Event-Based Financial Risk Analysis Toolkit

Rodz Andrie Amor, Keshav Ganapathy, Sash Lamba, Sakshi Gholap, The-Vinh Calix Tran-Luu
University of Maryland, College Park
ramor12@umd.edu, keshavg@umd.edu, sash19@umd.edu,
sgholap@umd.edu, tranluu@umd.edu,

1 Introduction & Intended Audience

Our group has developed a web-based application that serves to assist investors, financial analysts, and portfolio management with an alternate approach to risk analysis. Traditionally, financial risk analysts take a quantitative approach, relying heavily on historical financial data and statistical models to predict future risks and opportunities. However, this approach often overlooks qualitative data, such as news events and company filings, which can provide critical insights into emerging risks.

To address this gap, our toolkit references the risk sections of companies' SEC 10-K filings. These annual reports, required to be submitted for all public companies, provide an overview of business processes, financial statements, and risk factors. By analyzing a corpus of hundreds of company filings, we can identify frequently mentioned risk types, such as weather, economic, or business risks, which indicate high concern within the company.

Our method determines a company's exposure to risk by assessing whether a headline or event is semantically similar to risks mentioned in the company's filings. This is achieved through natural language processing techniques, detailed in the methodology section. In addition to the web-based toolkit, we have developed a command-line application for generating, building, and querying detailed knowledge graphs. These graphs visualize relationships between companies, their attributes, and risk factors, enhancing investors' ability to make informed decisions.

Our toolkit aims to be integrated alongside the general quantitative approach and help the investor approach news events and qualitative information. We hope to provide utility when an event occurs and an investor is looking to query which companies have listed a concern in their risk factors.

2 Motivation & Goals

The primary objective of this research is to develop a toolkit that helps investors connect events to company risk factors. An event in the context of our research represents a headline or occurrence that affects a business process. By providing a structured and automated way to correlate events with their potential financial impacts, the toolkit aims to enhance the decision-making process for investors.

The goal for the toolkit is to be able to list companies given a specific headline or news event. Alternatively, when provided with a dataset of news events and company risk filings, the toolkit should be able to prepare a knowledge graph with the appropriate connections. We chose to make this toolkit as a web application as it's an easy way to use it and a multitude of different features can be added. One useful feature would be the ability to download the SEC-10K filing data directly from the web application, as it's used by the toolkit in sentiment analysis and can provide more context to the user. In addition to the web-based toolkit, we'd offer a command line application that can build, query, and encode a more detailed knowledge graph than the one present in the web application. Finally, the last piece of the web-based toolkit is a visualization of the knowledge graph that connects companies and nodes together. The user will be able to click on a company or event to extract more information. This has its applications with allowing investors to visualize more complex relationships. The next sections will elaborate on the functionalities and implementations of the tools.

This is a set of tools that each serve their own purpose to an investor that is particularly interested in a company's risk. From the headline sentiment analysis, risk text downloader, to the knowledge graph, we intended for each of these to have utility for an investor and aid in their decision-making.

3 Literature Review

Mining Financial Risk Events From News and Assessing their Impact on Stocks [1]

This paper focuses on identifying and extracting financial risk events from news articles and analyzing their subsequent impact on stock markets. The research is grounded in the notion that financial news contains critical information that can significantly influence stock prices. By mining this information and correlating it with stock market data, the study aims to provide insights into how specific events affect financial markets. The key steps are in information extraction, event classification, impact assessment, and predictive modeling. Furthermore, this paper utilized natural language processing techniques to extract relevant events from news articles. By providing a systematic approach to event extraction and impact assessment, the research offers valuable tools for investors and analysts to understand and anticipate market movements better. Similarly, our project involves using natural language processing techniques to calculate the similarity between text in a risk filing and an event. However, instead of using a convolutional neural network based event identifier, we used the RoBERTa text transformer to classify events.

Event Prediction using Case-Based Reasoning over Knowledge Graphs [2]

In a high level overview, this paper explores an innovative approach to predicting events by leveraging case-based reasoning and knowledge graphs. The primary task addressed in this paper is the prediction of new events based on known causal relationships between existing events in a knowledge graph. For example, if the knowledge graph includes a causal relationship between a specific type of earthquake and subsequent tsunamis, the model can predict properties of a new earthquake event based on this historical data. Our research project was heavily inspired by the utility of this knowledge graph, and its potential application for financial risk analysis. Although our project was not able to incorporate the scenario planning advisor, we were heavily inspired by the utility of the knowledge graph and its potential applicability to financial risk analysis. Therefore, we built a knowledge graph that details information on the company, their risk factors, and their relationships to each other. We hoped to add utility by the visualization of these relationships and details within each node.

