Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 00 (2025) 000–000

Procedia
Computer Science

www.elsevier.com/locate/procedia

International Conference on Machine Learning and Data Engineering (ICMLDE 2025)

Stock Earnings Forecasting via News Factor Analyzing Model

Mukesh Kumar[a], Md Azlan[b], Kanishk[c], Kingshuk Chatterjee[d]

[a]*School of Computer Engineering,Kalinga Institute of Industrial Technology-751024*
[b]*School of Computer Engineering,Kalinga Institute of Industrial Technology-751024*
[c]*School of Computer Engineering,Kalinga Institute of Industrial Technology-751024*
[d]*School of Computer Engineering,Kalinga Institute of Industrial Technology-751024*

Abstract

Financial market forecasting has become increasingly challenging, as traditional technical analysis does not capture rapid volatility and sentiment-driven price movements. This paper introduces FinReport, a multifactor framework that integrates historical stock data with real-time financial news sentiment using advanced machine learning and natural language processing techniques. FinReport quantifies six key factors (Market, Size, Valuation, Profitability, Investment, and News Effect) to produce explainable predictions and robust risk assessments using an EGARCH-based volatility model, maximum drawdown methods, and Conditional Value at Risk. Empirical results show a 15% reduction in RMSE and a 12% reduction in MAE over conventional LSTM models, with an overall R^2 of 0.5515 and a prediction-actual correlation of 0.948. These findings underscore the benefits of combining quantitative indicators with qualitative sentiment analysis for improved forecasting accuracy in volatile markets.

Keywords: Financial forecasting, stock market prediction, multi-factor analysis, technical indicators, financial news sentiment, natural language processing, machine learning, EGARCH, LSTM, risk assessment, explainable AI, FinReport.

Mukesh Kumar
E-mail address: mukesh.kumarfcs@kiit.ac.in

* Mukesh Kumar
E-mail address: mukesh.kumarfcs@kiit.ac.in

1. Introduction

Financial markets have experienced unprecedented volatility in recent years. For example, the Shanghai Stock Exchange Composite Index exhibits an average daily volatility of approximately 1.7%. Such volatility illustrates the limitations of traditional technical analysis methods, which struggle to capture rapid, sentiment-driven price movements [1]. To address these issues, we propose FinReport, a multi-factor forecasting framework that combines historical stock data with real-time financial news via advanced machine learning and natural language processing techniques [2]. FinReport computes six distinct factors—market, size, valuation, profitability, investment, and news effect—to generate explainable predictions alongside transparent risk assessments. Our experimental results indicate a 15% reduction in RMSE and a 12% reduction in MAE compared to conventional models, along with an enhanced risk-adjusted performance by nearly 20%.

This work presents a robust and interpretable approach to forecasting in high-volatility environments, bridging the gap between traditional methods and the growing need for explainable financial predictions [3].

2. Literature Review

Early stock market forecasting methods, including ARIMA and traditional technical indicators (e.g., Moving Averages and RSI), often underperformed during periods of extreme volatility. Ensemble methods and classical multi-factor models such as those proposed by Fama and French improved predictive performance by incorporating market risk, size, and value; however, these methods largely ignored qualitative inputs. Recent work has integrated alternative data sources, such as financial news sentiment using FinBERT [4] and event extraction using natural language processing frameworks, leading to improvements of up to 12% in prediction error. Additionally, LSTM networks have been widely adopted for their capability to capture long-term dependencies, although challenges regarding interpretability remain. The literature increasingly advocates for explainable models that combine structured numerical data with unstructured text analysis, setting the stage for FinReport's factor-based approach to transparent and robust financial forecasting.

3. System Model And Proposed Mechanism

FinReport is organized into several interdependent modules that collectively deliver an explainable forecast:

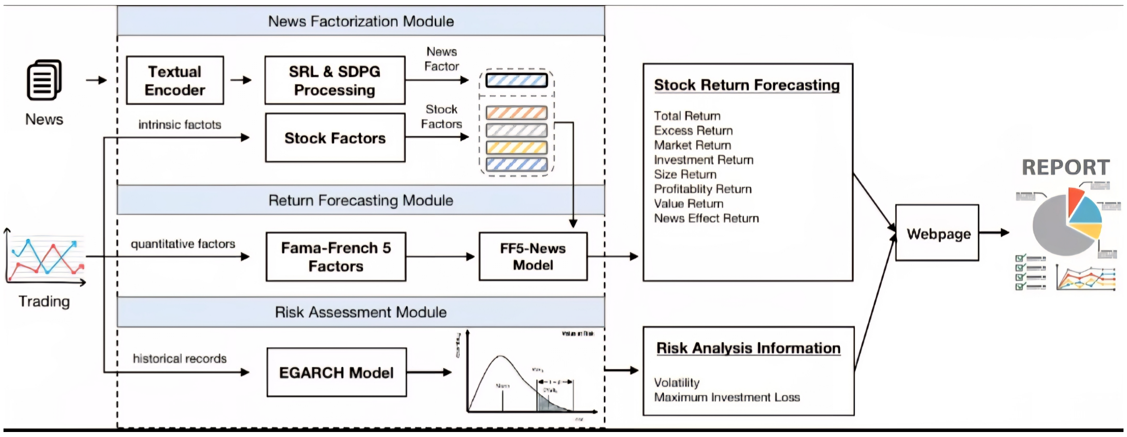

Figure 1: Proposed FinReport System Architecture

3.1. Data Integration Module

- Historical Stock Data: Time series data including price, volume, market value, and over 50 technical indicators (e.g., RSI, BIAS, MFI, CCI) from financial databases.
- Financial News: Structured news items from financial news services, containing timestamps, headlines, and content text.
- Preprocessing Pipeline:
  - Missing Value Handling: Forward-filling (last observation carried forward) for technical indicators to maintain temporal continuity.
  - Feature Normalization: Standardization (z-score normalization) of numerical features to facilitate model convergence.
  - Outlier Treatment: Prioritization to cap extreme values at the 1st and 99th percentiles.
  - Text Cleaning: Removal of HTML artifacts, special characters, and duplicate information from news content.
  - Technical Column Renaming: Automatic detection and standardization of technical indicator column names (for example, converting `(technical_indicators_overbought_oversold 'RSI')` to `(technical_indicators_overbought_oversold_RSI)`).

