Financial Sentiment Analysis using Various NLP Techniques

Petec Răzvan-Gabriel

Advanced Methods in Data Analysis – Research Report 2
Specialization: Applied Computation Intelligence
Group: 246/2

razvan.petec@stud.ubbcluj.ro

ABSTRACT

This study explores the application of sequential models in financial sentiment analysis, comparing the performance of Recurrent Neural Networks (RNNs), Long Short-Term Memory Networks (LSTMs), and Transformers. Utilizing a benchmark dataset of financial phrases annotated with sentiment labels (positive, neutral, and negative), the models were evaluated based on metrics such as precision, recall, F1-score, and accuracy. The results demonstrate that the Transformer model outperformed its counterparts, achieving the highest accuracy (64%) and consistently strong performance across all sentiment classes, particularly excelling in positive and neutral sentiments. LSTMs exhibited balanced results with improved generalization over RNNs, while RNNs struggled with the negative sentiment class due to their limited contextual understanding.

1. INTRODUCTION

In the rapidly evolving world of financial markets, timely and accurate sentiment analysis has become an invaluable tool for understanding market trends and predicting investor behaviour. Financial sentiment analysis focuses on extracting opinions and attitudes from text-based data sources, such as news articles, earnings reports, and social media posts, to discern their potential influence on market movements. Unlike traditional sentiment analysis in general domains, financial sentiment is uniquely challenging due to its domain-specific language, nuanced phrase structures, and directional dependencies of economic terms.

Sentiment in financial contexts often transcends simple word-level interpretations. For example, phrases such as “profit decreased by 10” convey negative sentiment despite containing neutral or positive terms like “profit”. Domain-specific lexicons, such as the Loughran-McDonald financial sentiment dictionary, have highlighted this issue, demonstrating that general sentiment tools often fail to capture the subtleties of financial texts effectively. Similarly, the establishment of datasets like the Financial Phrase Bank [1] has provided crucial benchmarks for evaluating and improving financial sentiment analysis models [2]. These resources underscore the need for specialized approaches that integrate financial domain expertise with advanced natural language processing (NLP) techniques.

Recent advancements in machine learning (ML) and NLP have paved the way for increasingly sophisticated models capable of handling complex linguistic structures. Deep learning architectures such as recurrent neural networks (RNNs), long short-term memory networks (LSTMs), and transformers have shown significant promise in capturing context and sentiment in text data. Particularly, transformers like BERT have demonstrated state-of-the-art performance across a variety of NLP tasks, including sentiment analysis, by leveraging attention mechanisms and pre-trained

contextual embeddings [3]. However, their application in financial sentiment analysis remains an area of active exploration, particularly concerning how these models can handle the intricate dependencies and unique lexicons of financial texts.

This study seeks to advance the field of financial sentiment analysis by leveraging a high-quality, annotated dataset of financial phrases and implementing state-of-the-art NLP models. Specifically, the research focuses on evaluating the performance of different architectures (RNNs, LSTMs, and transformers) in capturing the sentiment dynamics of financial texts. By systematically comparing these models and conducting error analyses, the goal of this paper is to identify the strengths and limitations of each approach, offering insights into their applicability in real-world financial forecasting and decision-making.

2. LITERATURE REVIEW

The field of sentiment analysis has seen significant advancements over the past decades, driven by developments in natural language processing (NLP) and machine learning (ML). While general sentiment analysis has been widely studied in domains like movie reviews and social media, financial sentiment analysis presents unique challenges due to its domain-specific language and the contextual nature of sentiment in financial texts. This section reviews key contributions to financial sentiment analysis, focusing on domain-specific lexicons, datasets, and the application of advanced ML models.

2.1. Domain-Specific Lexicons and Datasets

One of the foundational resources for financial sentiment analysis is the Loughran-McDonald (LM) lexicon, which addresses the limitations of general-purpose sentiment lexicons like the Harvard Dictionary. The LM lexicon identifies financial-specific vocabulary that can more accurately reflect sentiment in financial texts. For instance, words such as “liability”, which may be neutral in general contexts, often carry a negative sentiment in financial contexts [1].

The creation of annotated datasets has also been instrumental in advancing financial sentiment analysis. Malo et al. introduced the Financial Phrase Bank, a benchmark dataset containing over 5,000 financial phrases manually annotated for sentiment polarity (positive, negative, or neutral) by business experts. This dataset has become a critical tool for training and evaluating sentiment analysis models, providing the domain relevance and linguistic complexity necessary for robust model development [1].

2.2. Machine Learning Models for Sentiment Analysis

Early approaches to sentiment analysis in the financial domain relied on "bag-of-words" models and dictionary-based methods to detect sentiment from text [4]. However, these models often struggled to capture the nuances of financial language, particularly the influence of context and phrase structures on sentiment polarity. To address these limitations, researchers began exploring statistical and machine learning methods, such as Naïve Bayes and Support Vector Machines (SVMs), which offered improved accuracy by leveraging feature engineering and data-driven approaches [1].

More recently, deep learning has revolutionized sentiment analysis by enabling models to learn complex relationships between words and their contexts. Recurrent Neural Networks (RNNs) and Long Short-Term Memory Networks (LSTMs) have demonstrated significant success in handling sequential data and capturing dependencies in text. Devlin et al. (2018) introduced the Bidirectional Encoder Representations from Transformers (BERT), a transformer-based model pre-trained on vast

amounts of text data. BERT's ability to capture bidirectional context has made it a state-of-the-art tool in NLP, including sentiment analysis tasks [3].

While these advanced models have shown promise in general NLP tasks, their application in financial sentiment analysis remains an active area of research. The domain-specificity of financial texts means that models trained on general datasets often fail to generalize effectively to financial contexts. Additionally, the dynamic nature of financial language, influenced by market trends and events, necessitates continuous updates to lexicons and models. Future work could explore hybrid approaches that combine domain-specific lexicons, like the LM lexicon, with transformer-based architectures to enhance performance in financial sentiment tasks.

3. METHODOLOGY

3.1. Tokenizer

The BERT tokenizer is a fundamental component of the BERT model [3], responsible for converting raw text into a format suitable for model processing. It employs the WordPiece algorithm, a subword tokenization technique that effectively handles rare or out-of-vocabulary words by decomposing them into smaller, more manageable subword units.