4 Data Sources & Pipeline

4.1 Data Collection

The two sets of data that served as the foundation for our application is the company risk text and the event data. We needed text that can be used to identify risk factors for companies. While this can be anything from quarterly earnings reports to financial news broadcasts, we determined the best way to reliably get access to a large corpus of such text was the Risk Section of SEC-10K filings. On the other hand, event data needed to have information on the time, duration, and description in order to be viable as used to identify risk factors in the knowledge graph.

4.2 Downloading and Extracting SEC Data

We scraped SEC filings from the SEC Edgar Database using a tool we developed that utilized BeautifulSoup, a Python web scraping tool. The full 10-K corpus is processed, and the risk section is parsed out for about 300 companies going back 20 years. The text is cleaned by removing special characters and reducing common boilerplate text. The only information that remained after the cleaning is the text, company name, ticker, filing date, and the Fama French industry sector of the company.

4.3 Data Cleaning and Parsing

Our initial method for acquiring event data was to simply look for datasets of headlines, and news events in Kaggle. These were formatted as CSVs of news headlines, which made it easy to use and freely available, but was very limited in the scope of the information. Consequently, it was not detailed enough to label its risk factors, and we pivoted to other datasets. Next, we focused on a weather dataset on Kaggle to get accurate historical data and convert it into text headlines by using NLP. After this, we used the GPT API to turn them into events. This made it easy to use, accurate, and uniform across the dataset, but had the drawback of being too fine-grained. Unfortunately, during development we found that no real matches were being detected, which made it unviable for the purposes of our project.

4.4 Final Pipeline

The final pipeline for getting events data used a multitude of techniques and sources to acquire relevant data. It was determined that the best way to get data relevant for filings would be to work backwards from the filings itself. We followed the following steps written in a Python Jupyter Notebook:

1. Randomly select a number of companies and years from which news data can be extracted.
2. After obtaining the risk sections, clean it by stemming and lemmatizing the words. Feed this risk section to an NLP API along with instructions to extract events mentioned in the risk section. Although we tried to run a locally pretrained transformer model, RoBERTa, we ultimately used the GPT API.
3. The GPT API returns a list of possible events. Unfortunately, the majority consisted of boilerplate events such as “inflation” or “last quarter losses”. However, the list almost always includes actual real-life events mentioned in the text.
4. To filter real-world events, a we built a custom Google News Scraper.

- The scraper uses HTML and RSS feeds to query Google News and scrapes the responses into a Python Dictionary of news headlines
- The dictionary includes the event headline, a small description, time, and publisher for the event
- Inputs come in the form of a tuple of text query and time period

5. The list of possible events is fed to the Google News API and checked for results. If no headlines are returned for an event, then it's likely boilerplate and is skipped.

6. For possible events which have Google News headlines associated with them, a small similarity metric is used to rank which news headline is most similar to the associated event in the risk section.

7. The top ranked headline which also has a high enough similarity metric is chosen to be used in the final dataset.

8. This pipeline provides a clean, accurate, and relevant dataset which can then be used to find connections between events and SEC risk filings.

Under these steps, this had the benefits of providing clean and relevant data. However, the disadvantages of this data collection method are that the process was very slow, taking 5 to 10 minutes to collect and per entry and that we were only able to get a small amount of real-world events. It also seemed that popular events such as COVID and election news were overrepresented in the dataset.

4.4.1 Challenges & Solutions

The development of the various parts of the toolkit presented several challenges, which we approached with different implementations to solve. These challenges spanned various aspects of the project, from data scraping and cleaning to text analysis and visualization. In addition to building the tools, the frontend presentation needed to be visually appealing and easy to understand.

4.5 Datsets

One of the primary challenges we had was creating and designing the dataset itself. The rest of the toolkit, the frontend, backend, and the sentiment analysis were developed before the dataset itself could've been fully finalized. This led to a great deal of boilerplate and irrelevant text going through. It seemed that curating the dataset was the slow bottleneck which reduced the quality of the final output of the knowledge graph and sentiment page.