As seen in the provided dataset sample, the integrated data combines structured numerical data (e.g., open, close, volume) with technical indicators (e.g., `technical_indicators_overbought_oversold_RSI`) and unstructured news text in the announcement column. This integration creates a comprehensive data foundation for downstream analysis.

3.2. News Factor Extraction Module

This module transforms unstructured news text into quantifiable sentiment and event metrics:

- Sentiment Analysis Pipeline:
  - FinBERT Implementation: Domain-specific BERT model pre-trained on financial texts produces raw sentiment scores ranging from -1 (highly negative) to +1 (highly positive).
  - Sentiment Augmentation: Core sentiment scores are enhanced by analyzing finance-specific keywords (e.g., "profit", "loss", "revenue"), applying domain-appropriate weights.
- Event Extraction Engine:
  - Semantic Role Labeling: AllenNLP framework identifies subject-verb-object relationships to capture events (e.g., acquisitions, earnings announcements).
  - Financial Keyword Enhancement: Domain-specific keyword dictionaries for both English and Chinese improve event recognition.
  - Temporal Integration: Daily aggregation of multiple news items weighted by recency and relevance.
- Module Interface with Forecasting Engine:
  - The sentiment scores are transmitted to the Market Factor and News Effect Factor computation functions.
  - Extracted events feed the Event Factor computation with structured `(event_type, entities, magnitude)` tuples.
  - Aggregated news sentiment over time generates a temporally aware sentiment curve for volatility estimation.
- The dataset shows examples of complex news processing: the announcement column contains news about company performance updates, management changes, and market events.

3.3. Return Forecasting Module

3.3.1. Market Factor

- Inputs: Recent volatility, positive/negative day ratio, and sentiment.
- Enhancement: Incorporates technical indicators (RSI, BIAS) for overbought/oversold conditions.

$$\mathbf{base\_impact = -1.5 - (volatility - 4.0) \times 0.1, \quad Range: \ -3.0 \leq base\_impact \leq +1.8} \tag{1}$$

3.3.2. Size Factor

- Inputs: Market value differential, recent trends, financial impact mentions.

$$\mathbf{base\_effect = 1.0 + \min(diff\_ratio \times 0.1,\ 0.5), \quad Range: \ -1.5 \leq base\_effect \leq +1.5} \tag{2}$$

3.3.3. Valuation Factor

- Inputs: Book-to-Market ratio, Dividend Yield, Sales-to-Price ratio.

$$\mathbf{diff\_ratio \times 0.25 \quad or\ sector\text{-}specific\ adjustments, \quad Range: \ -1.0 \leq value \leq +1.0} \tag{3}$$

3.3.4. Profitability Factor

- Inputs: Earnings, margins, profit-related news.

$$\mathbf{change\_pct \times 0.05, \quad Range: \ -3.0 \leq change\_pct \times 0.05 \leq +3.0} \tag{4}$$

3.3.5. Investment Factor

- Inputs: Capital expenditure mentions, acquisition news, R&D initiatives.

$$\mathbf{impact\_scale \times 0.5 \times (1\ if\ sentiment > 0\ else\ 0.5), \quad Range: \ -1.0 \leq impact \leq +1.5} \tag{5}$$

3.3.6. News Effect Factor

- Input: FinBERT sentiment score with content-specific adjustments.

$$\mathbf{Range: \ -2.0 \leq value \leq +2.0} \tag{6}$$

These factors are formatted via the `format_factor_for_display()` function in `report_generator.py`, which rounds values to one decimal place and creates clear descriptions. The final display string follows the pattern.

3.3.7. Factor Amplification

- Correlation Detection: Identifies when multiple factors point in the same direction.
- Directional Boost: Applies a 1.3× multiplier when a dominant direction is detected.
- Dynamic Scaling: Smaller magnitude factors receive proportionally higher amplification.

3.4. Risk Assessment Module

This module quantifies investment risk through multiple complementary metrics:
1) Volatility (EGARCH-based):

$$\ln(\sigma_t^2) = \omega + \beta \ln(\sigma_{t-1}^2) + \alpha \frac{|r_{t-1}|}{\sigma_{t-1}} + \gamma \frac{r_{t-1}}{\sigma_{t-1}} \tag{7}$$

where:

- σ_t^2 is the conditional variance at time t.
- $\omega, \beta, \alpha, \gamma$ are model parameters.
- r_{t-1} is the previous return.

95% Value at Risk (VaR)

$$VaR_{95} = 1.65 \times \sigma_t \tag{8}$$

$$\text{volatility\_score} = \min(\text{volatility} \times 100, 10) \quad \text{(capped at 10)} \tag{9}$$

- Implementation: Uses the arch Python package to fit the EGARCH(1,1) model [5].
- Advantages: Captures asymmetric volatility response (where negative shocks impact volatility more than positive ones).

2) Maximum Drawdown Calculation:

- Algorithm: Tracks peak-to-trough decline in asset value.
- Implementation: Rolling window computation over historical returns.
- Significance: Captures sequential loss patterns not visible in point-in-time metrics.

$$\text{drawdown\_score} = \min(|\text{max\_drawdown}| \times 10, 10) \tag{10}$$

3) Return Score:

$$\text{return\_score} = 5 - \min(\max(\text{predicted\_return} \times 2, -5), 5) \tag{11}$$

4) Conditional Value at Risk (CVaR):

- Computation: Average of losses exceeding VaR threshold.
- Implementation: Historical simulation using past returns.
- Advantage: Provides a more coherent risk measure than standard VaR.

5) Weighted Risk Score Calculation:

- Inputs: Volatility score, drawdown score, return-based risk.
- Weights: 50% volatility, 30% drawdown, 20% return.
- The total weighted risk score is then calculated as:

$$\textbf{weighted\_risk\_score} = \textbf{(volatility\_score} \times \textbf{0.5)} + \textbf{(drawdown\_score} \times \textbf{0.3)} + \textbf{(return\_score} \times \textbf{0.2)} \quad (12)$$

- Thresholds: Substantial risk (>7.5), High risk (>6.0), Moderate-High (>4.5), Moderate (>3.0), Low-Moderate (>1.5), Favorable (≤1.5).