The tokenization process can be divided in the following:

- **Basic Tokenization.** The tokenizer begins by performing standard text preprocessing, which includes:
- **Lowercasing**. Converting all characters to lowercase (in the case of uncased models) to ensure uniformity.
- **Punctuation Handling**. Isolating punctuation from words to treat them as separate tokens.
- **Whitespace Management**. Splitting text based on whitespace to identify individual words or tokens.
- **WordPiece Tokenization**. After basic tokenization, each word is further decomposed into subword units using the WordPiece algorithm:
- **Vocabulary Lookup**. The tokenizer maintains a predefined vocabulary of subword units. It attempts to match the longest possible subword from this vocabulary to the beginning of the word.
- **Recursive Decomposition**. If a word is not found in the vocabulary, it is recursively split into smaller subword units until all components are recognized or reduced to individual characters.
- **Unknown Tokens**. Subwords or characters not present in the vocabulary are represented by a special [UNK] (unknown) token.

Subword units identified by the WordPiece algorithm are denoted with a specific prefix (commonly ##) to indicate that they are continuations of a preceding token. For example, the word “playing” might be tokenized into [“play”, “##ing”], signifying that “##ing” is a suffix attached to “play”.

3.2. Dataset

3.2.1. Overview

The Financial Phrase Bank, introduced by Malo et al. (2013), is a carefully curated dataset designed to address the challenges of sentiment analysis in financial contexts. It consists of approximately 5,000 phrases or sentences extracted from financial news, press releases, and other related sources. Each entry in the dataset is annotated with one of three sentiment labels: positive, negative, or neutral. The annotations were performed by 16 business-educated annotators, ensuring the dataset's reliability and domain relevance [1].

The Financial Phrase Bank serves as a benchmark resource for training and evaluating sentiment analysis models. Unlike general sentiment datasets, which often fail to capture the nuances of financial language, this dataset incorporates domain-specific complexities, such as the directional influence of financial events (e.g., “profit increased” vs. “profit decreased”) and the context-sensitive nature of sentiment in economic texts [1].

The annotation process for the Financial Phrase Bank adhered to rigorous standards to ensure accuracy and consistency. Each phrase was annotated multiple times by the annotators, with disagreements resolved through majority voting. This approach mitigated the risk of individual bias and provided a robust ground truth for model training and evaluation [1].

The annotators' background in finance and business was a crucial factor in the dataset's quality. Sentiment in financial contexts often depends on subtle linguistic cues and domain-specific knowledge. For instance, phrases like “a liability of $1 million” might seem neutral in general contexts but are interpreted as negative in finance. The annotators' expertise allowed them to accurately capture these subtleties, making the Financial Phrase Bank a valuable resource for domain-specific sentiment analysis [1].

3.2.2. Analysis

The distribution of token lengths across the dataset, shown in Figure 1, reveals key characteristics of the financial texts used. Most phrases in the dataset are relatively short, with token lengths peaking between 20 and 40 tokens. This is consistent with the concise nature of financial reporting and news excerpts, which aim to deliver clear and direct information. Longer phrases, exceeding 60 tokens, are rare, highlighting the dataset's focus on digestible text fragments. This distribution is crucial for model training, as it informs decisions about maximum sequence lengths and padding strategies.

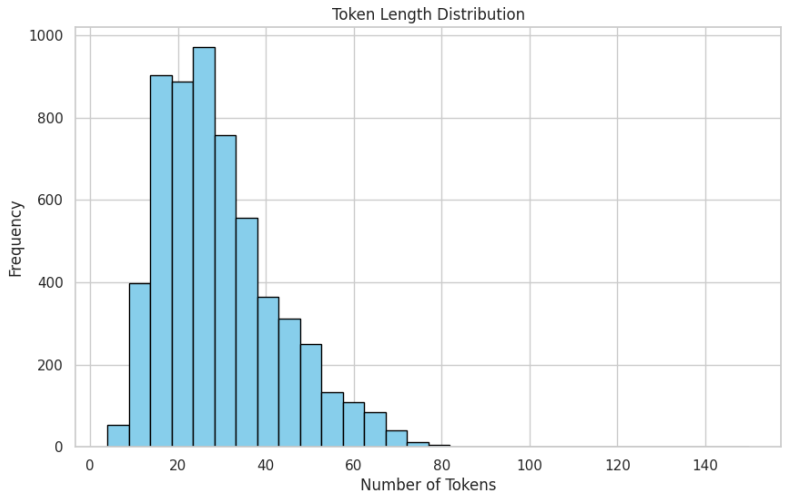

Figure 1. *Tokens length distribution in the Financial Phrase Bank Dataset.*

Using t-SNE (t-Distributed Stochastic Neighbor Embedding), the dataset's token embeddings are visualized in Figure 2. The figure demonstrates how the tokens are distributed in the embedding space after padding. While clusters are present, they are not distinctly separated, indicating the overlap and

complexity of sentiment expressions in financial texts. The colours correspond to the three sentiment classes (positive, negative, neutral), and the overlap reflects the subtle linguistic differences between these categories.

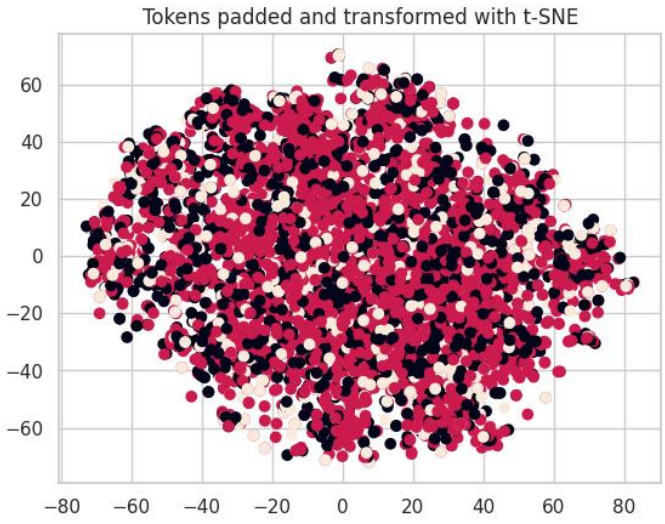

Figure 2. *Tokens padded and reduced using t-SNE (t-Distributed Stochastic Neighbor Embedding).*

Principal Component Analysis (PCA) is used in Figure 3 to reduce the token embeddings into a two-dimensional space. The resulting visualization shows a more compact but less interpretable representation compared to t-SNE. This highlights the high-dimensional nature of the embeddings and the inherent challenges in visualizing and analysing them. Similar to the t-SNE visualization, the PCA plot reveals substantial overlap between the sentiment classes, reflecting the complexities of financial sentiment analysis.

