ML Project Report

Léon, Kilian, Arthur, Lauryne, (Aziz)

June 10, 2025

1 Introduction

In financial markets, data is everywhere, but not all data looks the same. Traditionally, stock return prediction models rely on structured numerical data such as firm fundamentals and market factors. However, as the availability of unstructured data grows, there is increasing interest in exploring whether alternative signals–such as the content of corporate communications, might provide valuable insights. In this project we take a step in that direction : we investigate whether combining structured financial data with sentiment extracted from earnings calls can help improve quarterly stock returns predictions.

We construct and compare two models :

1. A baseline model that relies purely on structured firm characteristics, using *Principal Component Analysis* (PCA) to reduce dimensionality.
2. A multi-modal model that combines these PCA features with an additional signal: a sentiment score derived from earnings call transcripts, computed using the pre-trained financial language model FinBERT [1].

Our goal is to assess whether the textual sentiment signals extracted from earnings calls contain useful information for predicting quarterly stock returns, beyond traditional structured features.

This report is structured as follows. Section 2 describes the data preprocessing and feature engineering steps applied to both structured data and textual earnings calls transcripts. Section 3 introduces our LSTM-based modeling framework, detailing the architecture and training setup used to capture temporal patterns in the data. Section 4 presents our experimental results, comparing a baseline model using only structured features with an extended model incorporating sentiment signals derived from earnings calls.

2 Description of data preprocessing and feature engineering

2.1 Data preprocessing

Our project combines structured financial data and unstructured text data in a unified stock return prediction task. The dataset consists of three main components :

- **Quarterly Compustat Firm Characteristics** : firm-level accounting and market variables from Compustat and CRSP, originally provided at a monthly frequency.
- **Monthly Stock Returns** : monthly stock returns from CRSP (target variable)
- **Earnings calls** : transcripts of earnings calls, including both prepared remarks and Q&A sessions, timestamped at the monthly level

Temporal alignment. All three data sources are initially provided at a monthly frequency. However, firms do not necessarily report on exactly the same calendar, and earnings call dates can vary within each quarter. To ensure a consistent and aligned representation across firms and data sources, we aggregate all data to the quarterly level. For stock returns, we compute quarterly returns from the underlying monthly returns. For firm characteristics and earnings calls, we assign each observation to its corresponding quarter based on the observation date. This alignment guarantees that, for each quarter, we can merge the structured financial data, earnings call sentiment, and subsequent stock return in a consistent way.

Data cleaning and filtering. We only kept numerical and non constant columns from the *Compustat* dataset, dropping columns with more than 80% of missing values, filling the remaining missing values with a forward fill. We then retain only firms for which at least 8 consecutive quarters of data are available. This choice provides sufficient history for the sequence-based models we employ (See Section 3), while ensuring that the training and testing sets remain meaningful . We also apply winsorization[1] to the firm characteristics in order to mitigate the influence of extreme outliers.

2.2 Feature engineering

Our feature engineering process aims to extract informative signals from the available datasets while keeping the input dimension manageable for the LSTM model.

Temporal information. To help the model distinguish between seasonal effects and longer-term trends, we explicitly encode the quarter of the year. We create a one-hot encoding of the quarter ($Q1$, $Q2$, $Q3$, $Q4$), which is appended to the input features.

[1]Winsorization is applied at the 1% and 99% percentiles.

Motivation for dimensionality reduction. When analyzing the firm characteristics data, we observe that many features exhibit strong correlations. Figure 2.1 (Left panel) presents the correlation matrix of the original variables, highlighting substantial redundancy across features.

