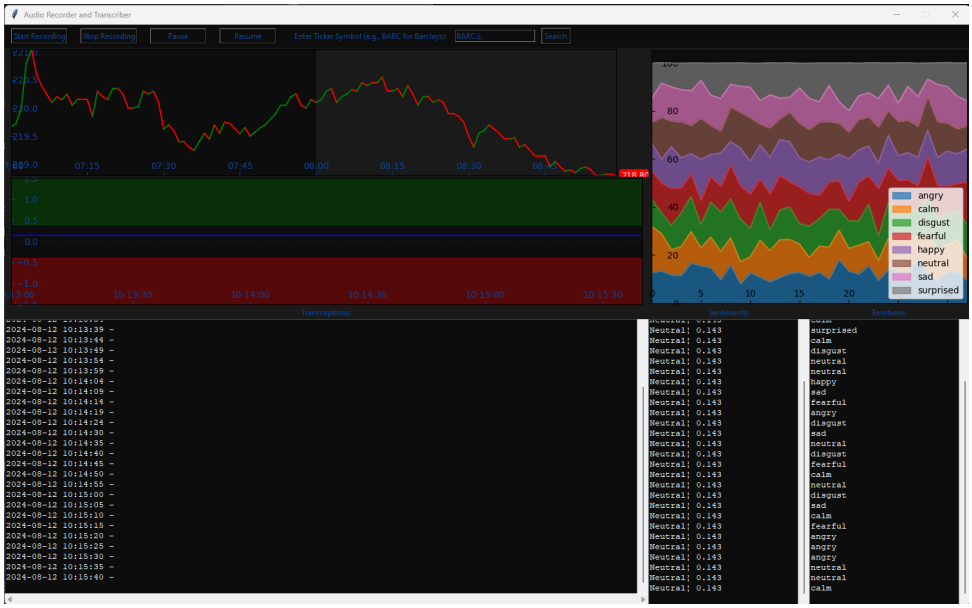
Real-Time Text Sentiment Analysis and Speech Emotion Recognition For Financial Markets



Emmanuel O. Olaoye

2319416

A thesis submitted for the degree of

Master of Financial Technology (Computer Science)

Supervisor: Dr. Maria Kryopoulou

School of Computer Science and Electronic Engineering

University of Essex

August 2024

Abstract

This dissertation presents the development and implementation of a real-time sentiment analysis and trading system that integrates text sentiment and speech emotion recognition to help enhance decision-making in financial markets. The project leverages advanced deeplearning Transformer models, including FinBERT for sentiment analysis and Wav2Vec for speech emotion recognition, to analyze audio data from financial sources such as conference calls, press meetings, and news broadcasts. By combining sentiment analysis with emotion detection, the system provides a more nuanced understanding of market sentiment, leading to more informed and timely trading decisions.

The system's innovative approach includes real-time transcription and analysis of spoken content, offering a dual-layered analysis of both the textual and emotional components of financial communications. This dual analysis improves the precision of trading signals, providing traders with actionable insights derived from both what is said and how it is said.

The dissertation highlights the challenges encountered, including achieving high accuracy in real-time processing, and discusses the system's robust performance in live market conditions. The work contributes to the ongoing advancement of FinTech by demonstrating how AI and machine learning technologies can be applied to enhance traditional trading strategies, paving the way for future innovations in algorithmic trading.

Contents

1 Introduction

1.1 Introduction

In recent years, the integration of Artificial Intelligence (AI) and Machine Learning (ML) into financial markets has revolutionized the way trading strategies are developed and executed. This dissertation focuses on the development and implementation of a real-time sentiment analysis trading system that incorporates speech emotion recognition to enhance decision-making in financial markets. The project, undertaken as part of the Master of Science in Financial Technology, aims to explore and extend the application of AI in the financial domain, particularly in the context of sentiment and emotion-driven trading strategies.

This project stems from the convergence of interests in computer science and financial markets. The rapid evolution of financial technology (FinTech) has demonstrated that AI and ML can be effectively leveraged to interpret market sentiment, which plays a critical role in shaping market movements and investor behaviour. Traditional financial analysis has relied heavily on quantitative metrics such as earnings reports and stock prices; however, with advancements in natural language processing (NLP) and sentiment analysis, there is a growing interest in incorporating qualitative data into trading models.

Courses such as "Computational Market Micro-structure for FinTech and the Digital Economy" and "Computational Models in Economics and Finance" provided a solid foundation for understanding semantic trading and computational methods in finance, sparking an interest in developing a system that can analyse real-time audio from financial sources to predict market sentiment and make informed trading decisions.

The primary goal of this project is to create a comprehensive application capable of processing and interpreting market sentiment in real-time. The system is designed to transcribe and caption audio from various financial sources, including conference calls, press meetings, and news broadcasts, and use this information to predict market sentiment and guide trading decisions. To enhance the accuracy of these predictions, the project integrates speech emotion analysis, allowing the system to detect and quantify the emotional tone of spoken content. This dual analysis of sentiment and emotion aims to improve the precision of trading signals, leading to more informed and timely trading decisions.

In summary, the contributions of this paper are fourfold:

- Creating a system that can transcribe audio, predict sentiment, and analyze speech emotion concurrently.
- Enhancing the sentiment analysis model and incorporating speech emotion recognition using Transformer models.
- Design an intuitive interface that caters to both novice and experienced traders, simplifying the trading process and providing actionable insights.
- Demonstrate how advanced AI and ML techniques can be applied to traditional trading practices, contributing to ongoing advancements in the FinTech sector.

The dissertation delves into the literature surrounding the integration of emotion recognition and sentiment analysis in financial modelling. It highlights recent developments in the field, including studies on text sentiment analysis, emotion detection in financial decision-making, and multi-modal sentiment analysis that integrates text, audio, and visual data. The project builds on these studies by focusing on real-time application and incorporating speech emotion recognition into the analysis process.

The methodology involves the use of advanced Deep Learning models such as FinBERT for sentiment analysis and Wav2Vec for speech emotion recognition. These models were selected for their high accuracy and suitability for financial data. The project also explores the challenges associated with real-time processing, including the need for optimization and the importance of model interpret-ability.

The developed system successfully integrates real-time audio transcription, sentiment analysis, and speech emotion recognition, offering a novel approach to sentiment-driven trading strategies. The system's ability to process live data and return its sentiment and emotion marks a significant advancement in the application of AI to financial markets. Despite the challenges encountered, particularly in achieving high accuracy in speech emotion recognition, the final implementation demonstrates robust performance and holds promise for further development and refinement.

This dissertation contributes to the field of FinTech by showcasing how emerging AI technologies can be leveraged to trading insights, paving the way for more sophisticated and adaptive trading platforms in the future.

1.2 Background Knowledge

Before embarking on the investigation and development of this dissertation, I already had a strong background in software development. Knowing that ML/AI techniques would be integral to the project, I decided to develop the application in Python, a language commonly used to develop ML models and which I am most proficient and comfortable with. As mentioned in my motivations and goals, I was particularly inspired by the master's courses "Computational Market Micro-structure for FinTech [2] and the Digital Economy and Computational Models in Economics and Finance" [1] These courses provided me with a solid foundational understanding of semantic trading and computational methods in finance.

Similarly, my experience from my Undergraduate Final Year Project [3] made me well-equipped to develop a practical application, utilizing source control tools such as GitLab and GitHub, as well as project management tools like Jira. However, this project required me to handle large streams of audio for further processing, an area with which I was unfamiliar. To prepare, I began researching possible solutions and learning from tutorials by "Dystopian Dev" [6] and "All About AI" [7]. These tutorials enabled me to build an early mock-up and demo of the live captioning frameworks that would be integral to the application.

In the realm of ML/AI, I had some knowledge of refining and training CNN models and had been exposed to BERT and Transformer models during my undergraduate course in "Natural Language Processing.[4]" Further research led me to discover that some of their best and most suitable models are hosted on Hugging Face. Consequently, I began taking introductory courses on Linked In Learning [5] to deepen my understanding.

Finally, when I decided to implement speech emotion recognition into my project, I began researching previous studies on the subject. It was during this research that I encountered the vital work by [47], which I chose to extend in this dissertation. Understanding how they were able to build a classifier to predict future financial performance, I realized the necessity of developing my project in a way that could be extended to achieve real-time predictive capabilities.

2 Motivation and Project Goals

My interest in computer science and financial markets has been longstanding. Initially, I focused on computer science, developing apps and projects. However, during my master's courses "Computational Market Microstructure for FinTech and the Digital Economy" [1] and "Computational Models in Economics and Finance," [2] I discovered the profound connection between these fields. I realized that computer science skills could be effectively applied to finance.

This realization sparked my fascination with algorithmic trading, particularly sentiment trading—the practice of making trades based on market sentiment. I learned how proprietary trading bots analyse sentiment from news events or public figures, quantify the prices of securities and commodities, and execute trades accordingly [25]. Inspired by these insights, I decided to explore sentiment analysis for real-time trading, finding that while sentiment analysis has been successfully applied to fundamental trading, real-time analysis remained under-explored.

My goal was to develop an application that facilitates real-time trading by analysing audio from areas such as conference calls, press meetings, and news broadcasts. This application needed to transcribe and caption audio accurately, predict sentiment from the text, and analyse speech emotion concurrently. Additionally, I hoped to add features that could also assess specific stocks and execute trades based on these analyses.

2.1 Initial Project Goals

Development of a Real-Time Sentiment Analysis Trading System: Create an application that processes and interprets market sentiment in real time. This involves developing a system that can transcribe and caption audio from various sources—such as conference calls, press meetings, and news broadcasts—and use this information to predict market sentiment and guide trading decisions.

Integration of Speech Emotion Analysis: Incorporate speech emotion analysis into the system. Develop algorithms to detect and quantify the emotional tone of spoken content, enhancing sentiment analysis and improving the accuracy of trading signals.

Real-Time Trading Execution: Implement a bot that can execute trades based on real-time sentiment and emotion analysis. Ensure the bot can evaluate stock performance and make informed trading decisions promptly, leveraging insights from the audio analysis.

User-Friendly Interface and Accessibility: Design an intuitive and user-friendly interface for the trading application that serves both novice and experienced traders. Ensure the platform simplifies the trading process and provides actionable insights based on real-time analysis.

Contribution to FinTech Innovations: Advance the field of financial technology by integrating innovative sentiment analysis and emotion detection techniques with trading applications. Showcase how these technologies can enhance traditional trading practices and drive progress in the FinTech sector.

3 Literature review

3.1 Introduction

The integration of emotion recognition and sentiment analysis into financial modelling has become an emerging research area, offering fresh insights into market behaviours and corporate financial performance. Traditional financial analysis has primarily relied on quantitative metrics, such as earnings reports and stock prices. However, advancements in big data, machine learning, and natural language processing have opened new opportunities to incorporate qualitative data, especially through the analysis of text sentiment and speech emotion. This literature review synthesizes recent developments in this interdisciplinary field, highlighting studies that explore how these non-traditional indicators can enhance financial predictions and decision-making.