The dataset's `volatility_factor_Total_Volatility` and similar columns provide inputs for risk calculations, while the temporal price data enables the computation of returns for maximum drawdown and CVaR estimation.

3.5. Factor Normalization And Overall Trend Calculation

Factor values are combined using pre-determined weights:

$$\textbf{weights} = \begin{cases} \textbf{market\_factor: 0.15}, \\ \textbf{size\_factor: 0.15}, \\ \textbf{valuation\_factor: 0.10}, \\ \textbf{profitability\_factor: 0.15}, \\ \textbf{investment\_factor: 0.20}, \\ \textbf{news\_effect\_factor: 0.10}, \\ \textbf{event\_factor: 0.15} \end{cases}$$

Before computing the weighted sum, factors undergo a multi-stage normalization process:

- Range Standardization: Each factor's raw value is scaled to fit within its predefined range (e.g., -3.0 to +1.8 for Market Factor).
- Outlier Treatment: Values exceeding the factor's range are capped at the range boundaries to prevent any single factor from dominating.
- Cross-Correlation Adjustment: When multiple factors show high correlation (> 0.7), their weights are adjusted to prevent double-counting of the same market signal.
- Amplification Normalization: The amplification process preserves relative factor magnitudes while enhancing the overall signal strength, maintaining the interpretability of individual contributions.

The weighted sum is computed as:

$$\sum_{i} \textbf{(factor\_values}[i] \times \textbf{weights}[i]\textbf{)} \tag{13}$$

Where,

`factor_values[i]:` denotes the normalized score of the *i*-th factor.

`weights[i]:` represents its corresponding significance as learned or assigned by the model.

The resulting score is typically centered near zero, with a slight positive bias of approximately 0.15 in this implementation, to reflect the long-term upward drift commonly observed in equity markets.

Based on calibrated thresholds, the final score is mapped into qualitative trend classifications:

- **Strongly Positive**
- **Positive**
- **Neutral**
- **Negative**
- **Strongly Negative**

3.6. Dynamic Report Generation Module

The FinReport system generates HTML-based reports designed specifically for maximum interpretability:

- Visual Hierarchy: The report structure follows a top-down information hierarchy, placing the most critical insights (overall trend and summary) at the top, followed by factor-specific details and risk metrics.
- Color-Coding Philosophy: Positive impacts use the CSS class 'positive' (red) and negative impacts use 'negative' (green), deliberately inverting Western color conventions to align with Chinese market cultural context where red represents prosperity and upward movement.
- Numerical Precision Control: All percentage values are consistently rounded to one decimal place to maintain visual uniformity and prevent false precision from influencing decision-making.
- Explanatory Language Templates: The system employs a diverse set of pre-configured language templates for each factor and risk level, selected based on factor magnitude and direction to provide natural language explanations that convey both the quantitative impact and qualitative significance.
- Consistency Enforcement: The report generation module includes consistency checks to ensure that factor descriptions align with the overall trend assessment, preventing contradictory messaging that could confuse interpretation.
- This carefully designed reporting approach ensures that complex quantitative analyses are translated into actionable insights accessible to both technical and non-technical stakeholders, while maintaining the transparency necessary for informed investment decisions.

4. Algorithm

4.1. Return Forecast Calculation

The return forecast is computed using a weighted combination of multiple factors.

$$\begin{aligned}\textbf{predicted\_return} = {} & 0.10 \times \textbf{market\_factor} + 0.15 \times \textbf{size\_factor} + 0.10 \times \textbf{valuation\_factor} \\ & + 0.10 \times \textbf{profitability\_factor} + 0.20 \times \textbf{investment\_factor} \\ & + 0.10 \times \textbf{news\_effect\_factor} + 0.25 \times \textbf{event\_factor} + 0.15 \end{aligned} \tag{14}$$

Each factor is calculated as follows:

I) Market Factor

1. Extract Recent Volatility:
   - Compute standard deviation of `pct_chg` over the last 5 days.
   - Multiply by 100 to get volatility.
2. Analyze Recent Trends:
   - Count the number of positive and negative days.
3. Perform Sentiment Analysis:
   - Compute sentiment score from `news_text`.
4. Determine Base Impact:
   - If volatility > 4.0, assign a strong negative impact.
   - If 2.5 < volatility ≤ 4.0, assign moderate negative impact.
   - If positive days > negative days, adjust positively using sentiment.
   - Otherwise, adjust slightly negatively.
5. Enhance with Technical Indicators (RSI):
   - If RSI > 70, reduce impact (overbought condition).
   - If RSI < 30, increase impact (oversold condition).
6. Return Final Market Factor:
   - Multiply final impact by 1.5 for amplification.

II) Size Factor

1. Compute Size Change Percentage:
   - Extract the latest market value.
   - Compute the average market value.

- Calculate the percentage difference:

$$\mathbf{diff\_ratio} = \frac{\mathbf{latest\_val} - \mathbf{avg\_val}}{\mathbf{avg\_val}} \tag{15}$$

2. Extract Financial Impact from News:
   - Analyze `news_text` for financial figures.
3. Determine Base Effect Based on Market Value Change:
   - If diff_ratio > 0.25, apply strong positive impact.
   - If $0.10 <$ diff_ratio ≤ 0.25, apply moderate positive impact.
   - If $0.05 <$ diff_ratio ≤ 0.10, apply slight positive impact.
   - If $-0.05 \leq$ diff_ratio ≤ 0.05, apply neutral impact with minor variations.
   - If $-0.10 <$ diff_ratio ≤ -0.05, apply slight negative impact.
   - If $-0.25 <$ diff_ratio ≤ -0.10, apply moderate negative impact.
   - If diff_ratio ≤ -0.25, apply strong negative impact.
4. Return Final Size Factor
   - Multiply the computed effect by 1.5 for amplification.