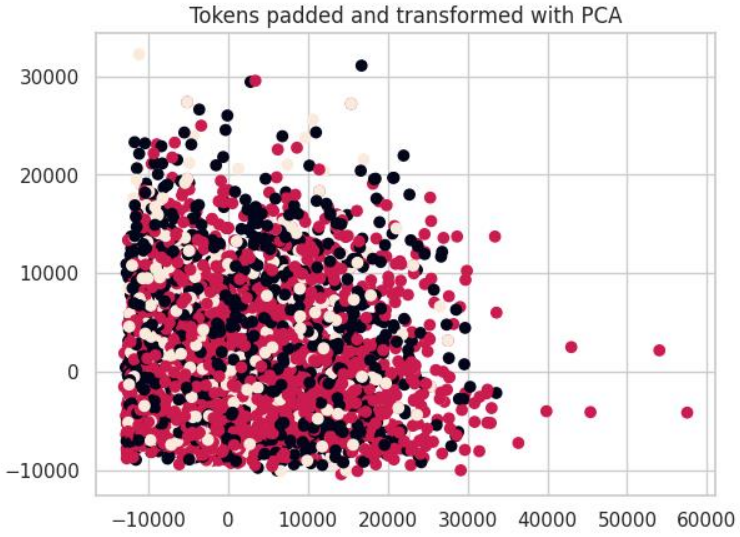

Figure 3. *Tokens padded and reduced using PCA (Principal Component Analysis).*

The frequency of the most common tokens is displayed in Figure 3, showcasing the distribution of tokens in the dataset. Punctuation marks (e.g., “.”, “,”) and stopwords (e.g., “the”, “of”) dominate the top positions, which is typical for natural language datasets. Domain-specific tokens like “##re" (indicative of subwords in tokenized financial terminology) and monetary symbols like “$” highlight the dataset's financial focus.

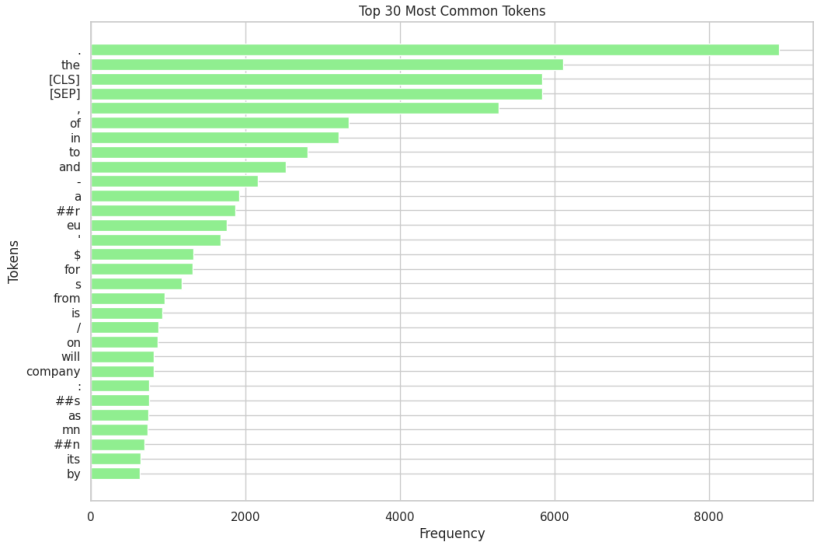

Figure 4. *Top 30 most common tokens in the Financial Phrase Bank Dataset.*

Lastly, the distribution of sentiment labels across the training and validation datasets is visualized in Figure 5. This figure highlights a significant class imbalance, with the neutral sentiment category being the most frequent, followed by the positive and negative classes. This is consistent with the nature of financial texts, where statements are often objective and descriptive, rather than overtly positive or negative. Positive sentiment phrases are moderately represented, while negative sentiment phrases make up the smallest portion of the dataset.

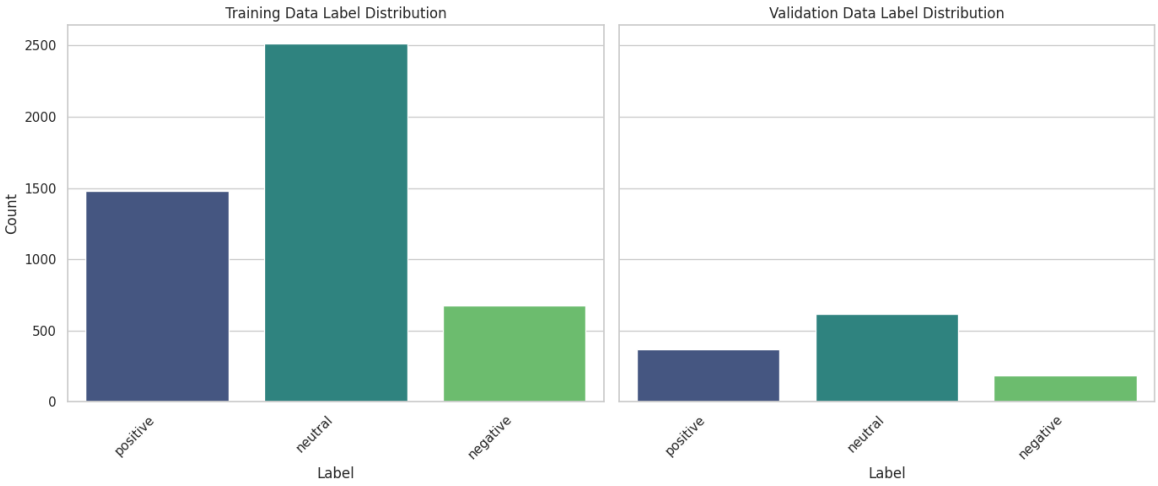

Figure 5. *Labels distribution across the training set and the validation set.*

The consistency of label distributions between the training and validation datasets is a positive attribute of the dataset, as it ensures that the model's performance on the validation set reflects its behavior on unseen data with similar characteristics. However, the imbalance poses a challenge for model training, as models are inherently biased towards predicting the majority class. This could lead to suboptimal performance, particularly in accurately identifying the minority classes, which are positive and negative sentiments in this case.