4.6 Aggregating SEC 10-K Filing History

Another challenge was extracting and processing risk sections from the SEC 10-K filings for several hundreds of companies. Fortunately, the SEC enforces strict and structured texts for all 10-K filings, which substantially helped us in the filtering process. Initially, a class built around SEC downloader APIs was utilized, but its limitation for only a certain number of companies meant it wasn't ideal. Consequently, a custom implementation of web scraping was utilized that did not have such limitations. The process itself is mentioned above, the final dataset is a series of CSVs containing the cleaned risk sections, date of filing, company ticker, and its fama french sector.

4.6.1 Document Cleaning and Parsing

Cleaning data was an essential part for determining correct connections. Data such as HTML tags, formatting keywords, and irrelevant sections were prevalent across the datasets. A data cleaning pipeline was implemented that removed unnecessary content, standardized the formatting, and retained only the relevant risk sections. As an example, all `&nbsp;` tags and `*` were removed, which are the remains of the HTML formatting. Regular expressions to pattern match the start and end of the risk factor section were used, discarding other sections. We used the spaCy NLP library to stem and lemmatize the words. However, filtering boilerplate texts from real-world events remained a challenge.

4.7 Textual Semantic Similarity Evaluation

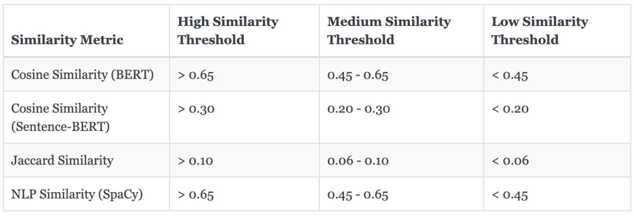
Calculating the semantic similarity between news headlines and risk sections from SEC filings was another significant challenge. This task involved evaluating the textual similarity to determine if a particular event mentioned in a news headline was discussed in a company's risk section. However, text, ideas, and concepts are abstract notions that are difficult to quantify. More NLP tools were used to aid in this process.

Given that a company's risk section discusses several different areas, it seemed necessary to perform text segmentation and perform similarity analysis for each segment. Using the spaCy NLP library, each risk section was segmented into sentences and paragraphs, allowing for a more granular analysis of the risk factors. Each of these sentences are ranked with respect to their similarity with the news headlines.

To calculate the semantic similarity, several similarity metrics were used:

Cosine Similarity with BERT

The BERT transformer was used to transform the text into word embeddings, which are dense vector representations that capture abstract meaning. The vector representation places similar concepts close together in the vector space, allowing



Similarity Metric	High Similarity Threshold	Medium Similarity Threshold	Low Similarity Threshold
Cosine Similarity (BERT)	> 0.65	0.45 - 0.65	< 0.45
Cosine Similarity (Sentence-BERT)	> 0.30	0.20 - 0.30	< 0.20
Jaccard Similarity	> 0.10	0.06 - 0.10	< 0.06
NLP Similarity (SpaCy)	> 0.65	0.45 - 0.65	< 0.45

Figure 1: Similarity Thresholds for Various Metrics

for similarity operations. Cosine similarity, which ranges from -1 to 1, was used to compare these vectors. A cosine similarity of 1 indicates that the texts are identical, whereas a similarity of 0 indicates no similarity. A similarity of -1, in this context, indicates complete dissimilarity. Through testing, a threshold of 0.66 was determined to represent high similarity, 0.4 to 0.65 as medium similarity, and anything lower as low similarity.

$$\cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\|\|\mathbf{B}\|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\sqrt{\sum_{i=1}^{n} B_i^2}}$$

Figure 2: Formula for Cosine Similarity

Jaccard Similarity

Jaccard Similarity is a metric that measures the similarity between two sets by comparing the size of the intersection to the size of the union of the sets. For textual data, this involves comparing the sets of words in the sentences. It is important to note that Jaccard Similarity often yields lower values. Thus, a similarity score above 0.10 indicated high similarity, 0.06 to 0.10 indicated medium similarity, and below 0.06 indicated low similarity.