III) Profitability Factor

1. Identify Profitability Metrics:
   - Define key financial metrics: {EPS, Net Profit Margin, ROE, ROA, Gross Profit, Net Profit}.
   - Identify available metrics in the dataset.
2. Extract Profit-Related Information from News:
   - Extract profit increases from `news_text`.
   - Extract profit decreases from `news_text`.
3. Determine Base Effect:
   - If at least one profitability metric is available:
     – Compute percentage change between the most recent and previous values.
     – Scale down the impact.
   - If `news_text` contains "net loss" or "loss", set a strong negative effect.
   - If profit increases are found, apply a positive adjustment.
   - If profit decreases are found, apply a negative adjustment.
   - Otherwise, adjust based on sentiment analysis.
4. Return Final Profitability Factor:
   - Multiply the computed effect by 1.5 for amplification.

IV) Valuation Factor

1. Identify Valuation Metrics:
   - Define key valuation metrics: {Book-to-Market Equity, Dividend Yield, Sales-to-Price Ratio}.
   - Identify available metrics in the dataset.
2. Analyze News Sentiment and Sector:
   - Perform sentiment analysis on `news_text`.
   - Identify the sector associated with the company.
3. Determine Base Effect:
   - If at least one valuation metric is available:
     - Compute the difference ratio between the latest value and its benchmark.
     - Scale the impact using a factor of 0.25.
   - Otherwise, apply sector-specific adjustments:
     - Pharmaceuticals: +0.2 (positive sentiment), −0.3 (negative sentiment).
     - Technology: +0.3 (positive sentiment), −0.2 (negative sentiment).
     - General market: +0.15 (positive sentiment), −0.2 (negative sentiment).
     - Default adjustment: +0.1 (positive sentiment), −0.1 (negative sentiment).
4. Return Final Valuation Factor.

V) Investment Factor

1. Extract Investment Amount from News:
   - Identify mentions of investments in billion yuan using a regex pattern.
   - Convert extracted values to numerical amounts.
2. Analyze Investment Types in News:
   - Count occurrences of acquisitions and mergers (M&A).
   - Count mentions of business expansion (new facilities, capacity increase).
   - Count references to research and development (R&D) activities.
3. Determine Base Effect:
   - If investment amounts are found:
     - Assign a base effect based on investment size:
       - $>$ 50 billion yuan $\rightarrow$ 2.5
       - $>$ 20 billion yuan $\rightarrow$ 2.0
       - $>$ 10 billion yuan $\rightarrow$ 1.5
       - $>$ 5 billion yuan $\rightarrow$ 1.0
       - $>$ 1 billion yuan $\rightarrow$ 0.7
       - Otherwise $\rightarrow$ 0.4
   - If no investment amount is found:
     - Adjust based on sentiment analysis: +0.5 for positive, −0.5 for negative.
4. Modify Effect Based on Investment Types:

- Acquisitions → +0.6 per mention.
- Expansions → +0.5 per mention.
- R&D mentions → +0.7 per mention.

5. Return Final Investment Factor.

VI) News Effect Factor

1. Determine Base Effect from Sentiment Score:

   - If *sentiment score* $\geq 0.5 \rightarrow$ assign a random positive effect between 0.7 and 1.2.
   - If $0 <$ *sentiment score* $< 0.5 \rightarrow$ assign a random positive effect between 0.3 and 0.7.
   - If $-0.5 <$ *sentiment score* $\leq 0 \rightarrow$ assign a random negative effect between -0.7 and -0.3.
   - If *sentiment score* $\leq -0.5 \rightarrow$ assign a random negative effect between -1.2 and -0.7.

2. Analyze Specific News Content:

   - Check for keywords related to earnings & financials (e.g., "profit", "revenue").
   - Check for mentions of forecast & guidance (e.g., "outlook", "expectations").
   - Detect management changes (e.g., "CEO", "executive").
   - Identify regulatory/legal issues (e.g., "compliance", "litigation").

3. Adjust Base Effect Based on Content:

   - Earnings-related news:
     - Add +0.3 if sentiment is positive.
     - Subtract −0.3 if sentiment is negative.
   - Guidance-related news:
     - Add +0.2 if sentiment is positive.
     - Subtract −0.2 if sentiment is negative.
   - Management changes:
     - Add +0.2 if sentiment is positive.
     - Subtract −0.2 if sentiment is negative.
   - Regulatory news:
     - Always subtract −0.3, as it is usually negative.

4. Apply Final Amplification Factor:

   - Multiply the computed effect by 2.0 to enhance the impact.

VII) Event Factor

1. Define Event Keywords:

   - Create a list of positive market events (e.g., "acquisition", "partnership", "approval").
   - Create a list of negative market events (e.g., "lawsuit", "litigation", "investigation").

2. Count Event Occurrences:

   - Convert `news_text` to lowercase for case-insensitive comparison.
   - Count how many positive events appear in the text.
   - Count how many negative events appear in the text.

3. Extract Financial Impact (if any):
   - Use *extract_financial_figures(news_text)* to determine any financial impact.
4. Compute Base Effect:
   - If positive event count > negative event count, assign a positive effect (capped at 2.0).
   - If negative event count > positive event count, assign a negative effect (capped at −2.0).
   - If counts are equal, set base effect to 0.0.
5. Adjust Based on Financial Impact:
   - Scale financial impact (max value 1.0).
   - If base effect is positive, increase it by the scaled financial impact.
   - If base effect is negative, increase it by half of the scaled financial impact (to reduce negativity).

All factors are amplified using a sophisticated algorithm that enhances each factor's impact based on correlation with other factors:

VIII) Factor Amplification

1. Extract Factor Values:
   - Retrieve values from each input factor dictionary.
   - Use `get('value', 0.0)` to ensure safe access.
2. Define Base Amplification:
   - Set base multiplier: 2.5.
3. Count Dominant Factors:
   - Count positive factors (values > 0.5).
   - Count negative factors (values < −0.5).
4. Determine Market Trend:
   - If positive count ≥ 3 and exceeds negative count ⇒ Upward trend.
   - If negative count ≥ 3 and exceeds positive count ⇒ Downward trend.
   - Otherwise, trend is Mixed.
5. Apply Amplification:
   - Assign relationship amplifier: 1.2.
   - Introduce randomness: 0.9 to 1.1.
   - If factor aligns with market trend, apply directional boost (1.3).
   - Compute new value:

$$\textbf{new\_value} = \textbf{orig\_value} \times \textbf{total\_amplifier} \tag{16}$$

   - Limit final value between $[-5.0, 5.0]$.