3.3. Sequential Models

The intricate nature of financial language presents unique challenges. Financial texts are often laden with domain-specific terminology, complex syntactic structures, and nuanced expressions that convey sentiment in subtle ways. Traditional natural language processing (NLP) techniques may struggle to effectively capture these complexities, leading to less accurate sentiment classification.

3.3.1. Recurrent Neural Networks

Recurrent Neural Networks (RNNs) are a type of neural network designed to work with sequence data. Unlike traditional neural networks, also called Feed-Forward Networks or Multi-Layer Perceptrons (MLPs), RNNs process data in a looped manner. While Feed-Forward Networks pass information straight through their layers, RNNs use loops that allow information from earlier parts of a sequence to influence later ones. This structure enables RNNs to take both the current input X_t and all prior inputs $X_{0:t-1}$ into account when making predictions. This concept is illustrated schematically in Figure 6.

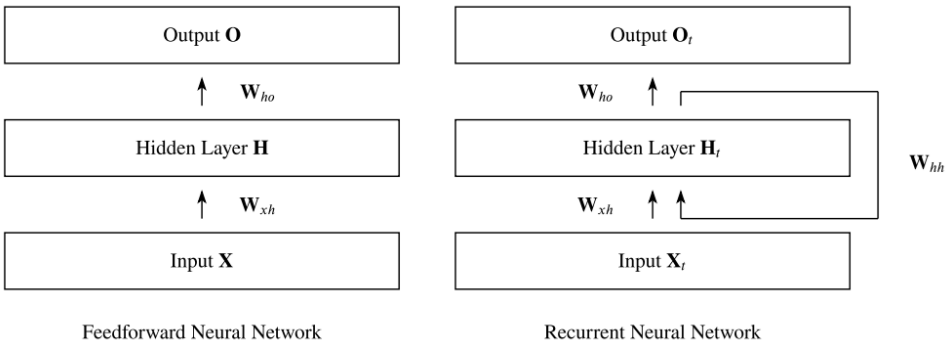

Figure 6. *Feed-Foward Neural Networks and Recurrent Neural Networks [6].*

However, RNNs face challenges like the vanishing or exploding gradient problem, which makes it hard for the network to learn long-term dependencies in data. To solve this issue, Long Short-Term Memory (LSTM) networks were created [6]. Another limitation of RNNs is that they process data step by step. Since each step depends on the previous one, this sequential nature makes it hard to fully use the parallel computing power of modern GPUs.

3.3.2. Long Short-Term Memory

Long Short-Term Memory networks (LSTMs) [7] were developed to overcome the vanishing gradient issue in RNNs. They include “gated cells” that allow the network to decide what information to keep, what to update, and what to forget. Each LSTM cell has three gates: an input gate, which allows new information to enter; a forget gate, which removes unnecessary information; and an output gate, which extracts the useful data from the cell. The structure of an LSTM cell is illustrated in Figure 7.

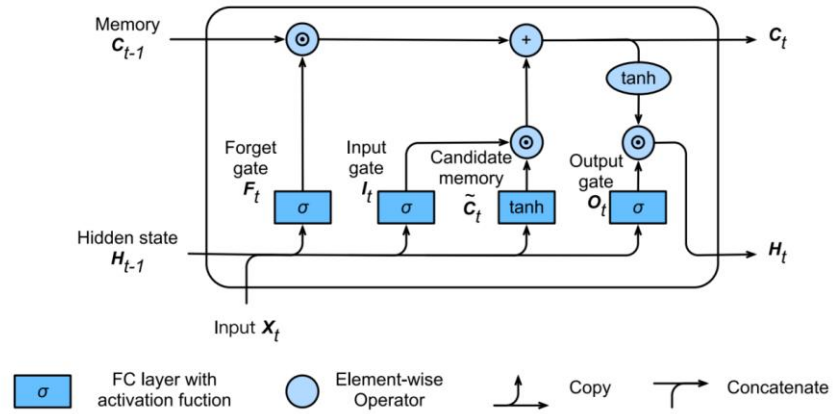

Figure 7. *Feed-Foward Neural Networks and Recurrent Neural Networks [6].*

LSTMs also maintain a memory component (C_t) that stores long-term information. The previous memory state (C_{t-1}) is combined with the outputs of the gates to determine how much of the old memory should remain and how much new information should be added. This process makes LSTMs particularly good at understanding long sequences. Despite their advantages, LSTMs also process data step by step, which limits their ability to use GPUs efficiently for parallel computations.

3.3.3. Attention Mechanism

The attention mechanism, inspired by how humans focus on important details while keeping track of the broader picture, helps improve sequence models. In text processing, attention allows the model to focus on relevant parts of a sentence or sequence when making predictions. For example, in the sentence "She is eating a green apple," the word "eating" suggests that the next word is likely related to food [6]. The attention mechanism uses this kind of context to better predict outcomes.

The Attention Mechanism operates by transforming two sequences, such as sentences, into a matrix where each word corresponds to a row or column. By populating the matrix, the relevant contexts or correlations between words can be discerned. This approach is versatile, applicable not only to pairs of sentences in different languages but also to single sentences, a process known as self-attention. The attention mechanism is computed using a set of queries, keys, and values, which are organized into matrices Q, K, and V respectively. The attention calculation is expressed as:

$$Attention(Q, K, V) = softmax\left(\frac{QK^T}{\sqrt{d_k}}\right)V,$$

where d_k represents the dimension of the keys. The query-key dot product is scaled by $\sqrt{dk}$ to form the “Scaled Dot-Product Attention” [8].

3.3.4. Transformer

The Transformer model [8], introduced with the attention mechanism, eliminates the need for processing sequences step by step. Instead, it uses parallel processing, making it much faster and more efficient. Transformers rely on self-attention to understand relationships between all parts of a sequence at once while encoding the position of each word or token in the sequence. An illustration of this model can be seen in Figure 8.