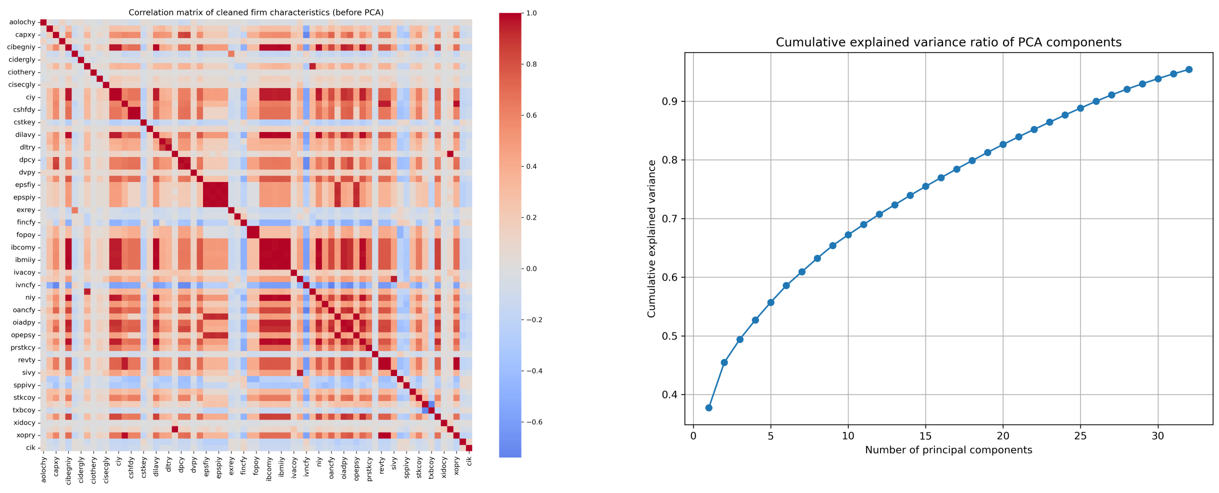

FIG. 2.1 – **Left panel:** Correlation matrix of original firm characteristics. **Right panel:** Cumulative explained variance ratio of PCA components.

PCA of firm characteristics. Before applying PCA, we split the dataset into training and testing sets (75/25). This ensures that no information from the test set leaks into the model during training. We then apply a feature standardization in order to prevent scale of features to influence our results. PCA is then fitted *only* on the training set, to avoid introducing test set information into the model. The same PCA transformation is applied to the test set. This approach allows us to summarize the most important patterns in the data while respecting the temporal separation between training and testing. We retain the first 32 principal components, which together explain approximately 95% of the variance in the original firm characteristics. The cumulative explained variance ratio across components is illustrated in Figure 2.1 (Right panel), confirming that the retained components capture most of the signal while enabling dimensionality reduction.

Sentiment signal from earnings calls. For each earnings call transcript, we compute a sentiment score using FinBERT [1]. FinBERT produces a soft score that reflects the model's estimate of the tone of the text. To make this signal more robust and interpretable, we map the average sentiment score to a discrete value: $+1$ for strongly positive sentiment, -1 for strongly negative sentiment, and 0 for neutral or weak signals. The threshold was empirically set at 0.3. The mapping is based on a predefined threshold applied to the average FinBERT score. This approach focuses the model on significant shifts in tone that are more likely to impact market reactions.

3 LSTM Models

Why we use LSTM. Initially, our plan was to approach the task using a *Multi-Layer Perceptron* (MLP) model, applied to the firm characteristics and sentiment signals. However, as we progressed, we recognized that the data exhibits a strong temporal structure: the return of a stock depends not only on the current state of the firm but also on its recent history. Since MLPs do not explicitly model temporal dependencies and have been shown to perform poorly on sequence modeling tasks [2], we decided to switch to *Long Short-Term Memory* (LSTM) networks [6], which are better suited for capturing patterns in sequences of past observations. This choice allows us to more effectively leverage the temporal dynamics present in the data.

3.1 Model Architecture

Our LSTM models are designed to predict a stock's return in the next quarter, based on its recent history. To capture these temporal dynamics, we feed the models sequences of past observations–specifically, rolling windows of 8 consecutive quarters.

Input features. For each quarter in the sequence, we provide the model with 32 PCA components summarizing the firm characteristics, 4 one-hot encoded quarter indicators, to help the model capture potential seasonality effects. In the second model, an additional sentiment score extracted from the firm's earnings call transcript. This results in an input of 36 features per quarter for the baseline model, and 37 for the augmented-sentiment model.

