enhance regulatory oversight and increase transparency and accountability." Financial Executive 29 (7):16-20.

Sutskever, Ilya, et. al. "Sequence to Sequence Learning with Neural Networks." December 14, 2014. arXiv:1409.3215v3 [cs.CL].

Sutskever, Ilya, Oriol Vinyals, and Quoc V. Le. "Sequence to Sequence Learning with Neural Networks." December 14, 2014. arXiv:1409.3215v3 [cs.CL].

Szegedy, Christian, et. al. "Going Deeper with Convolutions." September 17, 2014. arXiv:1409.4842 [cs.CV].

Tetlock, Paul C. 2010. "Does public financial news resolve asymmetric information?" Review of Financial Studies 23 (9):3520-3557.

———. 2014. "Information transmission in finance." Annu. Rev. Financ. Econ. 6 (1):365-384.

Thomas, Susan Marie, Xichuan Wu, Yue Ma, and Sean O'Riain. 2014. "Semantically Assisted XBRL-Taxonomy Alignment Across Languages." In Towards the Multilingual Semantic Web, 277-293. Springer.

Vinyals, Oriol, Alexander Toshev, Samy Bengio, and Dumitru Erhan. "Show and Tell: A Neural Image Caption Generator." April 20, 2015. arXiv:1411.4555v2 [cs.CV].

Wang, Ding, et. al. 2013. "Semi-automatic Generation Model of Elements in XBRL Taxonomy." DOI: 10.4028/www.scientific.net/AMM.411-414.467.

Williams, Kelly L, Mitchell R Wenger, and Rick Elam. 2012. "Analysis of Actual Company Filings Using XBRL: An Empirical Investigation into Interactive Use of XBRL Financial Statement Data." Available at SSRN 2172325.

Wu, Yonghui, et. al. "Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation." October 8, 2016. arXiv:1609.08144v2 [cs.CL].

"Accessing EDGAR Data." *Securities and Exchange Commission.* March 17, 2017. https://www.sec.gov/edgar/searchedgar/accessing-edgar-data.htm.

"EDGAR Log File Data Set." *U.S. Securities and Exchange Commission*. December 23, 2015. https://www.sec.gov/data/edgar-log-file-data-set.

"Hierarchical Data Format." *The HDF Group.* 2017. http://www.hdfgroup.org/HDF5/.

"Index to Forms, EDGAR Filer Manual Vol. II." *U.S. Securities and Exchange Commission*. August, 2015. https://www.sec.gov/info/edgar/forms/edgform.pdf.

"Structured Disclosure at the SEC: History and Rulemaking." *U.S. Securities and Exchange Commission*. April 27, 2016. https://www.sec.gov/structureddata/historyandrulemaking.

"Structured Disclosure RSS Feeds." *Securities and Exchange Commission.* February 23, 2017. https://www.sec.gov/structureddata/rss-feeds-submitted-filings.

Works Consulted

Low-Level Text

Bodnaruk, Andriy, Tim Loughran, and Bill McDonald. 2013. "Using 10-K Text to Gauge Financial Constraints." Available at SSRN 2331544.

Feldman, Ronen., et. al. 2010. "Management's tone change, post earnings announcement drift and accruals." Review of Accounting Studies (2010) 15:915-953.

Hoberg, Gerard and Gordon Phillips. 2010. "Product Market Synergies and Competition in Mergers and Acquisitions: A Text-Based Analysis." The Review of Financial Studies 23 (10).

Lee Jihwan, and Yoo S. Hong. 2014. "Business Model Mining: Analyzing a Firm's Business Model with Text Mining of Annual Report." Industrial Engineering & Management Systems 13 (4):432-441.

Loughran, Tim, and Bill McDonald. 2011. "When Is a Liability Not a Liability? Textual Analysis, Dictionaries, and 10-Ks." Journal of Finance 66 (1):35-65.

———. 2015. "Textual Analysis in Accounting and Finance: A Survey." Journal of Accounting Research. DOI: 10.1111/1475-679X.12123.

Shahi, Amir Mohammad, Biju Issac, and Jashua Rajesh Modapothala. 2012. "Intelligent corporate sustainability report scoring solution using machine learning approach to text categorization." 2012 IEEE Conference on Sustainable Utilization and Development in Engineering and Technology (STUDENT).

Tetlock, Paul C, Maytal Saar-Tsechansky, and Sofus Macskassy. 2008. "More than words: Quantifying language to measure firms' fundamentals." The Journal of Finance 63 (3):1437-1467.