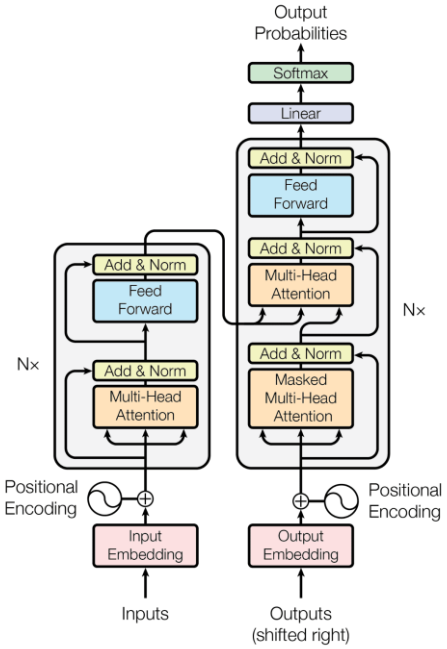

Figure 8. *The architecture of the Transformer model [8].*

The Transformer architecture includes multiple encoders and decoders. Each encoder has two main components: a self-attention layer, which focuses on relationships within the input, and a feed-forward neural network layer. Each decoder adds another layer for connecting with the encoders' output. Multi-Head Attention, a feature of the Transformer, allows it to consider multiple aspects of the input simultaneously, improving its understanding of the data [8].

In the final stage, the model converts the processed data into a prediction. It uses a linear layer to generate scores for all possible words, followed by a softmax layer to turn these scores into probabilities. The word with the highest probability is selected as the output. This parallel structure and use of attention make Transformers highly effective for tasks involving long and complex sequences [6].

3.2. Evaluation Metrics

For classification, these metrics are defined based on the relationships between the predicted values and the ground truth:

- **Accuracy**: The proportion of correctly classified data points among all the data points.

$$\text{Accuracy} = \frac{(TP + TN)}{(TP + FP + TN + FN)}$$

- **Precision**: The proportion of correctly classified data points among all the predictions.

$$Precision = \frac{TruePositives(TP)}{TruePositives(TP)+FalsePositives(FP)}$$

- **Recall**: The proportion of correctly data points among all ground truth objects.

$$\textbf{Recall} = \frac{TruePositives(TP)}{TruePositives(TP)+FalseNegatives(FN)}$$

- **F1 Score**: The harmonic mean of precision and recall.

$$\boldsymbol{F1 = 2 \times \frac{Precision \times Recall}{Precision + Recall}}$$

3.3. Procedure

For the financial sentiment analysis experiments, training was conducted using a variety of sequential models, including Recurrent Neural Networks (RNNs), Long Short-Term Memory Networks (LSTMs), and transformers. The models were trained and evaluated on a dataset of 5,842 financial sentences, tokenized using the BERT tokenizer.

Each model was trained for 200 epochs using an NVIDIA T4 GPU on Google Colab. The training and validation datasets were split at an 80:20 ratio, and the batch size was set to 32. The BERT tokenizer was used to preprocess the data, generating input tokens and attention masks with a maximum sequence length of 150 tokens, as determined by the dataset's characteristics.

For optimization, the Adam optimizer was used with a learning rate of 1e-5, and gradient clipping was applied with a maximum norm of 1.0 to stabilize training. The crossentropy loss function was employed as the criterion for multi-class classification, given the three sentiment categories (positive, neutral, and negative).

During training, the models were evaluated at regular intervals to monitor progress:

- **Training Loss.** Computed after each training step to track model convergence.
- **Validation Metrics**. Precision, recall, F1-score, and accuracy were measured for each sentiment category at the end of every epoch to assess model performance on unseen data.

4. RESULTS AND ANALYSIS

4.1. Recurrent Neural Networks

The training loss over steps is shown in Figure 9. The loss decreases consistently as training progresses, indicating that the model is effectively learning the patterns in the dataset. The stabilization of the loss toward the later steps suggests that the model converges successfully, minimizing the error in predictions. While some spikes in the training loss are observed, likely due to batch-specific variations, they are quickly resolved, and the general downward trend confirms steady learning.

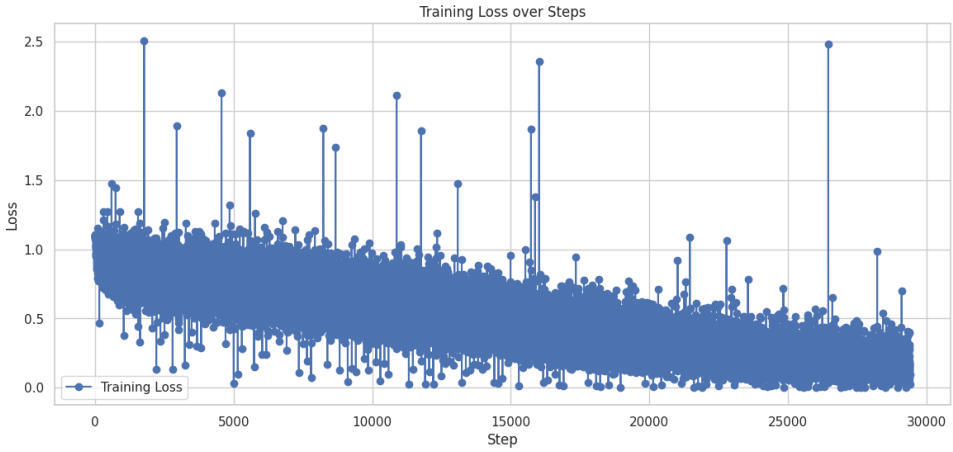

Figure 9. *Recurrent Neural Network train loss over train steps.*

Figure 10 depicts the accuracy of the RNN model across 200 epochs. Initially, the accuracy increases sharply, reflecting the model's rapid learning during the early epochs. However, after reaching a peak near 50 epochs, the accuracy begins to oscillate and gradually decline. This pattern could indicate overfitting, where the model starts to memorize the training data at the expense of generalizing to unseen validation data. The decline in accuracy over extended epochs emphasizes the importance of employing techniques such as early stopping to prevent overfitting and ensure optimal performance.