Sequence modeling. Each input consists of 8 such quarters of data for a given stock. The model processes this sequence using a LSTM layer.

Output. The final hidden state of the LSTM is passed through a *fully connected linear layer*, which produces a single scalar prediction : the forecasted return of the stock in the following quarter.

3.2 Training and Evaluation Setup

We split the dataset into training and testing sets based on time. The first 75% of available quarters are used for training, and the remaining 25% are reserved for testing. For each stock, we construct rolling windows of 8 consecutive quarters to serve as input sequences. The target for each sequence is the stock return in the quarter immediately following the window. Stocks with fewer than 8 consecutive quarters of data are excluded from the analysis. After preprocessing, the training set contains 32′507 rolling windows of 8 quarters, and the test set contains 1′776 such windows. Each quarter is represented by 36 features in the baseline model, or 37 features when adding the sentiment signal.

Training. The models are trained to minimize the *Mean Squared Error* (MSE) [5, 8] between the predicted and actual returns. We use the Adam optimizer with standard hyperparameters (`lr=1e-3`). Each model is trained for a fixed number of 50 epochs. We empirically observed that increasing the number of epochs beyond this point leads to overfitting : while the training loss continues to decrease, the test loss begins to increase, indicating reduced generalization. Throughout training, we monitor both training and test loss to evaluate performance and detect potential overfitting. The entire training and evaluation pipeline is implemented in PyTorch.

4 Model Results and Analysis

In this section, we present the results of our experiments. We first evaluate the baseline LSTM model trained only on structured firm characteristics. We then assess the impact of adding the sentiment signal derived from earnings calls. For both models, we report training and test losses and provide additional analysis to better understand their behavior and performance.

4.1 Model Construction

We implement two LSTM models trained to predict quarterly stock returns. Both models share a common sequence-based architecture consisting of an LSTM layer followed by a fully connected output layer, trained for 50 epochs using the Adam optimizer and the MSE loss. Each input sequence consists of 8 consecutive quarters of firm observations. The *baseline model* uses 36 input features per quarter, while the *sentiment model* adds a discrete sentiment score per quarter, increasing the input size to 37.

To ensure a fair comparison, all preprocessing, training settings and evaluation procedures are kept consistent across models.

4.2 Results and Comparison

We now present a direct comparison between the baseline LSTM model and the LSTM model augmented with sentiment signal. Both models share the same architecture : a 2-layer LSTM with 64 hidden units per layer, trained with the Adam optimizer and identical training/testing splits.

Table 1 summarizes the training dynamics and final test performance for both models. The training losses provide insight into the convergence behavior of each model, while the test metrics allow us to assess their ability to generalize to unseen data.

4.2 Results and Comparison

Metric	Baseline LSTM	Sentiment LSTM
Train Loss (Epoch 0)	0.0355	0.0352
Train Loss (Epoch 10)	0.0297	0.0287
Train Loss (Epoch 25)	0.0234	0.0222
Train Loss (Epoch 50)	0.0157	0.0132
Test Loss (MSE, Epoch 50)	0.0843	0.0790
Mean Absolute Error (MAE)	0.2112	0.2036
Root Mean Squared Error (RMSE)	0.2903	0.2811
Directional Accuracy	57.49%	57.49%

TABLE 1 – Training and test performance comparison of baseline and sentiment LSTM models.

We observe that both models converge steadily during training, with similar training loss trajectories. On the test set, the LSTM model with sentiment achieves slightly lower MSE, MAE and RMSE compared to the baseline. However the directional accuracy was the same, showing that we don't sacrifice this performance by adding the sentiment score. These results suggest that incorporating sentiment signals from earnings calls provides additional predictive value, helping the model better capture the direction and magnitude of future stock returns.

To further analyze model behavior, we include two plots for each model. The first compares predicted and annual returns, while the second displays the distribution of residuals. Figures 4.1 and 4.2 show the comparison between the two models.