3.2 Related Literature

3.2.1 Speech Emotion Recognition in Financial Distress Prediction

A pivotal study by Hajek and Munk [46] investigates the use of speech emotion recognition and text sentiment analysis for predicting corporate financial distress—a critical area in financial analysis that helps identify firms at risk of bankruptcy. Traditional models, such as the Altman Z-score, rely on financial ratios but often overlook qualitative aspects of managerial communication, which can offer early distress signals.

Hajek and Munk developed a deep learning model that integrates emotional indicators from speech with conventional financial metrics. Their study analyzed 1,278 earnings conference calls from 40 major U.S. companies (from 2010–2021), utilizing CNN-based SER models and FinBERT for textual sentiment analysis. The findings demonstrate that managerial emotions captured through SER significantly contribute to predicting financial distress, often outperforming textual sentiment. This suggests that incorporating emotional and sentiment analysis can provide a more nuanced and accurate prediction of financial distress, thus offering valuable tools for investors and analysts.

3.2.2 Advancements in Text Sentiment Analysis and Emotion Detection

The broader field of text sentiment analysis has also seen significant progress. Lai Po Hung and Suraya Alias (2023) [8] review the evolution of text-based sentiment analysis, noting a shift from simple polarity detection to more sophisticated emotion detection techniques. Traditional sentiment analysis primarily involved keyword matching, which is limited in capturing the complexity of human emotions—especially sarcasm, irony, and context-dependent meanings.

Recent advancements in machine learning and deep learning, particularly models like RNNs and CNNs, have improved the accuracy of sentiment and emotion detection by capturing contextual information and temporal dependencies in text. Pre-trained language models such as BERT and its financial variant FinBERT have further enhanced the ability to perform accurate sentiment classification by learning contextual relationships between words.

3.2.3 Emotion Detection in Cryptocurrency Markets

Sentiment and emotion detection have also been applied to the analysis of cryptocurrency markets, where public sentiment significantly influences market volatility. Naila Aslam and colleagues (2022) [10] explored sentiment and emotion detection in cryptocurrency-related tweets using an ensemble model combining LSTM and GRU

networks. The study found that this ensemble model outperformed traditional machine learning approaches in both sentiment analysis and emotion detection, highlighting its potential to provide reliable insights for investors in the volatile cryptocurrency market.

3.2.4 Multi-modal Sentiment and Emotion Analysis

Beyond text and audio, there is growing interest in multi-modal sentiment and emotion analysis, which integrates multiple data sources, such as text, audio, and visual content. Soujanya Poria and colleagues (2018) [11] introduced a multi-modal framework that leverages a combination of CNNs and RNNs to analyse sentiment and emotions in videos. Their approach, validated with benchmark datasets, significantly outperformed state-of-the-art methods, particularly when applied to video data, demonstrating the effectiveness of integrating diverse data sources for sentiment and emotion classification.

3.2.5 Emotion Detection in Financial Decision-Making

Research has also explored the role of emotion detection in financial decision-making processes. the researchers [12] examined the influence of emotions like ”rejoice” and ”regret” during trading decisions. Their study used physiological signals to measure emotions and found that ”blended” emotion models, which account for multiple emotions simultaneously, provided a more nuanced understanding of participants' emotional states. The findings suggest that integrating physiological data into decision support systems can help users become more aware of their emotions, potentially leading to more informed and rational financial decisions.

3.3 Challenges and Limitations

Despite the promising results, several challenges remain in integrating speech emotion recognition and text sentiment analysis into financial modelling. One significant challenge is the real-time application of these techniques. While some studies could theoretically adapt their methods for real-time processing, the focus has primarily been on improving accuracy and integrating multi-modal data. Achieving real-time capabilities would require further optimization, particularly in reducing latency in data processing and model inference.

Another challenge is the interpretability of the models used. Deep learning models, while powerful, often function as ”black boxes,” making it difficult for analysts to understand the decision-making process behind specific predictions. Future research could focus on developing interpretable models that provide clear explanations of their predictions, thereby increasing their trustworthiness and usability in real-world financial applications.

3.4 Case Studies and Applications

3.4.1 2013 Twitter and the AP News Hack

In April 2013 [13], hackers gained access to the Associated Press (AP) Twitter account and posted a false tweet claiming explosions at the White House had injured President Obama. This misinformation caused a brief but significant drop in the stock market, with the Dow Jones Industrial Average falling by about 150 points within minutes before quickly rebounding once the tweet was verified as false. Sentiment analysis tools could have played a crucial role in preventing this market panic by monitoring social media and news sources for signs of emerging misinformation. By analyzing the sentiment around news events and social media posts, traders and investors could have better assessed the credibility of the information and responded more appropriately.

3.4.2 The Brexit Referendum

The 2016 Brexit referendum, where the United Kingdom voted to leave the European Union, led to substantial volatility in financial markets. The results caused significant fluctuations in currency values, particularly the British pound, which fell sharply against major currencies [14]. Advanced sentiment analysis of social media, news articles, and public statements in the lead-up to the referendum could have provided insights into public sentiment and potential market movements. This information might have helped investors and financial institutions better prepare for or mitigate the impact of the referendum results.

3.4.3 2014 Alibaba IPO

The initial public offering (IPO) of Alibaba Group in September 2014 was one of the largest in history. Despite the overwhelming interest and initial success, there were fluctuations in the stock price [15] and ongoing discussions about the company's long-term prospects. Sentiment analysis of news articles, analyst reports, and social media chatter during and after the IPO could have provided valuable insights into investor sentiment and market perceptions about Alibaba's future performance. This could have helped investors make more informed decisions and gauge the stock's potential volatility.

3.4.4 GameStop Short Squeeze

In January 2021, the stock of GameStop, a video game retailer, experienced an extraordinary surge in price due to a short squeeze driven by retail investors on platforms like Reddit. This phenomenon led to significant financial losses for some institutional investors and raised questions about market dynamics [16]. Sentiment analysis could have been used to monitor discussions on forums, social media, and trading platforms to identify early signs of coordinated trading activity and potential short squeezes. By analysing sentiment trends and the volume of discussions, financial analysts could have detected unusual market behaviour and adjusted their strategies accordingly.

3.4.5 The COVID-19 Pandemic Market Reactions

The onset of the COVID-19 pandemic in early 2020 caused widespread market turbulence and uncertainty [17]. Financial markets reacted sharply to news about the pandemic, government responses, and vaccine developments. Sentiment analysis of news articles, public health updates, and social media discussions could have provided real-time insights into market sentiment and public reaction to the pandemic. This would have enabled investors to better understand market dynamics and adjust their investment strategies in response to rapidly changing information.

3.4.6 2008 Financial Crisis

The 2008 financial crisis, triggered by the collapse of Lehman Brothers and the subprime mortgage crisis, led to a severe global economic downturn and financial market turmoil [18]. Sentiment analysis of financial news, economic reports, and market commentary could have identified early signs of distress and negative sentiment regarding financial institutions and the housing market. This could have helped investors and policymakers anticipate and respond to the unfolding crisis more effectively.

3.5 Semantic Trading

3.5.1 Automation and Algorithmic Trading: A Historical Overview

The history of algorithmic trading represents a significant evolution in financial markets, beginning with the advent of electronic trading systems in the 1970s. A key milestone was the launch of the NASDAQ stock market in 1971 [19], which was among the first to use a computer-based system to facilitate trading. This innovation revolutionized the securities trading process and laid the groundwork for automated order-matching systems. During the 1980s and 1990s, advancements in computational power and the availability of historical market data enabled the development of more sophisticated trading algorithms. These algorithms were designed to execute large orders more efficiently by breaking them down into smaller transactions, minimizing market impact and optimizing execution prices.

During the 1980s and 1990s, advancements in computational power and the availability of historical market data enabled the development of more sophisticated trading algorithms. These algorithms were designed to execute large orders more efficiently by breaking them down into smaller transactions, minimizing market impact and optimizing execution prices.

The early 2000s saw the rise of high-frequency trading (HFT), a major leap forward in algorithmic trading. HFT leverages cutting-edge technology and high-speed data connections to execute trades within microseconds, exploiting minute price discrepancies across various markets and securities [20]. This practice became widespread due to advancements in communication technologies and the deregulation of financial markets. However, the 2008 financial crisis and events like the "Flash Crash" of 2010 [21] highlighted both the strengths and vulnerabilities of algorithmic trading, emphasizing the need for robust risk management and regulatory oversight.

3.5.2 Semantic Trading: A Historical Overview

Semantic trading, also known as text-based financial forecasting, is an emerging paradigm that builds on the foundations of algorithmic trading by incorporating semantic technologies to enhance the understanding and processing of financial data. The concept originates from the broader field of semantic computing, which aims to enable machines to understand and interpret data contextually, rather than merely processing it syntactically.

The roots of semantic trading can be traced back to the early 2000s when researchers began exploring ways to integrate semantic web technologies into financial trading systems. The semantic web, an extension of the World Wide Web, seeks to create a more intelligent and interconnected web where data is machine-readable and can be processed meaningfully by computers. By leveraging ontology, metadata, and other semantic tools, semantic trading systems aim to improve the accuracy and relevance of information used in trading algorithms.

A significant advancement occurred in the late 2000s and early 2010s with the integration of natural language processing (NLP) and semantic analysis into trading platforms. These technologies allowed trading systems to process and understand unstructured data sources—such as news articles, social media posts, and financial reports—in a way that mimics human comprehension. For example, one study proposed [23] using TF-IDF (Term Frequency - Inverse Document Frequency) to represent news headlines from 20 newsgroup datasets. These were then used as input to a support vector machine (SVM) model to predict stock movements [23]. However, the Bag of Words (BoW) approach, which was initially employed, had limitations due to its inability

to capture semantic relationships between words, as it treated each word independently. To address this, the n-gram feature was introduced. Unlike BoW, which extracts individual words, the n-gram approach captures contiguous sequences of n words from a text, thereby better preserving syntactic relationships between words. Research showed that bi-gram features outperformed both trigram and unigram features when evaluated using ad hoc announcement data [24].

As artificial intelligence (AI) and machine learning (ML) technologies continued to evolve, the concept of semantic trading expanded further [25]. Modern semantic trading systems combine AI, ML, and semantic technologies to analyze vast amounts of data from diverse sources, providing traders with a deeper, context-aware understanding of market conditions. These systems are designed to identify complex patterns and correlations that might be missed by traditional algorithms, offering a more nuanced approach to trading.

For instance, the FinALBERT model [26] was proposed to leverage and fine-tune the pre-trained FinBERT models for stock movement prediction. This approach exemplifies the cutting-edge of sentiment-based stock forecasting, as summarized in the findings of various studies.

Semantic trading represents a convergence of technological advancements in AI, ML, and semantic computing, paving the way for more intelligent, adaptive, and context-aware trading strategies. As this field continues to develop, it is poised to play an increasingly significant role in the future of algorithmic trading.