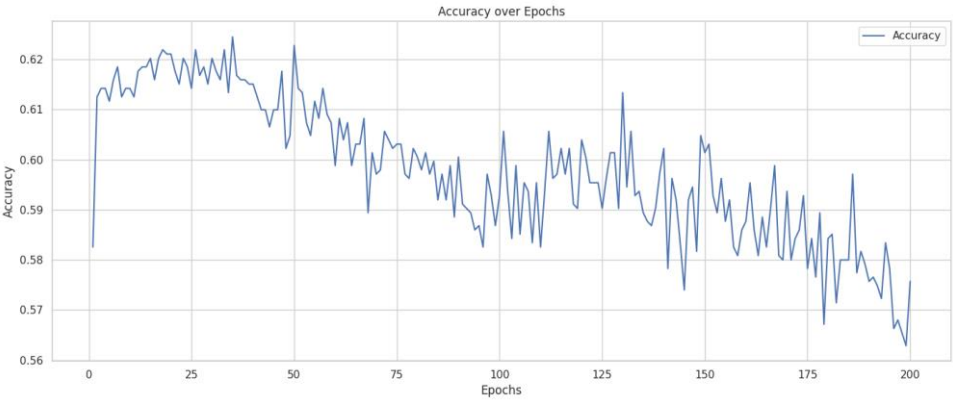

Figure 10. *Recurrent Neural Network accuracy over train epochs.*

The validation metrics (precision, recall, and F1-score) for the three sentiment classes are shown in Figure 11. For the “negative” class, all three metrics improve steadily during the first 100 epochs but plateau and exhibit fluctuations thereafter. This suggests that while the model captures the negative sentiment effectively early in training, its ability to refine predictions stagnates over time. For the “neutral” class, precision and recall show a gradual decline after an initial peak, whereas the F1-score stabilizes, reflecting a trade-off between precision and recall. The “positive” class displays consistent improvement in all metrics throughout the training process, with precision, recall, and F1-score converging to relatively high and stable values. This indicates the model's capability to handle the positive sentiment class more reliably compared to the other classes.

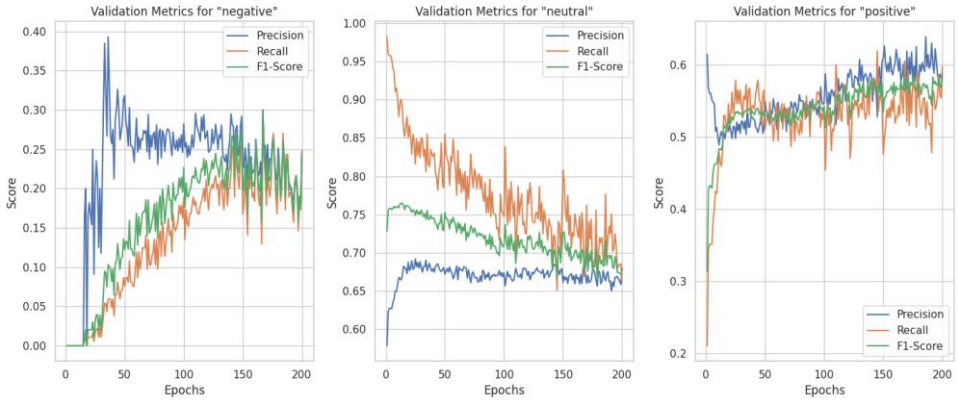

Figure 11. *Recurrent Neural Network classification metrics over train epochs.*

Overall, these figures highlight the strengths and limitations of the RNN in financial sentiment analysis. While the model demonstrates robust learning in the early stages and performs well for the "positive" class, the declining trends in accuracy and fluctuations in validation metrics for other classes suggest areas for improvement. Potential solutions include implementing regularization techniques, fine-tuning hyperparameters, or exploring more advanced architectures to enhance stability and generalization. These observations provide valuable insights into the RNN's behavior and inform strategies for optimizing its performance in financial sentiment analysis.

4.2. Long Short-Term Memory Results

Figure 12 illustrates the training loss over steps for the LSTM model. The loss decreases steadily throughout the training process, showing a smooth downward trend. This consistent reduction in loss indicates that the model effectively learns the underlying patterns within the dataset. The absence of significant spikes or plateaus further suggests that the training process was stable, and the LSTM was able to converge without major interruptions caused by batch-specific variations. The stabilization of the loss toward the end of the training indicates that the model has reached its capacity to minimize the error on the training data.

Figure 12. *Long Short-Term Memory train loss over train steps.*

Figure 13 shows the accuracy of the LSTM model over 200 epochs. The accuracy increases rapidly in the initial epochs, reflecting the model's quick adaptation to the dataset. After reaching a peak near 70 epochs, the accuracy exhibits slight fluctuations but remains stable. Unlike the RNN results, the accuracy for the LSTM does not decline significantly in later epochs, suggesting that the LSTM's gating mechanisms help mitigate overfitting by selectively retaining relevant information and

discarding irrelevant data. The sustained high accuracy indicates that the LSTM generalizes well across the dataset, performing reliably on both training and validation samples.

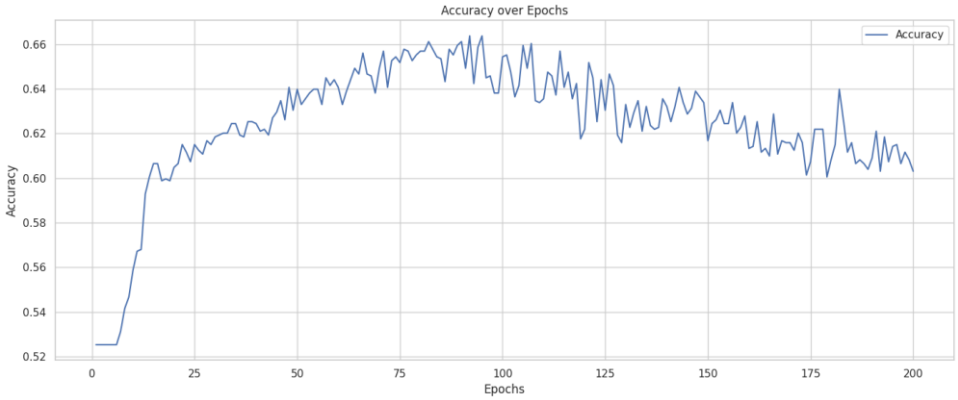

Figure 13. *Long Short-Term Memory accuracy over train epochs*.