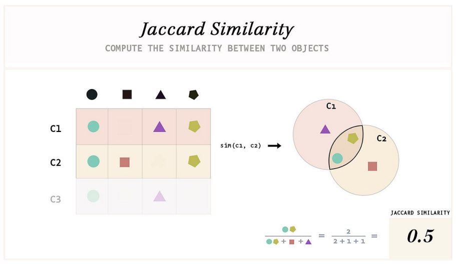


Figure 3: Formula for Jaccard Similarity

Natural Language Event Classification

Classifying events mentioned in the news headlines into predefined categories was a crucial step into organizing and analyzing the data in the knowledge graph. The categories of the classified events are: General, Weather, Political, Economy, Energy, Business. This classification needed to be accurate, given the diversity of event types and the variability in news reporting.

In order to perform this classification, the RoBERTa transformer model was utilized for its superior performance in text classification tasks. RoBERTa is based on the BERT architecture but optimized with better training techniques and larger datasets.

The RoBERTa model follows a similar procedure of tokenizing (stemming/lemmatization) the input and converting the text into word embeddings. The output of the model is fed into a softmax layer to get the classification probabilities among the classes. The event is labeled as the category with the highest predicted probability.

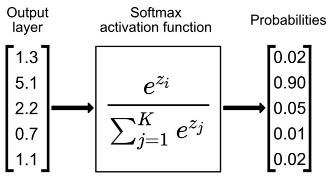

Output layer
Softmax activation function
Probabilities
1.3
5.1
2.2
0.7
1.1
$\frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}}$
0.02
0.90
0.05
0.01
0.02

Figure 4: Formula for Softmax

4.8 Website Interactive Knowledge Graph

Building and visualizing knowledge graphs that accurately represent relationships between events and companies was a complex task. Knowledge graphs needed to be interactive and user-friendly, allowing investors to explore and understand the data intuitively. We used the event dataset, which is outlined in the previous section, in order to populate the knowledge graph.

During the development process, we faced a tradeoff between the amount of detail that can be sent from the backend and what can be visualized in the frontend. There are limited React libraries that can encode a knowledge graph with the amount of information we wanted to display. Consequently, the knowledge graph displayed on the website consists of the company and events labeled by the event type or industry sector. This information is limited by the overall lack of complex React Graph Visualization libraries, as well as the ability to transmit the information from the Python backend to React frontend. For investors who want a more detailed knowledge graph, we developed a Python application that enables you to build and query a more complex graph that can represent more nodes than company and event.

5 Web-Based User Interface & Features

The user interface of the toolkit is designed to be intuitive and user-friendly to provide a useful experience for investors.

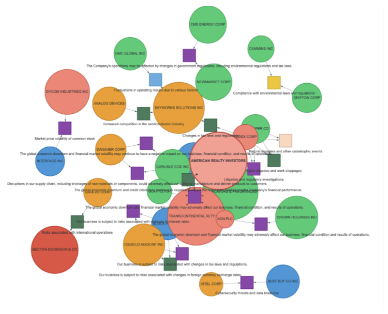


Figure 5: Website Knowledge Graph

5.1 Financial Event Risk Analysis

The primary page is the financial event risk analysis tool, which is used to evaluate how different companies are affected by a headline in a calendar year. The user has three dropdowns that affect which risk filings are evaluated: the filing year, the company farma french sector, and the semantic evaluation metric. Additionally, the user can either type in their own event or use the prefilled headlines that are obtained from the GNews API. Once the user clicks the analyze button, a table will be populated, with each row representing a company. An entry in the table will either have a semantic similarity labeled high, medium, or low and will be colored green, yellow, and red respectively. By doing so, this gives users a more interpretable way to understand the similarity, instead of the previous implementation which displayed a numerical value from 0 to 1. Additionally, the corresponding text segment in the SEC filing that had the highest recorded similarity displayed on the last column. At the bottom of this page, users can download the table as a CSV file. Finally, if a user clicks on a row entry, the user will be taken to a different page that displays stock information and basic information about the company.

5.2 SEC Risk Filing Query

The second feature is an SEC risk filing query system. This feature provides a resource for downloading the risk sections of SEC 10-K documents for various companies. The interface allows users to search for company risk filings by entering a stock ticker or selecting a sector from a dropdown menu. The page displays a table listing various companies, their corresponding sector classifications, and links to view stock charts. Additionally, users can download risk filings in CSV format for each company, enabling easy access to historical risk text data. If a user wants access to the direct text that is used in the semantic similarity evaluation, they can download a company's filing history as a CSV. This tool is designed to facilitate detailed analysis and monitoring of financial risks associated with different companies across various sectors.