3.6 Text sentiment in finance

Sentiment analysis (also known as opinion mining) is the method of using computer software through the use of natural language processing to find out people's opinions or feelings about something [28]. In this project, the sentiment was defined as a positive, neutral or negative feeling.

3.6.1 Dictionary-Based Methods

Also known as the Lexicon-based approach involves deriving the sentiment of a topic by taking a predefined list of words (sentiment lexicons) where each of these words is labelled with a sentiment (positive neutral, and negative) and summing up their appearance to return a sentiment score. However certain rules are applied when assessing the sentiment of a lexicon to handle certain cases such as negations. For example, in "never amazing" in which "amazing" is a positive statement adding "not" negates the phase making it negative. Similarly, rules can also mean "very amazing" becomes even more positive

For this project VADER (Valence Aware Dictionary and sEntiment Reasoner) and TextBlob were chosen as one of the auxiliary sentiment analysis models

3.6.2 Machine Learning-Based Methods

Using supervised learning (sl), labelled data is used to train a classifier. Naïve Bayes predicts the sentiment by finding the probability of words occurring in each sentiment category. Support Vector Machines (SVM) find the best boundary (or hyperplane) that separates different sentiment categories (ie positive or negative.) Logistic Regression is a classification algorithm that predicts binary outcomes using a linear regression equation [27]. Unlike linear regression, it employs a Sigmoid function as its cost function, which produces an S-shaped curve, limiting predictions to values between 0 and 1. This curve is also known as the logistic function. Feature

Engineering involves using such as Bag-of-Words (BoW), Term Frequency-Inverse Document Frequency (TF-IDF) and n-grams

3.6.3 Deep Learning-Based Models

Deep learning-based models have become highly effective tools in sentiment analysis, particularly for capturing complex patterns and nuances in text. Recurrent Neural Networks (RNNs), especially Long Short-Term Memory (LSTM) networks, are widely used due to their ability to process sequential data and maintain context over time, which is crucial for understanding sentiment in sentences. Convolutional Neural Networks (CNNs) are often employed alongside RNNs to extract and highlight key features from the text, enhancing the model's understanding of sentiment-related cues. More recently, transformer-based models like BERT [9] and GPT have revolutionized sentiment analysis by utilizing attention mechanisms that allow the model to focus on relevant parts of the text, enabling a more accurate interpretation of sentiment, even in complex and context-dependent cases.

For this project Bert [9] models FinBERT was chosen as the main Sentiment analysis model and roBERTA as an auxiliary model.

3.6.4 Domain-Specific Sentiment Analysis in Finance

In the context of financial technology, using general or non "domain-specific" models may misrepresent or misinterpret nuanced language or context-specific meanings that are unique to a particular domain, such as finance, healthcare, or customer reviews. For instance, in the financial sector, statements such as "Stocks rallied and the British pound gained." or a more specific one such as " The market experienced a slight correction due to Jerome Powell's FOMC address an hour ago." [29] A generic sentiment analysis model might not fully grasp the financial terminology. It could interpret "rallied" and "gained" as neutral or even negative terms if it associates "rallied" with social gatherings or protests and "gained" with weight or something unrelated to finance. This might lead the model to either incorrectly assign neutral sentiment or misinterpret the context, providing an inaccurate sentiment score. in the case of TextBlob and NLTK models which both returned a polarity of 0.00 meaning they were classified as neutral. A domain-specific model, trained on financial data, would recognize that "rallied" refers to a significant upward movement in stock prices and that "gained" indicates an increase in the value of the British pound. As in the case of FinBERT which classified it as positive with a polarity of 0.554.

Domain-specific models are trained on data that reflects the language and sentiment patterns of that particular field, leading to more accurate and relevant insights. By tailoring the model to understand the context and subtleties of a specific domain, businesses and researchers can achieve more reliable sentiment analysis, resulting in better decision-making and more targeted strategies.

3.7 Speech Emotion

3.7.1 Mel-frequency cepstral coefficients

Mel-frequency cepstral coefficients (MFCCs) are a set of features used in audio signal processing [30], particularly in the analysis and recognition of speech. They are derived from the Mel-frequency scale, which approximates the human ear's response to different frequencies, making them particularly effective for speech-related tasks.

3.7.2 MFCC Calculation

Mel-frequency cepstral coefficients (MFCCs) are calculated through a series of steps that transform the raw audio signal into a compact representation that highlights perceptually important aspects of the sound.

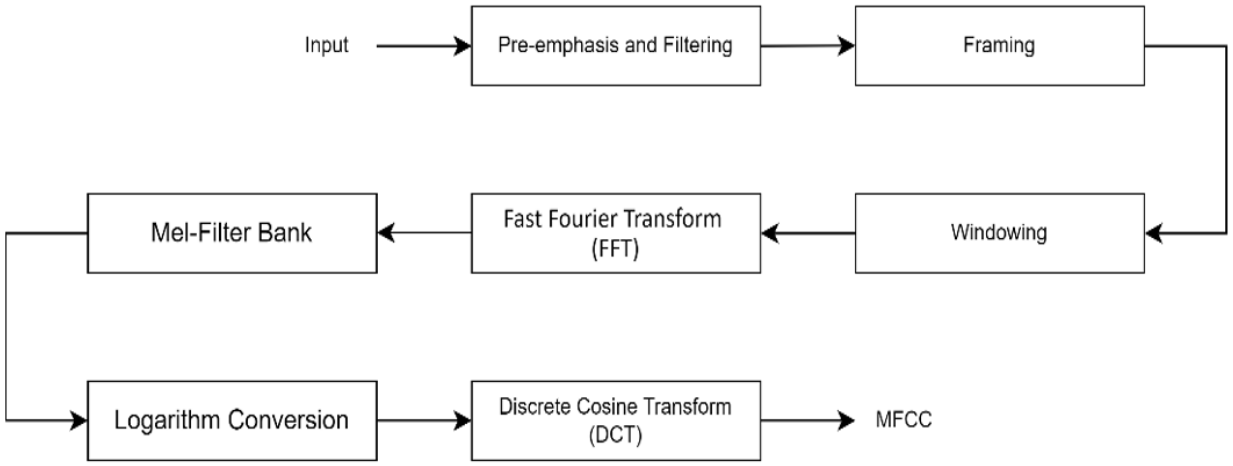


Figure 1: MFCC Sprectrogram Process

1. Pre-emphasis and Filtering The first step involves passing the audio signal through a pre-emphasis filter. This filter amplifies the higher frequencies in the signal, compensating for the natural tendency of speech signals to have higher energy at lower frequencies.

$$y(t) = x(t) - a * x(t-1) \tag{1}$$

Here, $y(t)$ is the output signal, $x(t)$ is the input signal, and a is typically set to a value around 0.95.

2. Framing The continuous audio signal is divided into short overlapping frames to capture the temporal changes in speech.

3. Windowing Each frame is then multiplied by a window function, typically a Hamming window, to reduce spectral leakage. This step ensures that the edges of the frames taper smoothly to zero.

$$w(n) = 0.54 - 0.46 * cos(\frac{2\pi n}{(N-1)}) \tag{2}$$

Here, N is the number of samples in a frame, and n is the sample index.

4. Fast Fourier Transform (FFT) The windowed frames are transformed from the time domain to the frequency domain using the Fast Fourier Transform (FFT). This step converts the time-based signal into a spectrum of frequencies.

A power spectrum that represents the signal's energy at various frequency components.

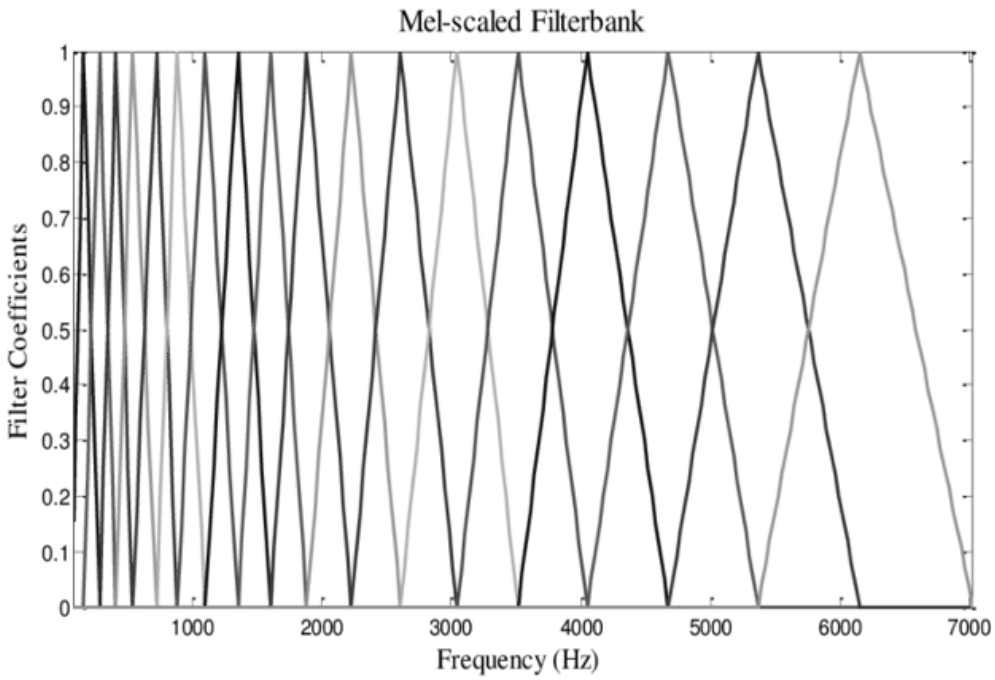


Figure 2: Mel filter banks basis functions using 20 Mel-filters in the filter bank [33]

5. Mel-Filter Bank The power spectrum is then passed through a series of triangular band-pass filters, known as Mel-filters, spaced according to the Mel scale (see Figure 2). The Mel-scale is a perceptual scale of pitches judged by listeners to be equal in distance from one another.

$$m = 2595 * log_{10}(1 + \frac{f}{700}) \tag{3}$$

These filters emphasize the frequencies that are more important for human hearing and produce a set of log energies for each filter.

6. Logarithm Conversion The logarithm of the energy output from each mel-filter is computed. This step helps in compressing the range of the power spectrum and making it more suitable for human perception.

7. Discrete Cosine Transform (DCT) The final step is to apply the Discrete Cosine Transform (DCT) to the log filter-bank energies. The DCT decorrelates the filter-bank energies and compacts the information into a small number of coefficients.

The first few coefficients (typically 13) are retained as the MFCCs, while the rest are discarded.

8. Additional Steps (Optional)
Liftering: Sometimes, a liftering operation is applied to the MFCCs to smooth them, enhancing the modelling process.

Delta and Delta-Delta Coefficients: Sometimes, a liftering operation is applied to the MFCCs to smooth them, enhancing the modeling process.