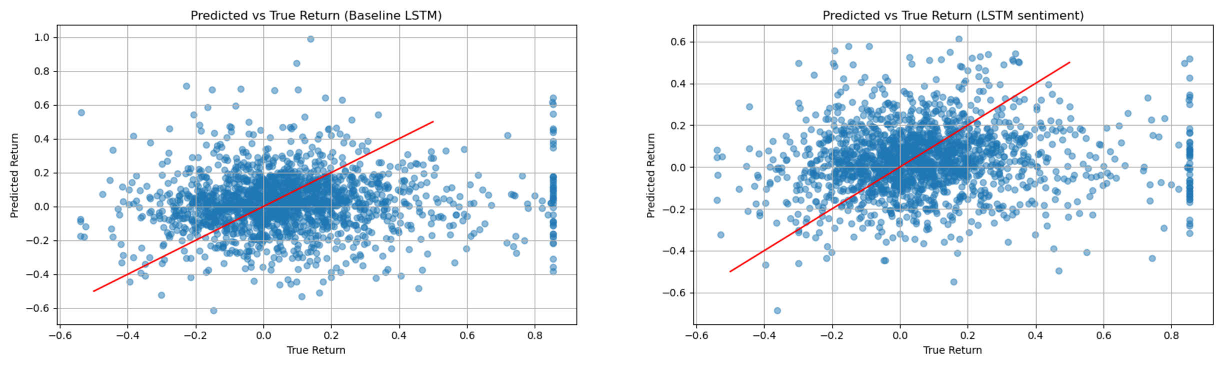

FIG. 4.1 – Predicted vs. true returns: comparison between baseline (left) and sentiment-augmented (right) LSTM models.

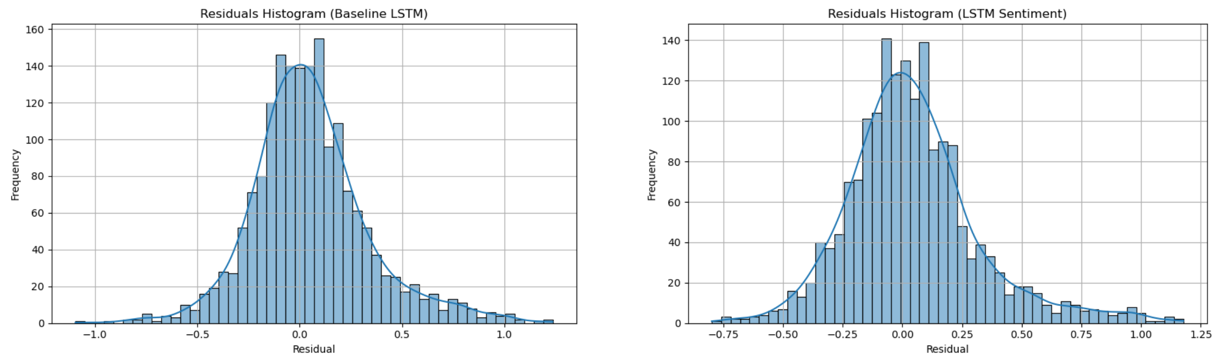

FIG. 4.2 – Residuals histogram: comparison between baseline (left) and sentiment-augmented (right) LSTM models.

Figure 4.1 compares the predicted vs. true returns for both models. The regression line in each panel illustrates the model's ability to capture directional trends. While both show a positive correlation between predicted and true returns, the sentiment-augmented model (right) exhibits slightly stronger alignment, with more predictions concentrated along the diagonal. On the other side, Figure 4.2 shows the distribution of residuals for both models. The histograms are approximately symmetric and bell-shaped, consistent with a well-behaved error distribution. The sentiment model (right) displays slightly tighter dispersion and thinner tails, indicating a modest improvement in predictive consistency and robustness.

4.3 Limitations and Future Directions

While our results demonstrate that incorporating signals from earnings calls can marginally improve stock return predictions, several limitations remain in the current modeling approach.