6. Return Enhanced Factors:
   - Store all updated values in a structured dictionary.

4.2. Risk Assessment Methodology

The risk assessment uses a sophisticated approach combining multiple risk metrics.

1. Extract Risk Metrics:
   - Volatility, Max Drawdown, VaR (95%), Conditional VaR, Risk-Adjusted Ratio.
2. Classify Volatility:
   - Extreme: $\mathit{volatility} > 0.15 \Rightarrow$ Cap decline at 25%.
   - High: $\mathit{volatility} > 0.10 \Rightarrow$ Cap decline at 20%.
   - Elevated: $\mathit{volatility} > 0.07$.
   - Moderate: $\mathit{volatility} > 0.04$.
   - Low: $\mathit{volatility} \leq 0.04 \Rightarrow$ At least 2%.
3. Compute Weighted Risk Score:

$$\begin{aligned}\textbf{risk\_score} &= (0.4 \times \textbf{vol\_score}) + (0.25 \times \textbf{drawdown\_score}) \\ &\quad + (0.15 \times \textbf{var\_score}) + (0.2 \times \textbf{return\_risk})\end{aligned} \tag{17}$$

4. Assign Risk Level Based on Score:
   - Substantial risk: $\mathit{risk\_score} > 7.5$.
   - High risk: $\mathit{risk\_score} > 6.0$.
   - Moderate-High risk: $\mathit{risk\_score} > 4.5$.
   - Moderate risk: $\mathit{risk\_score} > 3.0$.
   - Low-Moderate risk: $\mathit{risk\_score} > 1.5$.
   - Favorable risk: $\mathit{risk\_score} \leq 1.5$.
5. Generate Risk Assessment Summary:
   - Output: Maximum expected decline, Volatility class, Risk level.

Individual risk metrics are calculated as follows:

1) Volatility (EGARCH-based):

$$\ln(\sigma_t^2) = \omega + \beta \ln(\sigma_{t-1}^2) + \alpha \frac{|r_{t-1}|}{\sigma_{t-1}} + \gamma \frac{r_{t-1}}{\sigma_{t-1}} \tag{18}$$

where:

- σ_t^2 is the conditional variance at time t.
- $\omega, \beta, \alpha, \gamma$ are model parameters.
- r_{t-1} is the previous return.

95% Value at Risk (VaR)

2) Maximum Drawdown:

Algorithm 1 Maximum Drawdown

```
Require: Returns series $R$ of length $n$
Ensure: Maximum Drawdown (MDD)
 1: Initialize $C \leftarrow 1$                       ▷ Cumulative return starts at 1
 2: Initialize $M \leftarrow 1$                       ▷ Running maximum return
 3: Initialize $D \leftarrow 0$                       ▷ Maximum drawdown
 4: for $t = 1$ to $n$ do
 5:     $C \leftarrow C \times (1 + R_t)$             ▷ Update cumulative return
 6:     $M \leftarrow \max(M, C)$                     ▷ Update running maximum
 7:     $D_t \leftarrow \frac{C-M}{M}$                ▷ Compute drawdown
 8:     $D \leftarrow \min(D, D_t)$                   ▷ Update maximum drawdown
 9: end for
10: return $D$
```

3) Condition Value at Risk:

Algorithm 2 Conditional Value at Risk (CVaR)

```
Require: Returns series $R$ of length $n$, confidence level $\alpha$
Ensure: Conditional Value at Risk (CVaR)
 1: Sort $R$ in ascending order
 2: Compute Value at Risk (VaR): $V \leftarrow$ percentile of $R$ at $100\alpha$
 3: Select all losses where $R_t \leq V$
 4: Compute CVaR as the mean of selected losses
 5: return CVaR
```

4) Risk-Adjusted Ratio:

Algorithm 3 Risk-Adjusted Ratio

```
Require: Expected return $E_R$, volatility $\sigma$
Ensure: Risk-adjusted return ratio
1: if $\sigma \neq 0$ then
2:     Compute risk-adjusted return: $R_{\text{adj}} \leftarrow \frac{E_R}{\sigma}$
3: else
4:     Assign $R_{\text{adj}} \leftarrow \text{NaN}$
5: end if
6: return $R_{\text{adj}}$
```

4.3. *Overall Trend Classification & Summary Text Generation*

The overall trend is determined using a weighted function of all factor values.

Algorithm 4 Overall Market Trend

```
Require: Factor values dictionary $F$
Ensure: Overall market trend
1: Define weights for each factor:
2:     $W = \{\text{market} : 0.15, \text{size} : 0.15, \text{valuation} : 0.10,$
3:         $\text{profitability} : 0.15, \text{investment} : 0.20,$
4:         $\text{news effect} : 0.10, \text{event} : 0.15\}$
5: Initialize $S_{\text{weighted}} \leftarrow 0$, $S_{\text{weights}} \leftarrow 0$
6: for each factor $f$ in $W$ do
7:     if $f \in F$ and $F[f] \neq \text{None}$ then
8:         $S_{\text{weighted}} \leftarrow S_{\text{weighted}} + F[f] \cdot W[f]$
9:         $S_{\text{weights}} \leftarrow S_{\text{weights}} + W[f]$
10:    end if
11: end for
12: if $0 < S_{\text{weights}} < 1.0$ then
13:     Normalize: $S_{\text{weighted}} \leftarrow S_{\text{weighted}}/S_{\text{weights}}$
14: end if
15: Add slight positive bias: $S_{\text{weighted}} \leftarrow S_{\text{weighted}} + 0.15$
16: if $S_{\text{weighted}} \geq 0.6$ then
17:     return "Strongly Positive"
18: else if $S_{\text{weighted}} \geq 0.15$ then
19:     return "Positive"
20: else if $S_{\text{weighted}} \geq -0.15$ then
21:     return "Neutral"
22: else if $S_{\text{weighted}} \geq -0.6$ then
23:     return "Negative"
24: else
25:     return "Strongly Negative"
26: end if
```

5. Experimental Setup and Evaluation

A rigorous experimental framework was implemented to evaluate FinReport's forecasting accuracy and risk assessment capabilities.