5.3 Knowledge Graph

The Knowledge Graph feature is a comprehensive visualization tool. It displays various companies categorized by sector, such as Consumer, Manufacturing, HiTec, Health and Medical, Energy, and Other. The events impacting these companies are color-coded by type, including General, Weather, Political, Economy, Energy, and Business events. The interface allows users to filter data by sector and event type, as well as to adjust the minimum number of connections displayed on the graph. The interactive graph visualizes the relationships and risk factors associated with different companies. The graph visualization renders the company as a circle and an event as a square. The user can click on a company node to get a list of all the risk factors. Similarly, the user can click an event to will display a list of all the companies connected to it, indicating they are at high risk to the particular event.

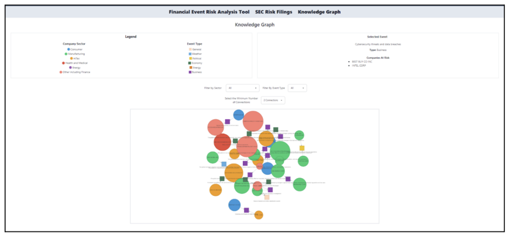


Figure 6: Knowledge Graph Page and Legend

6 Implementation Details

6.1 Web-Based Toolkit

The web-based financial risk analysis toolkit is designed using a modern technology stack to support a faster development process. Notably, the frontend is developed using the React framework, which allows us to leverage the large numbers of component libraries. For example, we used the React-Graph-Vis library in order to build the web knowledge graph. In order to quickly style the page, we used Tailwind CSS along with DaisyUI components with pre-designed UI elements. Finally, the frontend is hosted on Vercel to deploy the React application live on a domain that anyone can access.

On the backend, the toolkit leverages Flask, a lightweight WSGI web application framework written in Python. This backend handles various tasks, including data retrieval, processing, and serving data to the frontend through a REST API with multiple GET endpoints. Every endpoint is set to be a GET endpoint and returns a JSON formatted object, as the frontend will only need to be retrieving data. The backend is It is compiled into a Docker container and deployed on Google Cloud Run as a serverless instance. By doing this, we can easily configure how much compute to dedicate for the web application and test the speed online.

The frontend interacts with the backend by making API calls with GET requests every time it needs information. For example, to determine which tickers it should query, the main page will request a list of tickers. Furthermore, all the semantic similarity calculations are performed in the backend and are returned as a JSON formatted list. Similarly, the knowledge graph is filtered and processed in the flask backend, and sent as a JSON object of nodes and edges.

6.2 Detailed Knowledge Graph

This detailed knowledge graph serves as a Python application that can be used to build, populate, query, and export our knowledge graph. The program uses a Python library called rdflib to store the nodes and edges in an RDF graph. This allows the graph to be easily stored and exported into different formats. The application also keeps track of commonly used nodes (commonly referred to as objects) and edges (commonly referred to as predicates) using Python dictionaries for consistency.

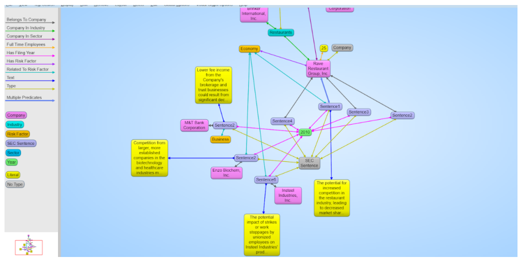


Figure 7: Detailed Knowledge Graph

6.2.1 Common Node Types and Predicates

In an RDF graph, the user can define nodes (objects) as a specific type of object. These are common node types that we decided to use and can be used when querying.

```
streetAddress, city, state, country, company,
sector, industry, riskFactor, year, month,
day, SECSentence, industry, sector,
longBusinessSummary, fullTimeEmployees,
auditRisk, boardRisk, compensationRisk,
shareHolderRightsRisk, overallRisk, symbol
```

Furthermore, a user can also define the name of edges, which is the relationship one node has to another (also called a predicate). These are common predicates that are frequently used and are useful when querying.

```
RDF.type, hasState, belongsToCountry, hasCity,
belongsToState, hasAddress, belongsToCity,
isYear, isMonth, text, relatedToRiskFactor,
belongsToCompany, hasRiskFactor,
companiesWithRiskFactor, hasFilingYear,
hasSECSentence
```

It is important to note that both of these lists can easily be expanded on in the future.