Limitations. Despite the improvements we observe, our current approach still has several limitations. First, predicting stock returns remains a very challenging task, as they are influenced by many factors outside the scope of our models. Both the baseline and sentiment LSTMs show directional accuracies slightly above random ($\sim$ 57%), which suggests that they do capture useful patterns, but are still far from sufficient to make reliable predictions. Second, while we incorporate sentiment signals from earnings calls, our treatment of the text is still relatively coarse. The FinBERT [1] model we used operates on small chunks (due to token length limitations), so we had to compute an average sentiment score per call. This averaging may dilute variations in tone that occur across different parts of the call. Moreover, because these average sentiment scores produced by FinBERT [1] fall in a narrow numerical range (between -1 and 1), they would likely be treated by the LSTM as just another small numerical feature, potentially drowned out by the PCA components. To address this, we discretized the sentiment into three levels $(+1, 0, -1)$ to make the signal more salient. However, this transformation inevitably loses some nuance in the original text. Finally, our models only use firm-specific inputs. However, it is well established that stock returns are also influenced by broader market factors and macroeconomic conditions– as formalized, for example, in multi-factor models such as the Fama-French framework [4], which are not included in our feature set.

Future Directions. There are several ways in which this work could be extended and improved. First, as previously discussed, more advanced MLP techniques could be explored. Rather than summarizing entire earnings calls into a single sentiment score, we could use models that process the full sequences of text, such as transformer-based architectures. This would allow us to capture not only the overall tone, but also how sentiment evolves during the call, and which parts of the discussion might be most predictive of returns. Another improvement would be to enrich the input data, maybe with market sentiment from news articles, or even sector-level variables. Prior work [3, 7] has shown that combining multiple data sources can improve stock return prediction by providing a more comprehensive representation of the factors influencing returns. Finally, we could explore more advanced model architectures. In this project we used a simple LSTM as our main model, but architectures such as Transformer networks [9] could potentially improve the model's ability to capture complex temporal patterns in the data.

References

[1] D Araci. *FinBERT: A Pretrained Language Model for Financial Communications*. 2019. arXiv: 1908.10063 [cs.CL].

[2] Shaojie Bai, J Zico Kolter **and** Vladlen Koltun. "An Empirical Evaluation of Generic Convolutional and Recurrent Networks for Sequence Modeling". **in***arXiv preprint arXiv:1803.01271*: (2018).

[3] Xiaodong Ding **andothers**. "Deep learning for event-driven stock prediction". **in***Proceedings of the 24th International Joint Conference on Artificial Intelligence (IJCAI)*: 2015, **pages** 2327–2333.

[4] Eugene F. Fama **and** Kenneth R. French. "Common risk factors in the returns on stocks and bonds". **in***Journal of Financial Economics*: 33.1 (1993), **pages** 3–56.

[5] Thomas Fischer **and** Christopher Krauss. "Deep learning with long short-term memory networks for financial market predictions". **in***European Journal of Operational Research*: 270.2 (2018), **pages** 654–669.

[6] Sepp Hochreiter **and** Jürgen Schmidhuber. "Long short-term memory". **in***Neural computation*: 9.8 (1997), **pages** 1735–1780.

[7] Zongcheng Hu, Xiaodong Zhang **and** Stephen Teo Swee Chia. "Listening to the news: News sentiment and economic indicators". **in***Knowledge-Based Systems*: 160 (2018), **pages** 1–15.

[8] David M Nelson, Adriano C M Pereira **and** Renato A de Oliveira. "Stock market's price movement prediction with LSTM neural networks". **in***2017 International Joint Conference on Neural Networks (IJCNN)*: (2017), **pages** 1419–1426.

[9] Haoyi Zhou **andothers**. "Informer: Beyond Efficient Transformer for Long Sequence Time-Series Forecasting". **in***CoRR*: abs/2012.07436 (2020). arXiv: 2012.07436. URL: https://arxiv.org/abs/2012.07436.