MFCCs are widely used as a feature set to capture the nuances of speech that may correlate with emotional states. Emotions can influence various aspects of speech, such as pitch, tone, and energy [31] [**?**], which are reflected in the MFCCs. By analysing the patterns and variations in these coefficients, machine learning models can be trained to recognize and classify emotions like happiness, anger, sadness, or surprise from spoken audio.

3.7.3 Reasoning for Mean Calculation

In some investigated and adopted models, the MFCC mean was calculated. As explained by [30], Given the large number of coefficients for each frame, if an audio input has a large number of frames, the extracted coefficients can result in a substantial data set. To simplify and condense this data, the mean of these MFCCs is calculated. This process involves averaging the MFCC values across all frames for each coefficient. Specifically, if 13 MFCCs are extracted per frame and averaged over time, the result is 13 mean values that summarize the entire audio segment. This approach significantly reduces the data size while still retaining the essential features of the original signal.

3.7.4 Chromagram

Chromagram features, also known as chroma features or pitch class profiles, are a set of audio features used in signal processing and music information retrieval. These features are derived from the chroma, which refers to the twelve distinct pitch classes in the Western music scale (i.e. C, C#, D, D#, E , F, F#, G, G#, A, A#, B), regardless of octave. Chromagram features capture the intensity or energy present in each of these pitch classes over time [35].

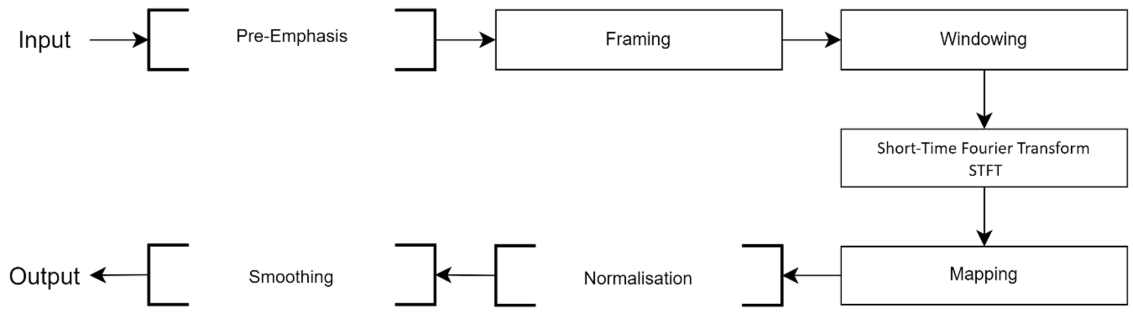


Figure 3: Chromagram feature extraction process

3.7.5 Chromagram Feature Extraction

1 - 3. Pre-processing the Audio Signal, Framing and Windowing

Similar to MFCC extraction, pre-processing, framing, and windowing are all performed to extract the chromagram features [34].

Pre-emphasis (Optional): Although not required, a pre-emphasis filter might be applied to the audio signal to boost higher frequencies. This step is sometimes used to enhance the clarity of the pitch-related features in the signal.

Framing the Signal: The audio signal is divided into short, overlapping frames. This allows for analysis of how the chroma features change over time, capturing the temporal dynamics of the signal.

Framing the Signal: The audio signal is divided into short, overlapping frames. This allows for analysis of how the chroma features change over time, capturing the temporal dynamics of the signal.

Windowing: Each frame is multiplied by a window function, usually a Hamming or Hann window, to taper the edges and minimize spectral leakage when transforming the signal to the frequency domain.

4. Short-Time Fourier Transform (STFT): The windowed frames are transformed from the time domain to the frequency domain using the Short-Time Fourier Transform (STFT). This transformation produces a spectrogram, which represents the signal's frequency phase content of the windowed cycle (i.e. the content across time.)

$$SIFT(x)(\tau, k) = X(\tau, k) = \sum_{n=0}^{N_s-1} x[n] \cdot h[n-\tau] \cdot e^{-\frac{j2\pi kn}{N_s}} \tag{4}$$

where:

- $X(m, k)$ is the STFT of the signal at time frame τ and frequency bin k
- $x[n]$ is the input signal
- $h[n]$ is the window function applied to the signal (e.g. Hamming or Hann window)
- N_s is the number of points in the FFT (Fast Fourier Transform).
- k is the frequency bin index, and m is the frame index.

5. Mapping to Pitch Classes To obtain a pitch representation, where notes or frequencies are present, the STFT spectrogram is mapped onto the 12 pitch classes of the chroma scale (such as C, C#, D, etc.). This process involves summing the energy of all harmonics that correspond to the same pitch class across different octaves. Specifically, for each frequency bin k in the spectrogram, the frequency $f(k)$ is mapped to one of the 12 pitch classes p, and the corresponding energy is added to the relevant pitch class. This process is repeated for all frequency bins and summed across octaves.

$$p(k) = 69 + 12 \cdot log_2(\frac{fk}{440}) \tag{5}$$

where $fk = \frac{k \cdot fs}{N}$, fs is the sampling rate and N is the of FTT points.

6. Energy Summation The spectral energy corresponding to each pitch class is summed across all octaves to produce a 12-dimensional vector for each time frame. This vector represents the intensity (or energy distribution across) of each pitch class in that frame.

$$C(m, c) = \frac{\sum}{p \in [0:127] | p \mod 12 = c} Y_{LF}(m, p) \tag{6}$$

where:

- $C(m, c)$ is the chroma feature vector at time frame m for chroma c

- $Y_{LF}(m, p)$ is the log-frequency spectrogram value for pitch p at the time m
- c is the chroma index, ranging from 0 to 11, corresponding to the 12 pitch classes in Western music (C, C#, D, etc.).
- $p \mod 12 = c$ ensures summation across all pitches p that belong to the chroma class c.

7 & 8. Normalization and Post-processing (Optional) [36]

Normalisation: To ensure consistency and remove the influence of amplitude variations, the chroma vectors can be normalized. One common approach is to normalize each vector so that its components sum to one, making the feature representation invariant to volume changes.

Smoothing Chroma Features: Smoothing the chroma features over time or aggregating them (e.g., computing mean or median chroma vectors over a segment) can help in capturing more stable harmonic patterns, especially in applications like music analysis or speech emotion recognition.

3.7.6 Reasoning Chromagram Extraction

Similarly, in speech emotion recognition, chromagram features can be used to capture pitch-related aspects of the speech signal. Emotions can influence the pitch and tonality of speech, and these changes are reflected in the chromagram features. By analysing the patterns in these features, machine learning models can be trained to recognize and classify different emotional states from spoken audio, just as they can with MFCCs.

3.8 Mel-Scaled Spectrogram

The Mel-scale Spectrogram [37] is used to align audio features with human pitch perception, emphasizing the importance of lower frequencies where the human ear is more sensitive. Unlike chromagram features, which focus on harmonic content and pitch classes, and MFCCs, which summarize spectral shapes into a compact set of coefficients, mel-scaled spectrograms provide a detailed frequency representation that is perceptually meaningful but less abstract than MFCCs. This makes the mel-scale particularly useful in tasks where perceptual accuracy across the frequency spectrum is crucial.

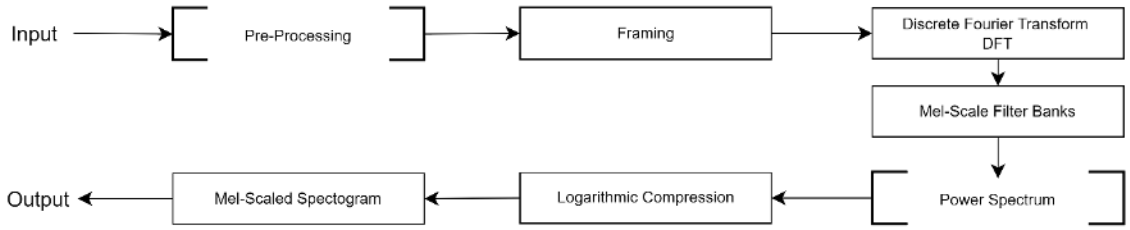


Figure 4: Mel-scale Spectrogram feature extraction process

3.8.1 Mel-Scaled Spectrogram Feature Extraction

1. Pre-emphasis (Optional) Similar to other audio feature extraction methods, the signal can be passed through a pre-emphasis filter to boost the higher frequencies, which tend to have less energy in speech signals. This step helps to balance the spectrum before further processing [37].

1. Pre-emphasis (Optional) Similar to other audio feature extraction methods, the signal can be passed through a pre-emphasis filter to boost the higher frequencies, which tend to have less energy in speech signals. This step helps to balance the spectrum before further processing.

2. Framing: Divide the Signal into Frames: The continuous audio signal is divided into short, overlapping frames. This allows the analysis to capture how the spectral content changes over time. Typically, frames are usually 20-30 milliseconds long, with an overlap between consecutive frames. Subsequently, each frame is multiplied by a window function (e.g., Hamming or Hann window) to minimize spectral leakage when converting the signal from the time domain to the frequency domain.

3. Discrete Fourier transform DFT: Compute the DFT: The windowed frames are then transformed into the frequency domain using the Discrete Fourier transform DFT. The DFT provides a spectrogram, which shows the magnitude of the signal at different frequencies over time.

4. Apply Mel Filter Bank: Compute the DFT: Mel Filter Bank: The spectrogram is passed through a bank of mel-scaled triangular filters. These filters are spaced according to the mel scale, which is a nonlinear scale that mimics the human ear's sensitivity to different frequencies. The Mel-scale filter-banks are computed as follows:

$$m = 2595 \cdot ln\left(\frac{f}{700} + 1\right) \tag{7}$$

where: $m$is the resulting Mel-scale and f is the frequency in the linear scale.

Now, each filter in the mel filter bank sums the energy of the frequencies within its range, producing a single value for each filter per frame. This effectively maps the linear frequency spectrum onto the mel scale.

Number of Filters: Typically, 20 to 40 mel filters are used, depending on the application. More filters provide finer frequency resolution, especially in the lower frequencies.

6. Power Spectrum (Optional): Compute Power Spectrum: The magnitude values obtained from the STFT can be squared to get the power spectrum before applying the mel filters. This step emphasizes stronger frequencies and makes the feature extraction more robust to noise.

7. Logarithmic Compression: The logarithm of the mel-filtered spectrogram values is taken to compress the dynamic range of the features. This step makes the mel-spectrogram more aligned with the human perception of loudness, where we perceive increases in intensity on a logarithmic scale rather than a linear one.