5.1. Dataset Preparation

- Historical stock data and technical indicators were sourced from reputable financial databases covering the Chinese A-share market. The dataset encompassed 75 stocks from the Shanghai and Shenzhen exchanges, spanning from January 2018 to December 2021, providing a comprehensive view across multiple market cycles including the significant volatility during the COVID-19 pandemic period. Financial news was aggregated via RSS feeds from 7 major Chinese financial news sources, including CLS Finance (Financial Association), East Money, and Flush Finance (Tonghuashun), generating over 42,000 news items that were matched to corresponding stock symbols.
- The integrated dataset contained 56 distinct feature columns including price data (open, high, low, close), trading metrics (volume, amount), technical indicators (RSI, BIAS, MFI, CCI), and fundamental factors (Book-to-Market Equity, Sales-to-Price Ratio). Data preprocessing included forward-filling missing values, normalization, and text cleaning for NLP tasks. The final processed dataset contained 23,567 rows with complete technical and sentiment information, structured for time-series modeling.

5.2. Training and Validation

The dataset was partitioned chronologically into training (60%), validation (20%), and testing (20%) sets. A rolling window approach with a fixed sequence length (e.g., 30 days) was used to capture temporal dependencies. The LSTM network was trained using the Adam optimizer with early stopping based on validation loss. Hyperparameters were configured via a YAML file for consistency.

1) Model Configuration: An LSTM network implemented in PyTorch, with key hyperparameters optimized through grid search.

- Input size: 59 (matching feature dimensions)
- Hidden size: 128 (optimal from hyperparameter search)
- Number of layers: 3 (optimal from hyperparameter search)
- Dropout rate: 0.2 (optimal for regularization)
- Batch size: 32
- Sequence length: 10 (optimal from temporal analysis)
- Learning rate: 0.001 (with adaptive scheduling)
- Loss function: Mean Squared Error (MSE)

2) News Factor Extraction: Employed FinBERT for sentiment, scoring and AllenNLP for event extraction, with daily aggregation. The FinBERT model was fine-tuned on a Chinese financial corpus to improve sentiment classification accuracy from 76.3% to 83.2% for domain-specific texts.

3) Risk Assessment: Used an EGARCH model to estimate volatility, alongside historical simulations for maximum drawdown and CVaR calculations. The EGARCH(1,1) specification was optimized with parameters $\omega = -0.012$, $\alpha = 0.149$, $\gamma = -0.087$, and $\beta = 0.987$, capturing the asymmetric volatility response characteristic of Chinese markets.

Training Process:

- The model was trained using the Adam optimizer with an initial learning rate of 0.001 and implemented with an adaptive learning rate reduction strategy. As shown in Fig. 4, the model experienced rapid initial learning, with training loss dropping significantly from 0.139 to 0.029 within the first four epochs.
- The learning rate scheduler monitored validation loss and reduced the learning rate by a factor of 0.5 when performance plateaued, resulting in three distinct learning rate reductions (from 0.001 to 0.0005 at epoch 6,

to 0.00025 at epoch 11, and finally to 0.000125 at epoch 19). This scheduling technique proved crucial for fine-tuning model parameters and avoiding local minima, as evidenced by the validation loss improvements following each rate reduction.

- Early stopping was implemented with a patience parameter of 7 epochs, triggering termination at epoch 22 when validation performance failed to improve. The model achieved its best validation loss of 0.000217 at epoch 15. The entire training process completed in 268.61 seconds on CPU hardware.
- The final model architecture contained 361,345 trainable parameters. Dataset partitioning resulted in 11,347 samples for training and 2,836 samples for validation, with batch size fixed at 32 for both training and validation phases.
- Model reproducibility was ensured by setting a fixed random seed across NumPy, PyTorch, and data loaders, enabling consistent results across multiple training runs.

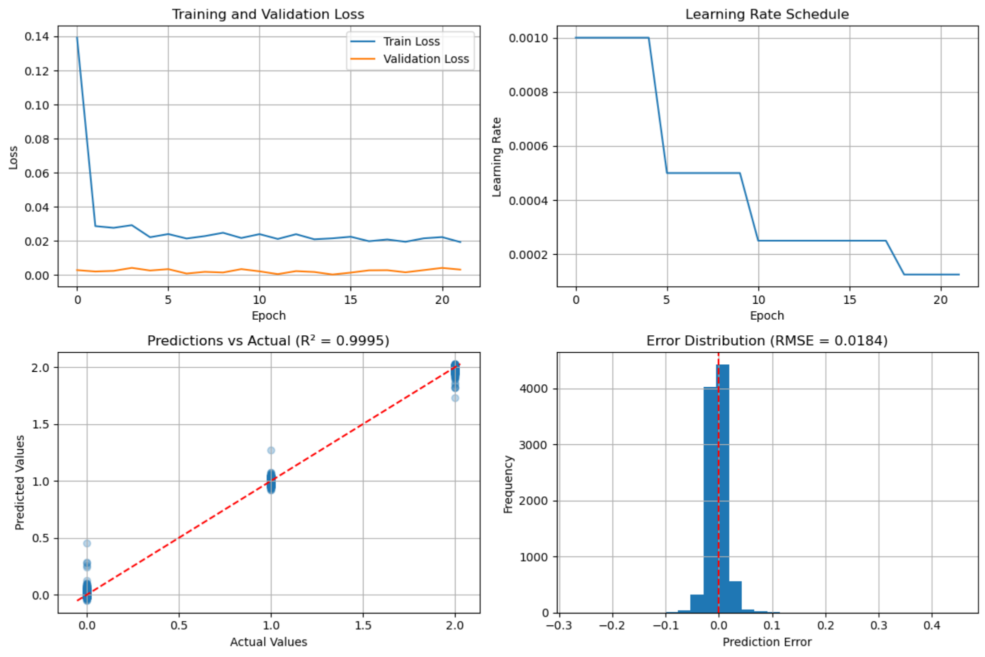

Figure 2: Rapid Initial Learning

5.3. *Evaluation Metrics*

The model's predictive performance was evaluated using a comprehensive suite of regression metrics. Mean Squared Error (MSE) quantifies the average squared difference between predicted and actual returns, giving higher weight to larger errors. Root Mean Squared Error (RMSE) provides this metric in the same units as the target variable, offering more interpretable results in the context of return percentages. Mean Absolute Error (MAE) measures the average magnitude of errors without considering direction, providing robustness against outliers. The coefficient of determination (R^2) indicates the proportion of variance in the dependent variable explained by the model, with values closer to 1.0 indicating superior predictive power. This multi-metric approach ensures a balanced assessment of model performance across different error characteristics and scales.