Other Indicators

Lee, Charles M.C., Paul Ma, and Charles C.Y. Wang. 2014. "Search-based peer firms: Aggregating investor perceptions through internet co-searches." Journal of Financial Economics (2015). DOI: 10.1016/j.jfineco.2015.02.003.

Seemakurthi, Prasad, Shuhao Zhang, and Yibing Qi. 2015. "Detection of fraudulent financial reports with machine learning techniques." Systems and Information Engineering Design Symposium (SIEDS), 2015.

Market-Level Analysis

Brown, Michael Scott, Michael J Pelosi, and Henry Dirska. 2013. "Dynamic-radius species-conserving genetic algorithm for the financial forecasting of Dow Jones index stocks." International Workshop on Machine Learning and Data Mining in Pattern Recognition.

Engelberg, Joseph. 2008. "Costly Information Processing: Evidence from Earnings Announcements." Available at SSRN 1107998.

Ludvigsen, Hans-Marius Lee, and Michael Kiel Vollset. 2014. "Activist Hedge Funds: The Characteristics and Determinants of Abnormal Returns in Activist Hedge Fund Targets–an Event Study."

Zheng, Ying, Harry Zhou, Zhijiang Chen, and Nnanna N Ekedebe. 2014. "Automated analysis and evaluation of SEC documents." Computer and Information Science (ICIS), 2014 IEEE/ACIS 13th International Conference on.

Appendix A – Definition of Acronyms and Esoteric Terms

XBRL – eXtensible Business Reporting Language, a term/definition taxonomy for financial statements.

SEC – Securities and Exchange Commission, the regulatory body overseeing securities markets in the United States.

EDGAR – Electronic Data Gathering, Analysis, and Retrieval; the SEC database of historical and current filings.

PDS – Public Distribution Service, a commercial subscription available primarily to institutions which provides a near-real-time feed of EDGAR documents as they are submitted—often before the general public has access.

FTP – File Transfer Protocol

HTTP – Hypertext Transmission Protocol

HFT – High Frequency Trading

RNN – Recurrent Neural Network

LSTM – Long Short-Term Memory, a special kind of perceptron that allows a neural network to have a concept of time and memory.

CNN – Convolution Neural Network

Bag of Words – A simple method of NLP in which a phrase is treated like a bag of words; each word is equal, regardless of position. For example, bag of words regards "the cat ran" and "ran the cat" as the same phrase.

Vectorize – Most machine learning models take vectors as input. To handle complicated real-world problems, input data must be converted to a vector representation. This may be a simple numerical mapping, a one-hot adaptation for categorical variables, or a more complex approach such as with text.

Appendix B – Supporting Libraries

Library	**Usage**	**Reference**
numy	Highly optimized array computations	http://www.numpy.org/
seq2vec	Sequence embedding class	https://github.com/Yoctol/seq2vec
joblib	Parallelzation of iterative proceedures	https://github.com/joblib/joblib
tensorflow	Large-scale, heterogeneous, and distributed machine learning	https://www.tensorflow.org/
h5py	HDF5 data storage and access interface	http://www.h5py.org/
scipy	Vector distance functions: euclidean, cosine, cdist (distance between every vector x in set a and every vector y in set b)	https://www.scipy.org/
pandas	Array aggregation and formatting for csv output	http://pandas.pydata.org/
scikit-learn	KMeans clustering implementation	http://scikit-learn.org/

Appendix C – Deep Learning on Consumer Hardware

Overview

So you think you want to build a deep learning rig? Before you get started, there are a few things you should consider. First, professional deep learning takes professional hardware at a professional cost. You won't be rivaling Google any time soon. Second, don't discount the cloud. Google Cloud Platform offers a great combination of performance and price, but you sacrifice the warmth of your own silicon space heater. Third, remember that custom-built also means custom-support and custom-sysadmin. Be prepared to spend countless hours troubleshooting and tweaking, but if it's a labor of love, it's worth it. Finally, recognize that most of these algorithms take hours if not days to run, meaning your new box is a server first and a client second. You'll need a different daily driver.

If you're still bent on a custom-built deep learning rig, great. I was too. You'll want to keep a few design considerations in mind. First, take the online build-a-pc communities with a grain of salt. Most of these builders are looking for gaming performance or, in rare cases, workstations or enterprise servers. While some aspects of performance apply across use cases, getting the most performance per dollar requires a dedicated design. Second, remember that computation is a process. Especially in deep learning, speed is limited by the slowest link—or bottleneck. You may find that no CPU in the world can overcome overwhelmed GPU memory (as I did…). Finally, just as most pc builders are focused on different use cases, so too are most manufacturers. You'll have to peer a bit through the marketing to really see the value of components in deep learning.