The validation metrics for the three sentiment classes (negative, neutral, and positive) are presented in Figure 14. For the "negative" class, precision, recall, and F1-score gradually improve over the first 100 epochs, reaching a plateau with slight fluctuations. This performance suggests that the model effectively captures the negative sentiment but struggles to further refine its predictions. The "neutral" class metrics, while initially strong, show minimal improvement after the initial 50 epochs, maintaining stability throughout. This reflects the dominance of the neutral class in the dataset, which the LSTM handles with ease due to its ability to model sequential dependencies effectively. For the "positive" class, precision, recall, and F1-score show consistent improvement and reach their highest levels by the end of the training process. This result demonstrates the LSTM's proficiency in learning the distinct patterns of the positive sentiment class.

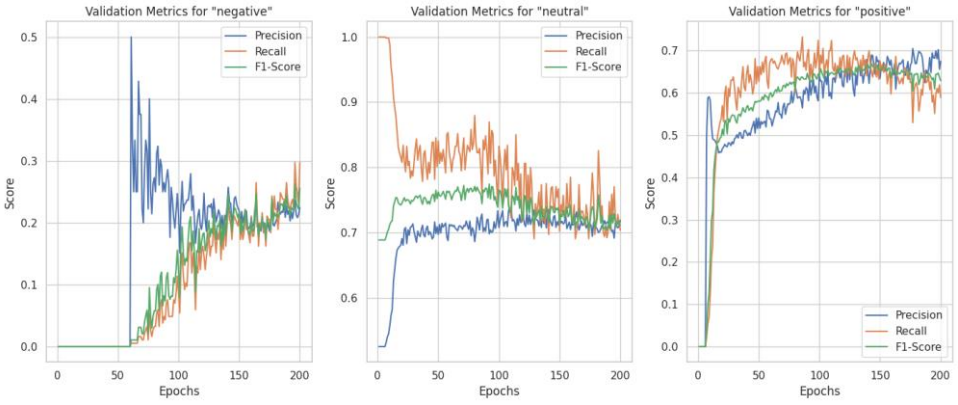

Figure 14. *Long Short-Term Memory classification metrics over train epochs.*

4.3. Transformer Results

Figure 15 presents the training loss over steps for the Transformer model. The training loss decreases in a constant way over the course of the training, indicating that the model effectively learns from the dataset. However, there are noticeable spikes in the loss at certain intervals, which may be attributed to specific challenging batches within the dataset. Despite these spikes, the overall downward trend and eventual stabilization suggest that the Transformer model converges effectively, minimizing errors on the training data.

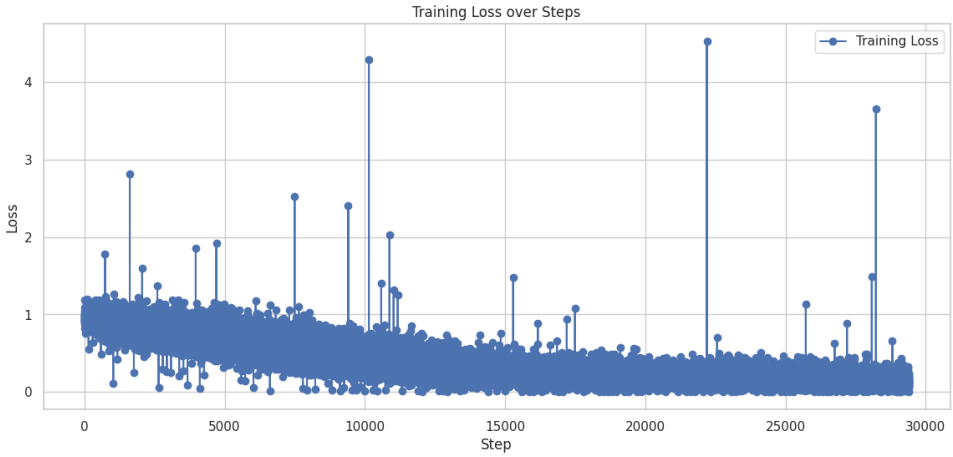

Figure 15. *Transformer train loss over train steps.*

Figure 16 shows the accuracy of the Transformer model across 200 epochs. The accuracy increases rapidly during the initial epochs, demonstrating the model's ability to capture the patterns in the data early in the training process. After reaching a peak near 100 epochs, the accuracy stabilizes with minimal fluctuations. This stability indicates that the Transformer model generalizes well without significant signs of overfitting, in contrast to simpler architectures that may suffer from performance degradation over extended training periods.

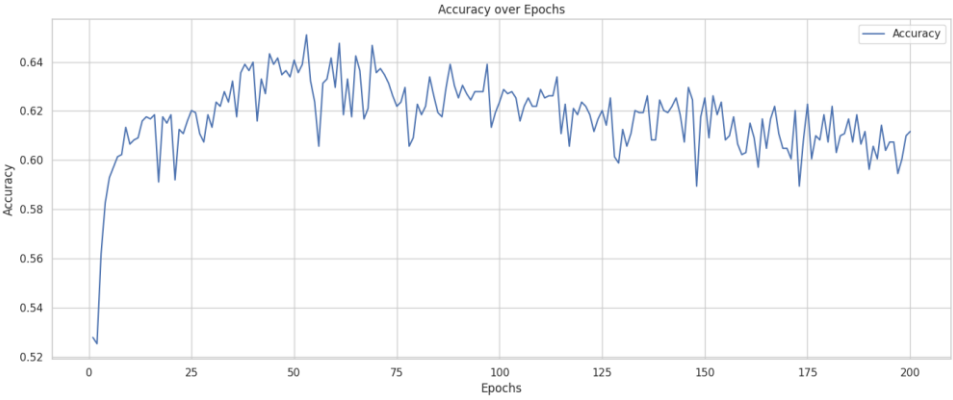

Figure 16. *Transformer train accuracy over train epochs.*

The validation metrics for the sentiment classes (negative, neutral, and positive) are presented in Figure 3. For the “negative” class, the precision, recall, and F1-score gradually improve during the first 100 epochs and stabilize thereafter, reflecting the model's ability to handle this underrepresented class effectively. For the “neutral” class, the metrics remain consistently high throughout training, with minor fluctuations. This is indicative of the Transformer model's capacity to handle the dominant class in the dataset without sacrificing performance on other classes. For the "positive” class, all metrics exhibit steady improvement, with precision, recall, and F1-score converging at relatively high levels by the end of training, showcasing the model's proficiency in capturing positive sentiment patterns.