8. Generate the Mel-Scale Spectrogram: The resulting output is a mel-scale spectrogram, which is a time-frequency representation where the frequency axis is mapped according to the mel scale. The intensity of each point in this spectrogram corresponds to the log-magnitude of the signal at that mel frequency band and time frame. This can be expressed as:

$$S[m] = \log\left(\sum_{k=0}^{N-1} |x[k]|^2 Hm[k]\right) \tag{8}$$

4 Methodology

4.1 Graphic User Interface

The collection of classes from the provided files is designed to create a comprehensive GUI application that handles various functionalities, including audio recording, live stock price plotting, sentiment analysis, and emotion score visualization. Here's how they work together:

4.1.1 GUI Files

GUI and Event Handling(`gui_handler.py`):
This class is responsible for managing the overall user interface, including initializing frames, buttons, and text areas within the main application window. It creates and links together the visual components provided by other classes like LiveStockPlot, SentimentPolarityPlot, and EmotionScorePlot.

Audio Management(`audio_manager.py`):
This class manages audio recording and processing. It interacts with the UIHandler for UI-related operations and AudioHandler (presumably from an external module) for processing audio. It serves as the core that ties the audio functionality to the GUI.

Sentiment Polarity Plotting (`polarity_handler.py`):
This class handles the visualization of sentiment polarity over time. It uses Matplotlib to plot the data and is integrated into the GUI via UIHandler, allowing for real-time updates of sentiment analysis results.

Emotion Score Plotting (`emotion_score_handler.py`):
Similar to SentimentPolarityPlot, this class visualizes emotion scores over time, allowing for a stacked area plot representation of different emotions. It is also integrated into the GUI through UIHandler.

Live Stock Price Plotting (`live_stock_price_handler.py`):
This class fetches and plots live stock prices using data from sources like Yahoo Finance. It displays this data in the GUI and updates it periodically.

4.1.2 GUI Classes

`UIHandler` (from `gui_handler.py`) Manages the entire user interface, setting up buttons, frames, and integrating plots for different functionalities (e.g., sentiment, emotion, and stock price).

Key Functions:

- `setup_ui()`: Initializes the UI components, including setting up the main frame, buttons for controlling audio recording, and entry fields for stock symbols.
- `update_transcription_text()`, `update_sentiment_text()`, `update_emotion_text()`: Methods to update the displayed text in the GUI based on transcription, sentiment, and emotion data.
- `plot_stock()`: Triggers the plotting of live stock prices based on user input.

`AudioManager` (from `audio_manager.py`) Manages audio recording and integrates with the GUI for user interaction.

Key Functions:

- `__init__()`: Initializes the audio settings, creates instances of `UIHandler` and `AudioHandler`, and sets up the main GUI loop.
- Manages the state of the audio stream (e.g., starting, stopping, and pausing recordings).

`SentimentPolarityPlot` (from `polarity_handler.py`) Visualizes sentiment polarity over time using a line plot with areas shaded to indicate positive or negative sentiment.

Key Functions:

- `add_polarity_data()`: Adds new data points (`polarity` and `timestamp`) to the plot and updates the visualization.
- `update_polarity_plot()`: Re-renders the plot to reflect the latest data, adjusting visual elements like color and labels.

`EmotionScorePlot` (from `emotion_score_handler.py`) Visualizes emotion scores over time with a stacked area plot, allowing for a clear representation of multiple emotions at once.

Key Functions:

- `add_emotion_data()`: Adds new emotion data points to the plot and updates the visualization.
- `update_emotion_plot()`: Similar to the sentiment plot, this function redraws the plot based on new data, updating the GUI accordingly.

`LiveStockPlot` (from `live_stock_price_handler.py`) Fetches and displays live stock prices in a dynamic plot, showing price movements in real time.

Key Functions:

- `fetch_stock_data()`: Retrieves the latest stock price data using the `yfinance` library.
- `update_plot()`: Updates the plot with new data, applying color coding for price increases/decreases and managing the time axis
- `plot_stock()`: Starts the process of fetching and plotting stock data for a given stock ticker.

4.2 Audio Recording

4.2.1 Audio_handler

The audio_handler.py file is designed to manage real-time audio recording, processing, and analysis within a Python application. It handles the capture of audio streams, processes these streams into manageable chunks, and performs tasks such as speech-to-text transcription, sentiment analysis, and emotion recognition using multithreading for efficient operation. Additionally, it manages the storage and cleanup of audio data, ensuring that the system remains organized and responsive. The file integrates with other components of the application to update the user interface with real-time transcription and analysis results, making it a critical part of an audio-based processing system.

The audio recording functionalities served as a focal points of the whole program. The requirements of the projects were as follows:

1. The Recording initiation — `gui_handler.py`
2. Continuously save an audio segment (or chunk) — `audio_handler.py`
3. Send the audio segment to be processed — `audio_chunk.py`
4. Repeat — `gui_handler.py`
5. End the Recording — `gui_handler.py`

Imports and Dependencies: The audio_handler.py file imports essential libraries such as os, queue, threading, and wave for file management, multithreading, and audio processing. It also uses pyaudio for real-time audio input/output and custom modules like Json, chunk, and WhisperTranscriber for handling JSON operations, audio chunk processing, and speech-to-text transcription.

Class Initialization: The AudioHandler class initializes with references to a parent object, a PyAudio instance for audio input, and instances of WhisperTranscriber and Json handler for speech-to-text and data storage. It sets up a queue for audio chunks, a ThreadPoolExecutor for concurrent processing, and variables to track recording states. A dedicated thread monitors and processes the chunk queue in real time.

Audio Stream Handling: The `open_stream method` in `AudioHandler` configures and opens an audio stream using `PyAudio`, preparing the system to capture and process audio data continuously from an input device.

Queue and Chunk Processing: The class uses a `multithreaded` approach for processing audio chunks, with a thread managing the queue and `ThreadPoolExecutor` handling concurrent chunk processing. The `process_chunk` method extracts features like text transcription and sentiment analysis from audio chunks and updates the UI while storing processed data and deleting chunk files.

Recording Management: The class provides methods for managing the recording process, including starting, stopping, pausing, and resuming recording. During recording, audio frames are accumulated and saved as `.wav` files, which are then added to the processing queue. The class ensures clean termination by saving the entire recording and clearing the chunk queue.

File and Chunk Management: Methods like `save_chunk`, `delete_chunk`, and `clear_chunk_queue` handle the saving, deletion, and cleanup of audio files and chunks, maintaining system efficiency and organization.

Final Cleanup: The close method performs cleanup tasks by terminating the `PyAudio` instance and disposing of resources, ensuring the system remains stable after the recording session.

4.2.2 Audio_Manager

The `audio_manager.py` file controls audio management by initializing the AudioManager class, which sets up the GUI, configures audio settings, and links the interface with the audio processing handled by the AudioHandler class. It ensures real-time interaction between audio recording, transcription, and the user interface, allowing users to manage these tasks seamlessly through the application's GUI.

4.2.3 Audio_chunk

1. Imports and Dependencies
The audio_chunk.py file imports necessary libraries such as os, wave, and pyaudio for file handling and audio playback, along with custom modules like Text, get_grey_text, WhisperTranscriber, and HuggingFaceEmotionPredictor for transcription, sentiment analysis, and emotion prediction, setting up the tools needed for processing audio chunks.

3. Retrieving Chunk Information
The `get_chunk_filename` method retrieves the chunk file's full path, while `get_chunk_text_and_datetime` returns the transcribed text with a `timestamp` and `colour` code, allowing for efficient access and display of the chunk's information.

4. Sentiment Analysis Methods
The Chunk class includes methods like `get_chunk_finbert_sentiment`, `get_chunk_roberta_sentiment`, `get_chunk_nltk_sentiment`, and `get_chunk_textblob_sentiment` for extracting sentiment using various models, along with `get_chunk_finbert_polarity` for sentiment polarity, providing a thorough analysis of the chunk's emotional content.

5. Emotion Prediction
Emotion prediction in the `Chunk` class is handled by `get_emotions` and `get_chunk_emotion`, which use `HuggingFaceEmotionPredictor` to determine the emotions in the audio chunk and identify the highest-predicted emotion, adding depth to the analysis.

6. Processing the Audio Chunk
The `process_chunk` method combines transcription, sentiment analysis, and emotion prediction, returning a comprehensive overview of the chunk's content, including timestamp, sentiment, and emotions, centralizing the chunk's analysis.

7. File Management
The `Chunk` class handles file management with `delete_chunk` to remove processed files and `save_chunk` to store audio data.

4.3 Live Captioning

Whisper Transcriber

Faster Whisper is an optimized implementation of OpenAI's Whisper, a state-of-the-art automatic speech recognition (ASR) system. Whisper is known for its ability to transcribe speech to text with high accuracy across a variety of languages and dialects. However, one of the main challenges with the original Whisper model is that it can be computationally intensive and slow, especially when dealing with large models or on less powerful hardware.

The Faster Whisper model addresses this by making Whisper's transcription process faster and more efficient without significantly sacrificing accuracy. which involved optimizations like reducing the model size, leveraging more efficient computation techniques, and better hardware acceleration. These improvements make it more suitable for real-time applications or use on devices with limited computational resources.

The implementation came about from trying to research more accurate audio transcribers in comparison to existing transcribers such as Google's or IBM's "cloud-speech-recogniser", firstly involved a costly API request token and secondly which requires sending audio data to remote servers for processing, unlike my Faster Whisper implementation which was deployed locally, reducing latency and enhancing privacy by keeping data on-premises. Furthermore, Faster Whisper's optimized architecture allows it to transcribe speech more efficiently, offering comparable or better accuracy than Google's service while requiring less powerful hardware. This combination of speed, accuracy, and local deployment gives Faster Whisper a distinct advantage, particularly in scenarios where low latency and data privacy are critical.

4.3.1 Live Captioning Implementation

In the "WhisperTranscriber" class uses the faster-whisper model to transcribe audio files into text. When given an audio file, the class processes the file by detecting and transcribing speech segments. It filters out non-speech elements to improve transcription accuracy. The final output is a continuous text transcription of the audio, making it useful for my low latency live captioning.

4.4 Text Sentiment Analysis

To save development time and based on prior research, I realized that developing a custom text sentiment classifier from scratch would not match the accuracy or performance of already established models. However, Generic sentiment analysis tools (like [41], [43] etc) would not be well-suited for the nuances of financial text. Therefore, I chose to use the `FinBERT` model, developed by [46] `FinBERT` [38] as my primary text sentiment classifier and the others as auxiliary classifiers to be stored in a `Json` file be compared later.

Model	Type	Architecture	Training Data	Domain	Accuracy
FinBERT [40]	Pre-trained Language Model	Transformer (BERT)	Financial Phrase Bank[phrase]	Financial	97.2% [ProsusAI]
FinBERT (finetuned)	Fine-tuned Language Model	Transformer (BERT)	Financial texts dataset	Financial	
Distilled - RoBERTAa [39]	Pre-trained Language Model	Transformer (BERT, optimized)	Large-scale text corpus	Financial	98.23%
NLTK Vader [41]	Rule-Based Sentiment Analysis	Lexicon-based	Manually created lexicon of words	General	
TextBlob [43]	Rule-Based Sentiment Analysis	Lexicon-based	Predefined word lexicon	General	

Table 1: Text Sentiment Models

4.4.1 Training

To further tailor `FinBERT` for financial text, I fine-tuned it using the dataset `Yahoo-Finance-News-Sentences` dataset. This fine-tuning process involved several steps. First, I used the `AutoTokenizer` library from the Hugging Face transformers library to tokenize the text data, with the tokenizer specifically loaded using `AutoTokenizer.from_pretrained("ProsusAI/finbert")`. The tokenized data was then prepared for training using the map function, which applied the `tokenize_function` to each example in the dataset, ensuring that the text was padded and truncated to the appropriate length.