In general, a deep learning rig needs to do three things well:

1. Most computation is done on GPU, so raw power largely depends on the GPU and the system's ability to get data to and from that GPU.
2. Deep learning requires big datasets, so system memory and storage can play a crucial role in deciding which problems you can tackle and how.
3. It takes forever, so cooling and noise over extended peak operation matter, particularly if you're planning to sleep next to a panting rig.

Component Breakdown

CPU – While the clock speed and thread count are less important than in other uses, clock speed can be valuable to the novice programmer (who doesn't multithread much) and thread count can be particularly useful in server environments where many processes are running at once. Most importantly, however, is PCIe lane support. GPUs need at least 8x PCIe-3 lanes. You'll probably want some PCIe based storage, and perhaps even additional networking or RAID support. If you're looking towards a multi-GPU setup, higher end CPUs with 40x PCIe lanes may be a requirement.

CPU Cooler – In pursuit of PICe lanes, the CPU will probably be plenty fast without overclocking. But that isn't to say extreme cooling can't help, particularly if you plan to throttle the CPU for hours on end while you're trying to sleep.

Motherboard – All of those cool devices that drive up the PCIe lane count also drive up the motherboard requirements. Not all boards support multi-GPU configurations, and many have

secret tradeoffs found only in the user manual. The board might have two USB 3.0 headers, but you must deactivate two SATA ports to use the second (happened on mine!). Design the system, and then consult the user's manual carefully to ensure everything will function together. Rear I/O requirements can vary – probably not going to need dozens of USB peripherals and might not even need Ethernet (if you plan on installing a stand-alone NIC).

Memory – There are few reasons for most users to go up to or beyond 32GB of memory, but deep learning almost requires it. Lifting the ceiling on the size of dataset you can manipulate without dealing with file-system operations can be a lifesaver. That said, it depends on problem type. You could spend a fortune to install 128GB, but that's still not a big dataset by deep learning standards. If you're resigned to file-system operations, there isn't much reason to go above 32GB.

Storage – Tiered storage is key, because it allows for a range of speed/space tradeoffs. PCIe based SSDs make great caching drives for larger HDD RAID arrays. One of the great things about deep learning is that the program knows exactly the next chunk of data it will need, allowing caching to work like a breeze—effectively bringing a RAID 5 array to PCIe speeds when training. Because some datasets, particularly if you plan on archiving old ones, can require huge amounts of storage, Network Attached Storage may become a necessity. In that case, 10Gb/s networking may become a necessity. Try transferring 4TB at 1Gb/s.

GPU – It's the bread and butter, so keep three things in mind. First, the performance that matters is FLOPS (floating-point-operations per second). Too few cores at too high a clock rate, and you have an expensive CPU. This is a parallel computing device, so let it be massively parallel. Second, GPU memory is the killer of most algorithms. Gaming doesn't require much, so consumer GPUs lack the memory capabilities of their commercial counterparts. Carefully evaluate the memory requirements of the models you expect to use. You may decide to give up and flee for the cloud. Finally, most deep learning software libraries only support Nvidia CUDA, not open-CL. If you're after deep learning and you don't want to know how to write GPU code, you're stuck with team green.

Power Supply – A combination of a RAID array, multiple GPUs, and assorted whistles can require a lot of juice, particularly under load. While the absolute best isn't a necessity, the system does need reliable power delivery at the top end of its range. Whether you accomplish this with a higher wattage or better performance is a pretty moot point.

Extras – Just like any other server deployment, a UPS is probably essential. If your rig loses power in the middle of a week-long training, it's back to square one. Don't worry much about other peripherals; this isn't a daily-driver.

[i] These questions have been modified since the original thesis proposal on October 28, 2016. After parsing text data, the author discovered the sheer volume of XBRL data (285,102 different tags in 2016 alone). While storage requirements were minor, computational complexity quickly outpaced all but high-end cloud infrastructure. In the interest of keeping work accessible and duplicable, the question regarding "Performance Considerations" was introduced.

Additionally, the following research question was removed due to the shutdown of EDGAR FTP servers on December 30, 2016. While it remains possible to analyze and parse documents collected prior to the shutdown, this is an inconsequential exercise without regular and programmatic access to the previous dataset.

Intra-Document Retrieval *(removed)*: Can an algorithm learn to identify relevant sections of a document and parse its content first for human readers and ultimately as input for another algorithm?