6.2.2 Schemas in the Knowledge Graph

The two most important schemas in the knowledge graph are for the object company and SECSentence. Below are the schemas for those objects, with each line containing a name and the associated predicate.

Company

- Object Type: RDF.type
- Ticker: symbol
- Location of HQ: headquarteredIn
- Long Business Summary: longBusinessSummary
- Full-time Employees: fullTimeEmployees
- Audit Risk (1-10, low-high): auditRisk
- Board Risk (1-10, low-high): boardRisk
- Compensation Risk (1-10, low-high): compensationRisk
- Shareholder Rights Risk (1-10, low-high): shareHolderRightsRisk
- Overall Risk (1-10, low-high): overallRisk
- Industry: industryKey
- Sector: sectorKey
- SEC 10-K Filing Sentence [Optional]: hasSECSentence
- Risk Factors [Optional]: hasRiskFactor

SECSentence

- Object Type: RDF.type
- SEC Sentence: text
- Filing Year: hasFilingYear
- Related Risk Factor: relatedToRiskFactor
- Related Company: belongsToCompany

Similar to the node types and predicates, these schemas can easily be updated based on the structure that the code enforces.

6.2.3 Building and Populating the Knowledge Graph

The first functionality is building and populating the knowledge graph. We have developed two functions to accomplish that. The first allows the user to pass in a company ticker, which the program then takes to retrieve information from that company from the yfinance API. The information obtained from that API is used to create attribute edges for that company. This information includes things like different types of risk, the location of the company's headquarters, and the company's sector. The second function also allows the user to pass in a company ticker. It determines if the company already exists in the knowledge graph and adds it if it does not exist in the graph. Then, we use the provided ticker and extract its SEC 10-K filing sentences, its related risk factor, and the filing year that sentence is taken from our dataset. Using the same process in the semantic similarity analysis, the SEC 10-K filing sentences and related risk factors are used to connect companies to types of risk they might be vulnerable to, with the corresponding text segment that has the highest similarity score. The filing year of these sentences is used as an additional query parameter and provides a time context to the risk factors and sentences.

6.2.4 Querying and Exporting Knowledge Graph

The detailed knowledge graph library also has three types of built-in queries that a user can make: query companies that meet a condition, query a specific company by ticker, and query SEC-10K filing sentences that meet a condition. The first query type, allows the user to query companies that are in a certain sector, or industry, or have a specific risk factor. This returns a list of companies and their tickers that meet those criteria. The second query allows you to get more information on a company from their ticker and returns a list of the attributes–sector, industry, headquarters location, risk, etc. Finally, the third query finds all SEC-10K filing sentences that meet a condition and returns them as a list. These are not the only queries that can be done on the knowledge graph. Since the knowledge graph is stored in an RDF graph from rdflib which has a built-in function for queries, a user can also pass in custom queries. The library also has two functions to export the knowledge graph into various file formats such as Turtle, JSON-LD, and RDF/XML.

7 Contributors

The development of this toolkit was a collaborative effort, with each team member contributing based on their strengths and expertise. Listed below is each team member's contributions:

Rodz Andrie Amor

- **Baseline Implementation of the Toolkit:** The groundwork for the frontend and the backend which involved maintaining the repository, preparing the initial classes, connecting the pipelines and designing a REST API service for the web application. The initial version of sentiment analysis was also implemented.

- **Frontend Implementation:** Most of the features and functions added by other group members need to be implemented in the front end, along with presenting any results from these features

- **SEC 10-K Web Scraper:** The development of the BeautifulSoup web scraper, which can yield the risk section of companies in the form of CSVs

- **Website Knowledge Graph:** Developed the page that renders and builds the knowledge graph. Made functionality to allow you to click on a company or event node to get more information.

- **Documentation:** Documenting parts of the process for other team members along with partially writing, revising, and submitting the required deliverables

Sash Lamba

- **Dataset Pipeline:** The pipeline for the events data which involves parsing and cleaning the risk section, using the GPT API to parse events, and filtering the events to get accurate data.

- **Sentiment Analysis Metrics:** Multiple metrics are used by the toolkit to find connections between risk text and events. The implementation of metrics such as Jaccard, TF-IDF, and Bert along with another variation of cosine similarity.