Next, I utilized the `AutoModelForSequenceClassification` class to load the pre-trained `FinBERT` model, which was transferred to the GPU using `model.to(device)`. The training process was managed by the `Trainer` class, where I set up the training arguments via the `TrainingArguments` class. These arguments included parameters like `num_train_epochs`, `learning_rate`, and `per_device_train_batch_size`, among others, ensuring that the model was fine-tuned effectively. The training loop was initiated with the `train()` method.

Finally, the fine-tuned model and tokenizer were saved using and `save_pretrained()` The fine-tuned `FinBERT` was then used as the main sentiment classifier for financial text, with its predictions being compared to those from auxiliary classifiers stored in a `JSON` file for further analysis.

4.4.2 Texting and Evaluation

After further testing and comparing the performance to the original `FinBERT` model, I realized the challenges of fine-tuning without causing overfitting, misclassification. I experimented with different approaches, such as using callbacks to stop training when accuracy stopped improving, and adjusting the size of the dataset, both limiting and increasing it. However, these adjustments did not yield any significant improvements and, in some cases, decreased the model's accuracy. Consequently, I decided to implement the original pretrained model as it was, without further modifications.

4.5 Speech emotion

As highlighted, above there have been various prior research that has looked into the applications on speech emotion in finance [46] [48], however only gauged the emotion of the speaker as a whole and not in real time. To tackle this issue we used a CNN model proposed by [44] for live speech emotion recognition. Various models have been developed and have been explored. At first the model proposed by [45] [1]. However, during at the evaluation stage of development I could not matching the proposed accuracy, the recall of the model to "`male_suprised`" was highly skewed. After investigation I narrowed down that this could be due "overfitting." This led me to attempt the model proposed at first by [46] and [48] but I could replicate their development. After all this, I decided to adopt a pretrained model by [44] and at testing it produced the highest results of any model investigated.

To enhance the model's performance, I fine-tuned it using a combined dataset of 27,739 recordings created from multiple established datasets, including RAVDESS (Ryerson Audio-Visual Database of Emotional Speech

[1]The original model proposal was in French. However, following additional research, an author Mitesh Puthran translated and publicly posted the proposal on GitHub. The relationship between the two researchers remains unclear.

and Song) [51], SAVEE (Surrey Audio-Visual Expressed Emotion) [52], TESS (Toronto Emotional Speech Set) [53], CREMA-D (Crowd-sourced Emotional Multimodal Actors Dataset) [54], and ESD (Emotional Speech Database) [55]. This combined dataset, which contains seven emotion states recorded (neutral, happy, sad, anger, fear, disgust and surprised).

Dataset	Date	Number of Recordings	Genders	Emotions	Notes
RAVDESS - Ryerson Audio-Visual Database of Emotional Speech and Song [51]	2018	1440	Male/Female	7 (neutral, calm, happy, sad, angry, fearful, surprise, and disgust)	different levels of emotional intensity were recorded, normal and strong, (There is no strong intensity for the 'neutral' emotion.)
SAVEE - Surrey Audio-Visual Expressed Emotion [52]	2011	420	Male only	7 (neutral, happy, sad, angry, fearful, surprise, and disgust)	
TESS – Toronto emotional speech set [53]	2010	2800	Female only	7 (angry, disgust, fear, happiness, pleasant surprise, sadness, and neutral)	
CREMA-D - Crowd-sourced Emotional Multimodal Actors Dataset [54]	2014	6171	Male/Female	6 (happy, sad, angry, fear, disgust, and neutral)	
ESD - Emotional Speech Database [55]	2021	17500	Male/Female	5 (neutral, happy, angry, sad and surprise)	The count of recordings includes only those that are in English.

Table 2: Text Sentiment Models

By consolidating these datasets into a single comprehensive dataset (controlled by "`data_paths`"), the model was exposed to a broader variety of emotional cues and contexts, which hopefully would lead to improved generalization and accuracy across different emotional states. After finetuning on this combined dataset, the model demonstrated superior reliability and accuracy in live speech emotion recognition, making it the optimal choice for our application. The methods and proposals for feature extraction in speech emotion recognition were largely consistent across different models, as most were implemented using Python and they largely shared the same datasets (see chapter 4.2.12).

Using the Librosa library [49] (or the Python Speech Features [50]), 180 spectral features were extracted which was broken down as follows:

- 40 mel-frequency cepstral coefficients spectrogram (`mfcc`) features
- 12 chromogram spectrogram features
- 128 mel-scaled spectrogram (`mss`) features

4.5.1 Training

Following the feature extraction process, the training of the model was approached with a focus on leveraging the power of Convolutional Neural Networks (CNNs) for the task of emotion recognition. A sequential model architecture was selected, consisting of multiple convolutional layers designed to capture the intricate patterns in the audio features extracted. The model was built to handle the 180 spectral features (`MFCCs`, `chroma`, and `MSS`) as an input layer that processes these features as a one-dimensional array. The first convolutional layer contains 256 filters, each with a kernel size of 5, followed by "`ReLU`" activation. This is succeeded by a max-pooling layer with a pool size of 4 to reduce dimensionality. To prevent overfitting, a dropout layer with a 0.5 dropout rate is applied. The second convolutional layer has 128 filters with the same kernel size and activation

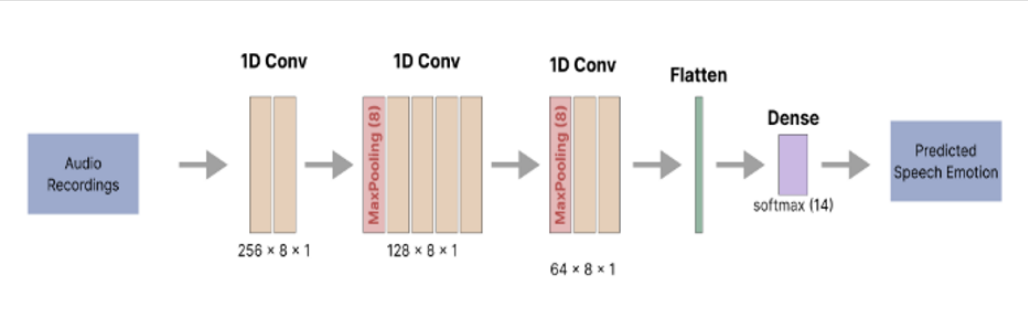


Figure 5: Training and Validation Loss Across Epochs.

function, followed by another max-pooling and dropout layer. After these layers, the output is flattened into a one-dimensional vector, which is fed into a fully connected dense layer with 64 units and "`ReLU`" activation, along with another dropout layer. Finally, a `SoftMax` layer with 8 units corresponding to the emotion classes produces the predicted emotional states. The model is first trained on a labelled testing dataset using the Adam optimizer and categorical cross-entropy loss function. Finally, the model was used to classify unlabelled speech emotions from audio recordings.

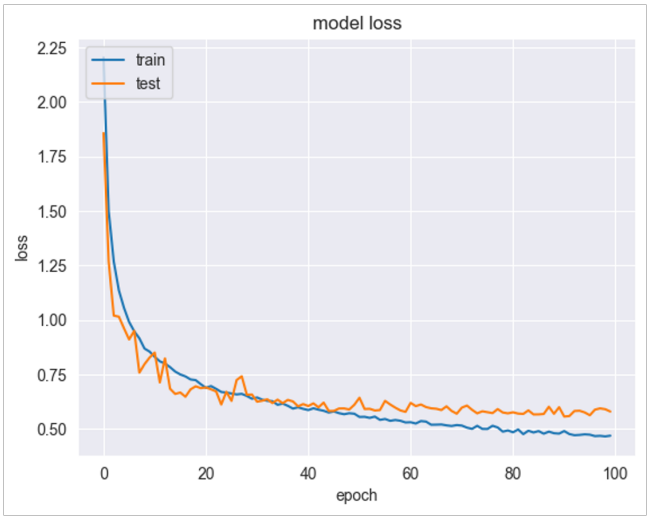


Figure 6: CNN Model Loss Graph

The training was performed over multiple epochs (different versions of models ranging from 100 – 300 epochs) with a random portion of 25% of the dataset reserved for validation, allowing for continuous monitoring of the model's performance and adjustment of hyperparameters as needed. This iterative training process, combined with the use of a robust CNN architecture, ultimately led to a model that could reliably predict emotions from speech, outperforming earlier models that were prone to overfitting and poor generalization.

4.5.2 Evaluation

Through evaluation and testing, the CNN model achieved a 79.8% accuracy but revealed certain biases in its predictions, particularly in the misclassification of specific emotions. For example, the model initially showed a tendency to overfit to certain emotional states, such as "`male_surprised`" and "`male_sad`," where the recall was significantly skewed. This issue was identified through the confusion matrix and classification report, which provided detailed insights into the model's performance across all emotion classes. By analyzing these metrics,

these biases were uncovered, prompting further adjustments and fine-tuning of the model, such as increasing the number of epochs and balancing the model to enhance its accuracy and generalization. This iterative process aimed to ensure that the final model was robust, reducing the likelihood of overfitting and improving its ability to accurately classify a diverse range of emotions in speech recordings. However, these attempts were ultimately unsuccessful, leading me to implement a more robust speech emotion recognition model, such as the selected model "`Wav2Vec 2.0.`" [57].

4.5.3 Wav2Vec

After unsuccessfully attempting to recreate the model proposed by [47], further research revealed that using transformer models outperform CNNs by far [56]. I found a model on Hugging-Face "`Wav2Vec`" [57]. this model claimed an overall accuracy of 82.23% and a loss of 50.23% [57]. the model was a finetuned version of a speech emotion transcriber by Facebook [58] [59] which was trained on the RADVESS [51] dataset which classified speech in to 8 emotional states (`angry`, `calm`, `disgust`, `fearful`, `happy`, `neutral`, `sad`, `surprised`).

The "`HuggingFaceEmotionPredictor`" class is designed to predict speech emotions from an audio file using the "`Wav2Vec`" model. When an audio file is provided, the class first converts the audio into a numerical array through the "`speech_file_to_array`" method, which resamples the audio to the required sampling rate. This processed audio data is then passed to the "`Wav2Vec2FeatureExtractor`" to extract features suitable for the model. These features are subsequently fed into the pre-trained "`Wav2Vec`" model [57], which computes logits representing the likelihood of each possible emotion. These logits are then transformed into probabilities using the "`SoftMax`" function. Finally, the class outputs a list of emotions with their corresponding probabilities, allowing the identification of the most likely emotional state present in the audio input.