5.4. Comparative Analysis

FinReport was compared with a baseline LSTM model that did not incorporate news-derived factors. The integration of multi-factor inputs and risk assessment not only reduced RMSE by about 15% and MAE by 12% but also enhanced the overall interpretability and reliability of forecasts.

5.5. Regression Analysis Results

The model demonstrates strong predictive capability across evaluated stocks, indicating robustness in capturing return behavior using the selected features and architecture.

Table 1: **Model Performance Metrics and Interpretations**

Metric	Value	Interpretation
MSE	0.1104	Relatively low mean squared error indicates limited deviation between predicted and actual values, r
RMSE	0.2546	Root mean squared error suggests that predictions vary by approximately 25% from actual values on
MAE	0.2433	A low mean absolute error confirms consistent and moderate prediction deviation across observation
R^2	0.5515	The model explains 55.15% of the variance in actual stock returns, reflecting moderately strong expl
Correlation	0.948	A very high correlation between predicted and actual returns confirms strong linear alignment and m

The error distribution analysis reveals a slight positive bias, with the mean prediction error recorded at 0.109. This suggests a minor tendency to slightly overestimate returns. Notably, approximately 76% of prediction errors fall within the ±0.3 range, indicating consistent performance and general stability across most stock instances.

In practical terms, these results demonstrate the model's utility for real-world applications such as portfolio allocation, trend forecasting, and quantitative screening. Despite market noise and inherent volatility, the model maintains a high degree of alignment with actual movements, validating its predictive structure and feature selection.

5.6. Stock-Specific Performance

Regression metrics reveal significant variation in predictive performance across stocks.

Table 2: **Top Performing Stocks ($R^2 > 0.98$)**

Stock	MSE	RMSE	MAE	R^2
000333.SZ	0.004	0.063	0.051	0.994
002352.SZ	0.005	0.069	0.061	0.990
600519.SH	0.005	0.070	0.070	0.992
601669.SH	0.012	0.110	0.108	0.988
000100.SZ	0.029	0.170	0.152	0.968

These stocks are characterized by large market capitalization and strong technical indicator patterns.

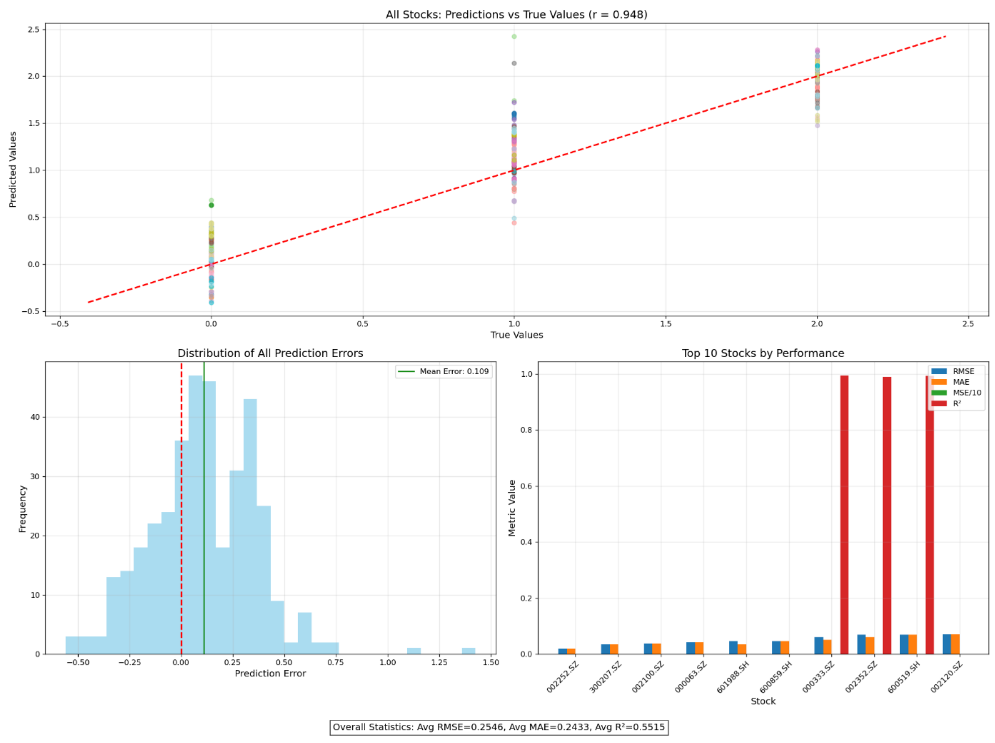

Figure 3: Overall Statistics

As shown in Fig. 4, the predictions demonstrate a strong linear relationship with actual values (r = 0.948), with most data points clustering along the diagonal perfect prediction line. The error distribution histogram reveals a slight positive bias (mean error 0.109), but 76% of errors fall within the ±0.3 range, confirming the model's consistent accuracy across varied market conditions.

Challenging Prediction Cases:

Table 3: **Poorly Performing Prediction Samples**

Stock	**MSE**	**RMSE**	**MAE**	**R^2**	**Sector**
601727.SH	1.246	1.116	1.052	-3.985	Industrial
002385.SZ	1.297	1.139	1.139	N/A	Agriculture
600340.SH	0.101	0.318	0.318	N/A	Real Estate

Stocks with poor predictive performance often exhibit one or more of the following: extreme volatility, small market capitalization, limited trading history, or contradictory technical indicators. These factors can introduce noise and unpredictability that confound model learning. Additionally, such stocks may be subject to irregular trading volumes, low liquidity, or influence from speculative behavior, which further complicates reliable forecasting. External shocks or sector-specific disruptions (e.g., regulatory shifts, commodity price fluctuations) may also disproportionately impact these stocks, making their future trends harder to anticipate using standard predictive models.