- **GNews Scraper:** A custom Gnews scraper which utilizes the common HTML and RSS web scraping techniques to convert queried headlines into a dictionary of real-world news events

- **Documentation:** Writing, revising, and submitting required deliverables.

Keshav Ganapathy

- **Frontend Features:** Building flask API endpoints to pull stock information for finance, creating a visualization of stock prices, and several other front end features

- **Label Generation:** Exploring label generation with the help of Dolly, GPT, Ollama and other open source LLMs, leveraging the same in the pipeline to label events. Prompt engineered to ensure better results of event recognition.

- **Data Pipeline:** Designing the initial pipeline for more fine-grained weather events, adding the ability to filter by date to the

- **Documentation:** Writing and revising slides for presentations, and deliverables.

The-Vinh Calix Tran-Luu

- **Preliminary Research:** Researched and tested News APIs to obtain news to build a dataset; also researched the concept of knowledge graphs and tested different graph APIs to determine what would fit our requirements for a knowledge graph.

- **Knowledge Graph:** Developed the knowledge graph library to build, populate, query, and export knowledge graphs; also built a command-line interface for the library and a program to obtain and extract necessary data to populate the knowledge graph.

- **Documentation:** Contributed to writing and revising deliverables and presentations; wrote documentation for the knowledge graph library, including schemas and functions.

Sakshi Gholap

- **Frontend Implementation:** Implemented features including coloring, CSV download, and table information.

- **GUI and Visualizations:** Developed graphical user interface and visualizations for better user interaction and data representation.

- **Initial Flask Application:** Created initial Flask application endpoints and the first static website.

- **Google News Scraper:** Used a Google News scraper to generate labels for weather events in a Kaggle dataset.

- **Label Entry Coloring:** Added coloring for label entries to signify similarity.

8 Future Work

8.1 Web Application

Moving forward, there are several avenues for enhancing the functionality and user experience of our web-based toolkit. One potential area of improvement lies in the integration of real-time data feeds. By incorporating live data sources such as social media feeds, financial news websites, or specialized APIs, users can access up-to-the-minute information on events and financial data, thereby enabling more informed decision-making in response to rapidly evolving market conditions.

Additionally, another area could be to improve the visualization capabilities of the toolkit to provide users with more interactive and insightful representations of data. Particularly, it may be substantially useful to get a detailed knowledge graph on the web application, instead of one limited by only events and companies. These improvements would help users to gain a clearer understanding of complex financial relationships and identify emerging risks with greater certainty.

Finally, we believe the most significant improvement to the project would be an improvement in the events dataset. Evidently, the dataset contains a great deal of boilerplate information that is repeated throughout various companies. These events should be filtered out as they are not useful in providing any insight into the company's risk.

8.2 Detailed Knowledge Graph

Knowledge graphs typically contain a wide range of relevant information that can be queried for many uses. Our current iteration only contains companies and their attributes, SEC 10-K filing sentences, and risk factors. However, there is still additional information that could be beneficial to it. In the future, it would be helpful to add specific instances of events (such as Hurricane Katrina, for example). You could also add more company attributes in the future such as historical dividend yields, financial statements, and company leadership. Our current iteration also does not support many companies and years of SEC 10-K filings. Building and improving this would provide the user with more information and how risks may shift over time. In addition to than adding more information to the knowledge graph, the library could benefit from additional built-in query functions to assist people who are less familiar with querying. Currently, there is not a function to query companies by attributes like different types and values of risk, or location. Implementing these functions might be helpful for investors in evaluating companies and their vulnerabilities to local events. Fortunately, the infrastructure to incorporate these changes already exists in the library, the only challenge is obtaining and formatting the data into a usable format.

References

[1] Saumya Bhadani, Ishan Verma, and Lipika Dey. Mining financial risk events from news and assessing their impact on stocks. In *Mining Data for Financial Applications: 4th ECML PKDD Workshop, MIDAS 2019, Würzburg, Germany, September 16, 2019, Revised Selected Papers 4*, pages 85–100. Springer, 2020.

[2] Sola Shirai, Debarun Bhattacharjya, and Oktie Hassanzadeh. Event prediction using case-based reasoning over knowledge graphs. In *Proceedings of the ACM Web Conference 2023*, pages 2383–2391, 2023.