5 Results

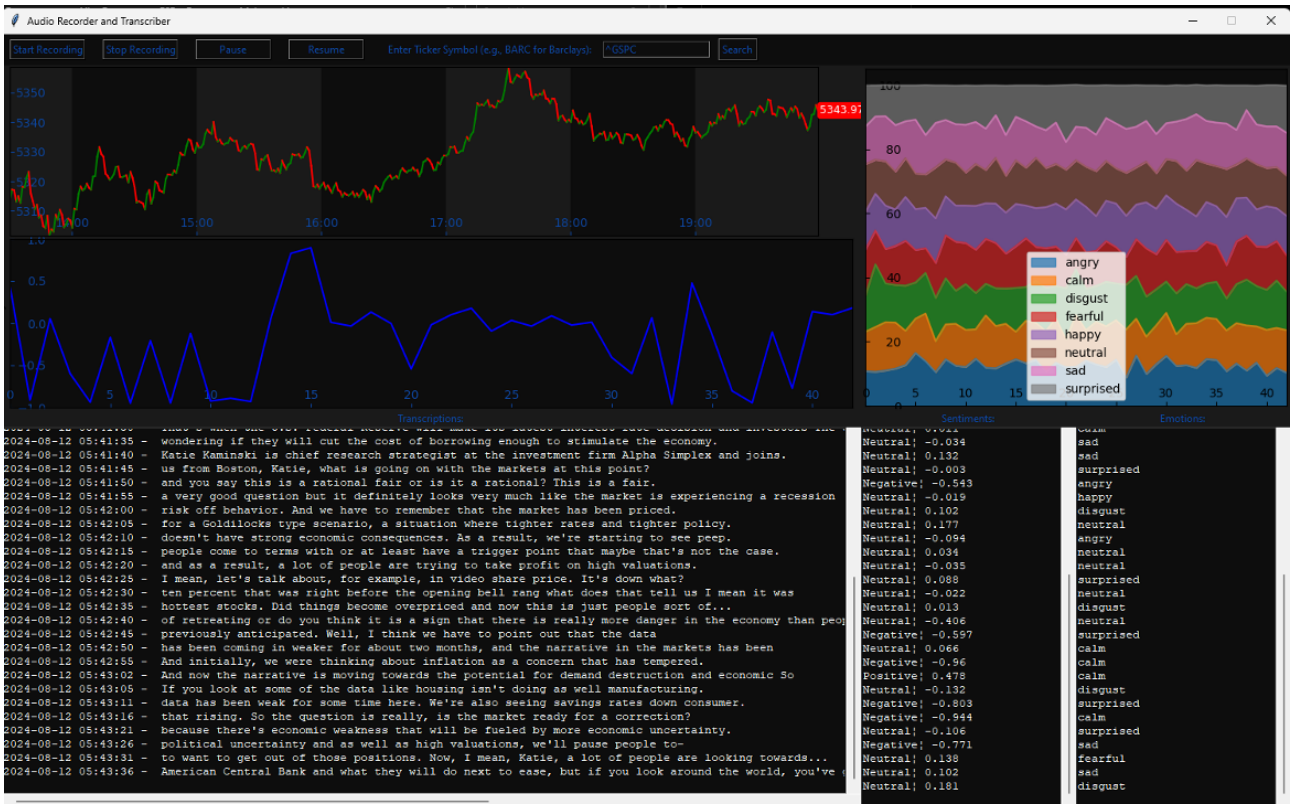


Figure 7: Screenshot of the platform.

The platform offers the following features:

- Records the audio signal in real-time from either the PC audio or a microphone.
- Processes the recorded audio into 5-second segments and saves them.
- Concurrently transcribes/captions (`Whisper`) the audio segments with 81% accuracy.
- Utilizes an NLP BERT model (`FinBERT`) to determine the sentiment of the transcribed audio.
- Simultaneously applies a speech model (`wav2vec`) to analyse the audio for speech emotion.
- Displays the transcribed text along with its captured date and time on the screen.
- Shows the text sentiment and detected speech emotion.
- Continuously updates the displayed text, sentiment, and emotion in real-time.
- Plots sentiment polarity over time.
- Generates a stacked area graph depicting detected speech emotions over time.
- Retrieves live ticker data for any security or commodity. via a built-in search bar

6 Conclusion

In conclusion, this projects presents the successful development of a real-time sentiment analysis and trading system that integrates speech emotion recognition to enhance decision-making in financial markets. The project represents a significant step forward in the field of FinTech, bridging the gap between computer science and finance through innovative applications of machine learning and natural language processing.

The system developed in this project has several key features, including real-time audio transcription, sentiment analysis using fine-tuned FinBERT [44] models, and emotion detection through advanced speech recognition models like Wav2Vec [57]. The integration of these technologies enables the system to analyze market sentiment from live audio sources, such as conference calls and news broadcasts, providing traders with actionable insights based on both the content and emotional tone of spoken information.

Despite the challenges faced, such as the initial difficulty in achieving high accuracy in speech emotion recognition and the complexities involved in fine-tuning sentiment analysis models for financial data, the final implementation demonstrated robust performance. The system achieved significant accuracy in both sentiment detection and emotion recognition, proving its potential as a tool for enhancing trading strategies and decision-making processes in real time.

This work contributes to the ongoing advancements in financial technology by demonstrating how emerging AI and ML technologies can be applied to traditional trading practices. The system's ability to process and analyze live data provides a foundation for future developments in real-time trading platforms, particularly as AI continues to evolve.

7 Future Work

In the future, further optimization and adaptation of the system could have led to more sophisticated trading tools that incorporated additional data sources and predictive capabilities. I would have liked to truly extend the work done by [47], which included the ability to predict the future financial distress of a company using sentiment and speech emotion analysis. However, I was unable to find ready-made datasets containing live conference calls. Even if such datasets had been available, I struggled to determine a method for classifying and quantifying the impact of these factors on real-time stock prices or company performance.

I also would have loved to continue exploring different features of speech, such as measuring the level of trust in a speaker's voice or detecting cognitive dissonance as studied by [60] and [61]. Additionally, I was interested in investigating whether a speaker's facial expressions matched their vocal expressions and examining the potential effects on the respective stock price. I also wish I had improved the way I displayed sentiment polarity. Looking back, I would have plotted the stock price and polarity together to explore whether there was any correlation between the two at certain points.

In my opinion, incorporating any of these methods would have ultimately led to more refined investor insights and better decision-making, resulting in more informed and effective financial decisions.

8 Appendices

8.1 Abbreviations

- **AI** - Artificial Intelligence
- **ML** - Machine Learning
- **NLP** - Natural Language Processing
- **FinTech** - Financial Technology
- **SER** - Speech Emotion Recognition
- **CNN** - Convolutional Neural Network
- **RNN** - Recurrent Neural Network
- **LSTM** - Long Short-Term Memory
- **BERT** - Bidirectional Encoder Representations from Transformers
- **FinBERT** - Financial Bidirectional Encoder Representations from Transformers
- **GUI** - Graphic User Interface
- **MFCC** - Mel-Frequency Cepstral Coefficients
- **STFT** - Short-Time Fourier Transform
- **FFT** - Fast Fourier Transform
- **DCT** - Discrete Cosine Transform
- **TF-IDF** - Term Frequency - Inverse Document Frequency
- **HFT** - High-Frequency Trading
- **IPO** - Initial Public Offering
- **AP** - Associated Press
- **RAVDESS** - Ryerson Audio-Visual Database of Emotional Speech and Song
- **SAVEE** - Surrey Audio-Visual Expressed Emotion
- **TESS** - Toronto Emotional Speech Set
- **CREMA-D** - Crowd-sourced Emotional Multimodal Actors Dataset
- **ESD** - Emotional Speech Database
- **API** - Application Programming Interface
- **SVM** - Support Vector Machine
- **BoW** - Bag of Words
- **TF** - Term Frequency
- **BoW** - Bag of Words

List of Figures

References

[1] [ec911] "Computational Market Microstructure for FinTech and the Digital Economy" moodle.essex.ac.uk 2024. [Online]. Available: https://moodle.essex.ac.uk/course/view.php?id=3874 [Accessed 10 08 2024]

[2] [cf963] "Computational Models in Economics and Finance moodle.essex.ac.uk" 2024. [Online]. Available: hhttps://moodle.essex.ac.uk/course/view.php?id=3874 [Accessed 08 07 2024]

[3] [capstone] E. Olaoye,"Visual Pathfinding Visualiser" CE301 Final Year Capstone Project, 2023. [Online]. Available: https://github.com/Emmanuelolaoye/Visual-Pathfinding-Visualiser/blob/main/Final%20Project/1905600%20CE301%20Final%20Report.pdf. [Accessed: 15-Aug-2024]

[4] [NLE] "Natural Language Engineering" moodle.essex.ac.uk 2024. [Online]. Available: https://moodle.essex.ac.uk/enrol/index.php?id=3711 [Accessed 10 08 2024]

[5] [Linkedin] K. Ponnambalam, 'Getting started with Transformers - Applied Ai: Getting Started with hugging face Transformers video tutorial: Linkedin learning, formerly Lynda.com', LinkedIn. Feb-2023.

[6] Dystopian Dev, Audio Spectrogram - Python + OpenGL + PyAudio, Youtube. Available at: https://www.youtube.com/watch?v=uapmmpA1wMk, (Accessed: 14 May 2024). 2024

[7] All About AI, SUPER Fast AI Real Time Speech to Text Transcribtion - Faster Whisper / Python, YouTube. Available at: https://www.youtube.com/watch?v=k6nIxWGdrS4 (Accessed: 26 April 2024). 2024

[8] [Hung] L. P. Hung and S. Alias, 'Beyond Sentiment Analysis: A Review of Recent Trends in Text Based Sentiment Analysis and Emotion Detection', Journal of Advanced Computational Intelligence and Intelligent Informatics, vol. 27, no. 1, pp. 84–95, 2023.

[9] M. T. Riaz, M. Shah Jahan, S. G. Khawaja, A. Shaukat, and J. Zeb, 'TM-BERT: A Twitter Modified BERT for Sentiment Analysis on Covid-19 Vaccination Tweets', in 2022 2nd International Conference on Digital Futures and Transformative Technologies (ICoDT2), 2022, pp. 1–6.

[10] [Aslam] N. Aslam, F. Rustam, E. Lee, P. B. Washington, and I. Ashraf, 'Sentiment Analysis and Emotion Detection on Cryptocurrency Related Tweets Using Ensemble LSTM-GRU Model', IEEE Access, vol. 10, pp. 39313–39324, 2022.

[11] [Poria] S. Poria, N. Majumder, D. Hazarika, E. Cambria, A. Gelbukh, and A. Hussain, 'Multimodal Sentiment Analysis: Addressing Key Issues and Setting up the Baselines', arXiv [cs.CL]. 2019.