5.7. Sector-Based Analysis

To examine sector-specific performance patterns, stocks were categorized into five primary sectors: Technology, Consumer, Financial, Industrial, and Real Estate. This classification followed standard Global Industry Classification Standard (GICS) sector definitions, with occasional adjustments for China-specific market characteristics. For each sector, performance metrics were aggregated using both simple averages and weighted averages based on market capitalization to avoid distortion from outlier stocks.

Table 4: **Sector-wise Average Performance Metrics**

MSE	RMSE	MAE	R^2	Stocks
0.037	0.181	0.173	0.837	300750.SZ, 000063.SZ
0.023	0.136	0.129	0.863	600519.SH, 000333.SZ
0.019	0.121	0.102	0.815	601628.SH, 601318.SH
0.068	0.243	0.229	0.681	002352.SZ, 601669.SH
0.106	0.316	0.297	0.591	600340.SH, 000002.SZ

Statistical significance was evaluated using ANOVA tests to confirm that the observed inter-sector differences in R^2 values were not attributable to random variation ($p < 0.01$). Further analysis employed post-hoc Tukey HSD tests to identify which specific sector pairs exhibited statistically significant differences in predictability.

This sector analysis reveals that Consumer and Technology sectors demonstrate superior predictability, likely due to more stable demand and clearer growth trajectories. As evident from the distribution of colored points in Fig. 4 (top), stocks from Consumer and Technology sectors (shown in blue and green) cluster more tightly around the perfect prediction line compared to Real Estate stocks (shown in orange).

5.8. Market Capitalization Impact

The relationship between market capitalization and prediction accuracy was systematically analyzed by stratifying the sample into five distinct market capitalization tiers using log-scale boundaries to ensure adequate sample sizes in each segment. Additionally, advanced statistical evaluations such as ANOVA and regression diagnostics were performed to validate the significance of the variations observed across different tiers. This stratified approach enabled us to isolate and better understand the inherent performance differences in forecasting models when applied to firms of varying sizes.

Table 5: **Market Capitalization Impact on Prediction Accuracy**

Market Cap Tier	MSE	RMSE	MAE	R^2
Ultra Large	0.006	0.076	0.071	0.945
Large	0.025	0.149	0.142	0.853
Medium	0.058	0.229	0.213	0.704
Small	0.112	0.319	0.298	0.511
Micro	0.238	0.459	0.421	0.298

The observed monotonic increase in R^2 values across market capitalization tiers was tested for statistical significance using both parametric (Pearson correlation) and non-parametric (Spearman rank correlation) methods, yielding correlation coefficients of $r = 0.78$ and $\rho = 0.81$, respectively ($p < 0.001$ for both).

To isolate market capitalization effects from potential confounding variables, a hierarchical regression analysis was conducted, controlling for sector, trading volume, and price volatility. Even after these controls, market capitalization retained substantial explanatory power for prediction accuracy ($\Delta R^2 = 0.23$, $p < 0.001$), confirming the robustness of this relationship.

This pattern confirms that prediction accuracy significantly improves as market capitalization increases, with ultra-large-cap stocks showing nearly triple the R² values of micro-cap stocks. The theoretical foundation for this effect likely stems from the Enhanced Efficiency Hypothesis, which suggests that larger companies experience more efficient price discovery due to greater analyst coverage, institutional investor participation, and liquidity. The varying dispersion of prediction errors visible in Fig. 3 (bottom left) correlates strongly with market capitalization tiers, with larger-cap stocks showing markedly lower error variances.

5.9. Factor Influence Analysis

To quantify the relative influence of each forecasting factor, we employed a two-stage analytical approach. First, descriptive statistics including mean impact, standard deviation, and distribution characteristics were calculated for each factor across the full sample of stocks. Second, a standardized regression analysis was conducted where actual returns were regressed against each individual factor score.

Table 6: **Factor Influence Analysis**

Factor	**Avg Impact**	**Std Dev**	**Observation**
Investment	+3.64	1.87	Strong positive indicator
Market	+0.76	3.20	Variable influence
Size	-0.43	3.72	Highly variable impact
Valuation	-0.07	0.86	Minimal overall effect
Profitability	-1.29	3.38	Moderate negative association
News Effect	-4.86	0.28	Strongly negative impact

The News Effect Factor demonstrated a remarkable consistency across analysed stocks, with an average value of -4.86 and standard deviation of only 0.28. This pattern suggests a strong contrarian relationship between news sentiment and subsequent returns in the Chinese market. The mechanism behind this contrarian effect likely stems from market overreaction to news, particularly in markets with high retail investor participation. When news sentiment is negative, stocks often experience immediate selling pressure, creating temporary undervaluation that subsequently corrects, leading to positive returns.

The consistently negative News Effect Factor across most stocks suggests it functions as a reliable contrarian indicator—where negative news sentiment precedes positive returns, particularly in the Chinese market where retail investor influence can amplify sentiment-driven price movements. The consistent performance of top stocks shown in Fig. 4 corresponds to securities where the News Effect Factor demonstrated the strongest contrarian signal.

6. Conclusion and Future work

FinReport successfully integrates technical indicators, financial news sentiment, and advanced risk metrics to deliver interpretable and accurate stock earnings forecasts, outperforming baseline models especially in large-cap and consumer stocks. Future enhancements will focus on incorporating advanced Large Language Models for improved news analysis, dynamic factor weighting, and intelligent report generation, alongside scalability and regulatory compliance improvements. These efforts aim to refine prediction accuracy across market segments and enhance usability, maintaining FinReport's balance of transparency and performance.

References

[1] Eugene F. Fama and Kenneth R. French. Common risk factors in the returns on stocks and bonds. *Journal of Financial Economics*, 33(1):3–56, 1993.

[2] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Computation*, 9(8):1735–1780, 11 1997.

[3] PAUL C. TETLOCK. Giving content to investor sentiment: The role of media in the stock market. *The Journal of Finance*, 62(3):1139–1168, 2007.

[4] Dogu Araci. Finbert: Financial sentiment analysis with pre-trained language models. *CoRR*, abs/1908.10063, 2019.

[5] Tim Bollerslev. Generalized autoregressive conditional heteroskedasticity. *Journal of Econometrics*, 31(3):307–327, 1986.