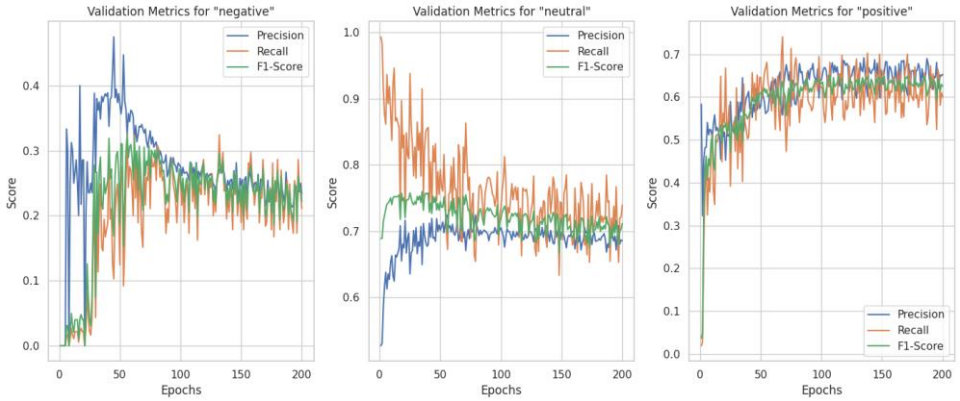

Figure 16. *Transformer classification metrics over train epochs.*

4.4. Comparison between the results of each dataset

The comparison across RNN, LSTM, and Transformer models from Table 1 reveals that the Transformer consistently outperforms the other architectures in accuracy and F1-scores across all sentiment classes. Its superior performance, particularly in handling positive (F1-score: 0.64) and neutral (F1-score: 0.77) sentiments, can be attributed to its attention mechanisms, which allow it to effectively capture complex contextual relationships. LSTM follows with balanced performance, benefiting from its ability to retain long-term dependencies, while the RNN struggles, especially with the negative sentiment class, due to its limited contextual understanding.

Neutral sentiment shows the highest metrics across all models, reflecting its dominance in the dataset, while negative sentiment remains the most challenging class, with relatively low precision, recall, and F1-scores. The Transformer demonstrates a clear advantage here as well but highlights the need for strategies like data augmentation or specialized class-balancing techniques to improve performance for underrepresented classes. Overall, the results affirm the increasing sophistication and effectiveness of modern architectures like Transformers in financial sentiment analysis.

Model	Sentiment	Precision	Recall	F1-Score	Accuracy
RNN	Positive	0.58	0.6	0.59	0.61
	Negative	0.26	0.24	0.25	
	Neutral	0.72	0.75	0.73	
LSTM	Positive	0.62	0.63	0.62	0.6
	Negative	0.3	0.28	0.29	
	Neutral	0.74	0.77	0.75	
Transformer	Positive	0.64	0.65	0.64	0.64
	Negative	0.32	0.31	0.31	
	Neutral	0.76	0.78	0.77	

Table 1. *Comparison of the metrics between all the 3 studied models*.

5. CONCLUSIONS AND FUTURE WORK

This study evaluated the performance of three sequential models - Recurrent Neural Networks (RNNs), Long Short-Term Memory Networks (LSTMs), and Transformers - for financial sentiment analysis. The results highlight the evolution in model capabilities, with Transformers achieving the highest accuracy and F1-scores across all sentiment classes, outperforming both LSTMs and RNNs. The Transformer model's ability to capture complex contextual dependencies, particularly for the positive and neutral sentiment classes, underscores its effectiveness in handling the nuanced language of financial texts. LSTMs, benefiting from their gating mechanisms, showed balanced performance but fell short of Transformers in terms of accuracy and handling underrepresented classes. RNNs demonstrated limited capacity to generalize effectively, particularly struggling with the negative sentiment class.

The findings emphasize the importance of advanced architectures in financial sentiment analysis, where the intricate nature of financial language requires models capable of understanding contextual and directional nuances. Future work should focus on enhancing performance for underrepresented sentiment classes, such as negative sentiment, through techniques like data augmentation and class-balancing strategies. Additionally, exploring hybrid approaches that combine domain-specific lexicons with state-of-the-art architectures may further improve sentiment classification in financial contexts.

Acknowledgement:
This work is the result of my own activity, and I confirm I have neither given, nor received unauthorized assistance for this work. I declare that I did not use generative AI or automated tools in the creation of content or drafting of this document.

REFERENCES

[1] Malo, Pekka, et al. "*Good debt or bad debt: Detecting semantic orientations in economic texts.*" Journal of the Association for Information Science and Technology 65.4 (2014): 782-796.

[2] Loughran, Tim, and Bill McDonald. "*When is a liability not a liability? Textual analysis, dictionaries, and 10-Ks.*" The Journal of finance 66.1 (2011): 35-65.

[3] Devlin, Jacob. "*Bert: Pre-training of deep bidirectional transformers for language understanding.*" arXiv preprint arXiv:1810.04805 (2018).

[4] Tetlock, Paul C. "*Giving content to investor sentiment: The role of media in the stock market.*" The Journal of finance 62.3 (2007): 1139-1168.

[5] Pascanu, Razvan, Tomas Mikolov, and Yoshua Bengio. "*On the difficulty of training Recurrent Neural Networks (2013).*" arXiv preprint arXiv:1211.5063.

[6] Schmidt, Robin M. "*Recurrent neural networks (rnns): A gentle introduction and overview.*" arXiv preprint arXiv:1912.05911 (2019).

[7] Hochreiter, S. "*Long Short-term Memory.*" Neural Computation MIT-Press (1997).

[8] Vaswani, A. "*Attention is all you need.*" Advances in Neural Information Processing Systems (2017).