[12] [Adam] A. Hariharan and M. T. P. Adam, 'Blended Emotion Detection for Decision Support', IEEE Transactions on Human-Machine Systems, vol. 45, no. 4, pp. 510–517, 2015.

[13] [Twitter] S. A. Dojan, 'Federal Reserve holds interest rate steady amid cooling inflation', investment week, Jun. 2024.

[14] [Brexit] Y. Gorodnichenko, T. Pham, and O. Talavera, 'Social media, sentiment and public opinions: Evidence from #Brexit and #USElection', European Economic Review, vol. 136, p. 103772, 2021.

[15] [Alibaba] B. Zhang, 'An Analysis of Factors Affecting IPO Performance Based on Alibaba IPO', Highlights in Business, Economics and Management, vol. 35, pp. 258–263, 06 2024.

[16] [GameStop] A. Anand and J. Pathak, 'The role of Reddit in the GameStop short squeeze', Economics Letters, vol. 211, p. 110249, 2022.

[17] [COVID] M. A. Harjoto and F. Rossi, 'Market reaction to the COVID-19 pandemic: evidence from emerging markets', International Journal of Emerging Markets, vol. 18, no. 1, pp. 173–199, 2023.

[18] [2008] A. Bandopadhyaya and D. Truong, 'Who knew: Financial crises and investor sentiment', 2010.

[19] A. Grody and H. Levecq, 'Past, Present and Future: The Evolution and Development of Electronic Financial Markets', 11 1993.

[20] A. B. Schmidt, Financial Markets and trading: An introduction to market microstructure and trading strategies. Wiley, 2013.

[21] D. Easley, M. Lopez de Prado, and M. O'Hara, 'The Microstructure of the "Flash Crash": Flow Toxicity, Liquidity Crashes and the Probability of Informed Trading', The Journal of Portfolio Management, vol. 37, 11 2010.

[22] D. Shah, H. Isah, F. Zulkernine, Predicting the effects of news sentiments on the stock market. In Proceedings of the 2018 IEEE International Conference on Big Data (Big Data), Seattle, WA, USA, 10–13 December 2018; IEEE: Piscataway, NJ, USA, 2018; pp. 4705–4708.

[23] S. M. H. Dadgar, M. S. Araghi, and M. R. M. Farahani, 'A novel text mining approach based on TF-IDF and Support Vector Machine for news classification', 2016 IEEE International Conference on Engineering and Technology (ICETECH), pp. 112–116, 2016.

[24] M. Hagenau, M. Liebmann, and D. Neumann, 'Automated news reading: Stock price prediction based on financial news using context-capturing features', Decision Support Systems, vol. 55, no. 3, pp. 685–697, 2013.

[25] W. K. Cheng, K. T. Bea, S. M. H. Leow, J. Y.-L. Chan, Z.-W. Hong, and Y.-L. Chen, 'A Review of Sentiment, Semantic and Event-Extraction-Based Approaches in Stock Forecasting', Mathematics, vol. 10, no. 14, 2022.

[26] M. Jaggi, P. Mandal, S. Narang, U. Naseem, and M. Khushi, 'Text Mining of Stocktwits Data for Predicting Stock Prices', Applied System Innovation, vol. 4, no. 1, 2021.

[27] A. Tyagi and N. Sharma, 'Sentiment Analysis using Logistic Regression and Effective Word Score Heuristic', International Journal of Engineering and Technology(UAE), vol. 7, pp. 20–23, 04 2018.

[28] C. McIntosh, Cambridge advanced learner's dictionary /, 4th ed. Cambridge University Press, 2013.

[29] S. A. Dojan, 'Federal Reserve holds interest rate steady amid cooling inflation', investment week, Jun. 2024.

[30] M. A. Yusnita, M. P. Paulraj, S. B. Yaacob, R. Yusuf, and A. B. Shahriman, 'Analysis of Accent-Sensitive Words in Multi-Resolution Mel-Frequency Cepstral Coefficients for Classification of Accents in Malaysian English', International Journal of Automotive and Mechanical Engineering, vol. 7, pp. 1053–1073, 2013.

[31] C. E. Williams and K. N. Stevens, 'Emotions and Speech: Some Acoustical Correlates', The Journal of the Acoustical Society of America, vol. 52, no. 4B, pp. 1238–1250, 10 1972.

[32] R. Cowie and R. R. Cornelius, 'Describing the emotional states that are expressed in speech', Speech Communication, vol. 40, no. 1, pp. 5–32, 2003.

[33] M. A. Yusnita, M. P. Paulraj, S. Yaacob, R. Yusuf, and A. B. Shahriman, "Analysis of accent-sensitive words in multi-resolution Mel-frequency cepstral coefficients for classification of accents in Malaysian English," International Journal of Automotive and Mechanical Engineering, vol. 7, pp. 1053–1073, Jun. 2013, Figure 3: Mel filter banks basis functions using 20 Mel-filters in the filter bank.

[34] A. Shah, M. Kattel, A. Nepal, and D. Shrestha, 'Chroma Feature Extraction', 01 2019.

[35] K. Tarunika, R. B. Pradeeba, and P. Aruna, 'Applying Machine Learning Techniques for Speech Emotion Recognition', in 2018 9th International Conference on Computing, Communication and Networking Technologies (ICCCNT), 2018, pp. 1–5.

[36] M. Müller and S. Ewert, 'Chroma Toolbox: Matlab Implementations for Extracting Variants of Chroma-Based Audio Features', 01 2011, pp. 215–220.

[37] T. H. Chowdhury, K. N. Poudel, and Y. Hu, 'Time-Frequency Analysis, Denoising, Compression, Segmentation, and Classification of PCG Signals', IEEE Access, vol. 8, pp. 160882–160890, 2020.

[38] ProsusAI D. Araci, 'FinBERT: Financial Sentiment Analysis with Pre-trained Language Models', CoRR, vol. abs/1908.10063, 2019.

[39] roBERTA V. Sanh, L. Debut, J. Chaumond, and T. Wolf, 'DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter', ArXiv, vol. abs/1910.01108, 2019.

[40] [ProsusAI] D. Araci, 'FinBERT: Financial Sentiment Analysis with Pre-trained Language Models', CoRR, vol. abs/1908.10063, 2019.

[41] [Vader] C. J. Hutto and E. Gilbert, 'VADER: A Parsimonious Rule-based Model for Sentiment Analysis of Social Media Text', 01 2015.

[42] [phrase] P. Malo, A. Sinha, P. Takala, P. Korhonen, and J. Wallenius, 'Good Debt or Bad Debt: Detecting Semantic Orientations in Economic Texts', Journal of the American Society for Information Science and Technology, 04 2014.

[43] [TextBlob] S. Loria, 'textblob Documentation', Release 0. 15, vol. 2, 2018.

[44] FinBERT D. Araci, "FinBERT, (Revision 4556d13)," Hugging Face, 2023. [Online]. Available: https://huggingface.co/ProsusAI/finbert.

[45] [Fabien RINGEVAL] F. Ringeval, 'Ancrages et modèles dynamiques de la prosodie: application à la reconnaissance des émotions actées et spontanées', Université Pierre et Marie Curie - Paris VI, 2011.

[46] [Hajek] P. Hajek and M. Munk, 'Speech emotion recognition and text sentiment analysis for financial distress prediction', Neural Computing and Applications, vol. 35, no. 29, pp. 21463–21477, Oct. 2023.

[47] [Hayek] P. Hajek and M. Munk, 'Speech emotion recognition and text sentiment analysis for financial distress prediction', Neural Computing and Applications, vol. 35, no. 29, pp. 21463–21477, Oct. 2023.

[48] [loans] N. Zhao and F. Yao, 'Innovative Mechanism of Rural Finance: Risk Assessment Methods and Impact Factors of Agricultural Loans Based on Personal Emotion and Artificial Intelligence', Journal of Environmental and Public Health, vol. 2022, pp. 1–9, 05 2022.

[49] [Librosa] B. McFee, "librosa/librosa: 0.10.2.post1". Zenodo, May 14, 2024. doi: 10.5281/zenodo.11192913.

[50] [Python Speech Features] J. Lyons, "jameslyons/python_speech_features: release v0.6.1". Zenodo, Jan. 14, 2020. doi: 10.5281/zenodo.3607820.

[51] [RAVDESS] S. R. Livingstone and F. A. Russo, 'The Ryerson Audio-Visual Database of Emotional Speech and Song (RAVDESS): A dynamic, multimodal set of facial and vocal expressions in North American English', PLOS ONE, vol. 13, no. 5, pp. 1–35, 05 2018.

[52] [SAVEE] P. Jackson and S. Ul haq, 'Surrey Audio-Visual Expressed Emotion (SAVEE) database'. 04 2011.

[53] [TESS] K. Dupuis and M. K. Pichora-Fuller, 'Toronto emotional speech set (tess)-younger talker_happy', 2010.

[54] [CREMA] H. Cao, D. G. Cooper, M. K. Keutmann, R. C. Gur, A. Nenkova, and R. Verma, 'CREMA-D: Crowd-Sourced Emotional Multimodal Actors Dataset', IEEE Transactions on Affective Computing, vol. 5, no. 4, pp. 377–390, 2014.

[55] [ESD] K. Zhou, B. Sisman, R. Liu, and H. Li, 'Emotional Voice Conversion: Theory, Databases and ESD', arXiv [cs.CL]. 2022.

[56] [transformer] A. Baevski, H. Zhou, A. Mohamed, and M. Auli, 'wav2vec 2.0: A Framework for Self-Supervised Learning of Speech Representations', arXiv [cs.CL]. 2020.

[57] [w2v] Enrique Hernández Calabrés, 'wav2vec2-lg-xlsr-en-speech-emotion-recognition (Revision 17cf17c)'. Hugging Face, 2024.

[58] [Facebook] A. Conneau, A. Baevski, R. Collobert, A. Mohamed, and M. Auli, 'Wav2Vec2-XLSR-53, (Revision c3f9d88)'. Hugging Face, 2022.

[59] [facebook] model from paper A. Conneau, A. Baevski, R. Collobert, A. Mohamed, and M. Auli, 'Unsupervised Cross-lingual Representation Learning for Speech Recognition', arXiv [cs.CL]. 2020.

[60] [dissonance] J. L. Hobson, W. J. Mayew, and M. Venkatachalam, 'Analyzing Speech to Detect Financial Misreporting', Journal of Accounting Research, vol. 50, no. 2, pp. 349–392, 2012.

[61] [trust] M. Deng et al., 'Using voice recognition to measure trust during interactions with automated vehicles', Applied Ergonomics, vol. 116, p. 104184, 2024.

9 GitLab Link:

https://cseegit.essex.ac.uk/22-24-ce901-ce911-cf981-su/22-24_CE901-CE911-CF981-SU_olaoye_emmanuel_o.git *Note: The project files referenced in this work are available in the "Final-Project" branch.*