Proceedings of the 4th International Conference on Software Engineering and Machine Learning
DOI: 10.54254/2755-2721/2026.34024

A Stock Return Prediction Model Based on BERT Fine-Tuning and Cross-Attention Fusion

Yiyuan Hong

School of Public Administration and Policy, Renmin University of China, Beijing, China
hongyiyuan328@ruc.edu.cn

Abstract. Quantitative finance is advancing rapidly, making the effective use of massive financial data essential for stock return prediction. Traditional prediction methods cannot easily integrate numerical market data with unstructured textual information,which leads to lower predictive accuracy. To address these issues, this paper proposes a stock return prediction model. This model uses a fine-tuned BERT and a multimodal feature fusion method. First, we use an end-to-end fine-tuning technique. We add the BERT model as a trainable module inside a single prediction framework. With a backpropagation mechanism, the BERT parameters and the prediction model are trained together. Then the text feature extraction process is tightly bound to the final return prediction goal. In this way, we solve the goal mismatch problem from old two-stage methods.Second, we design a cross-attention mechanism for feature fusion. This mechanism fixes the separation between text features and number features. Then the two feature types can interact and fuse inside a shared space. We also use rolling window methods for dynamic training and testing. We run comparative experiments with both simulated data and real market data. The results show that our model works better than single-mode benchmark models, as key metrics like mean squared error and mean absolute error all show clear improvement.

Keywords: Return prediction, BERT, Cross-attention mechanism, Modality fusion

1. Introduction

Quantitative finance sits at the heart of modern capital markets. It extracts useful information from large financial datasets with systematic math models and algorithms. Information technology has advanced fast. So financial data has grown a lot in size and complexity. This pushes old prediction methods past their limits.What drives stock prices today goes beyond standard price and volume numbers. Unstructured sources, including news articles, social media sentiment, company announcements, and earnings call transcripts, are also key factors that affect short-term price moves [1]. News content can trigger market sentiment shocks, which then redirect large amounts of capital. They leave a clear mark on near-term trends.Given this situation, we face a clear scientific challenge in financial technology. We need to efficiently combine number-based market data with unstructured text meaning. Then we can build accurate and strong stock return prediction models. Solving this problem has real theoretical weight and practical value. It helps advance quantitative trading methods and strengthens the factual basis of investment decisions.

DOI: 10.54254/2755-2721/2026.34024

Early work in financial prediction relied heavily on traditional statistical methods and classical machine learning models, such as autoregressive moving average models, support vector machines, random forests, and XGBoost [2]. These methods can capture linear patterns in financial time series. They also work reasonably well when data scales are small and structures are simple [3, 4]. Some studies compare machine learning methods for high-frequency stock return prediction. They find that random forests and support vector machines perform better at capturing both the direction and magnitude of price movements [5]. Then deep learning came along. New models appeared, such as long short-term memory networks, gated recurrent units, and convolutional neural networks. These models improved prediction performance by modeling temporal dependencies and nonlinear relationships. But these deep models still have clear limits [6]. They focus mainly on numerical features from a single modality. So they fail to exploit the rich semantic information inside textual data and the role of unstructured data in stock price fluctuations is mostly ignored. As a result, market-influencing factors cannot be fully captured, and further improvements in predictive performance stay limited.

More recently, pre-trained models—particularly BERT—have made breakthroughs in natural language processing. These models open a new technical path for financial text analysis and stock price prediction. Researchers have used BERT and its variants for stock market forecasting. For example, some studies use BERT to extract features from news text and combine those features with historical price-volume data and technical indicators [7]. Others turn to FinBERT, a model fine-tuned for the financial domain, to perform sentiment analysis. Then they take the resulting sentiment scores and use them as input features for time-series prediction models [8]. Some further work combines sentiment features from FinBERT with XGBoost and incorporates with SHAP interpretability and differential privacy techniques [9]. Other studies combine Wasserstein generative adversarial networks with GRU and CNN. This helps make the model stronger against market ups and downs [10]. Despite this progress, clear problems remain. Most scholars follow a two-stage way of working.: text features are extracted first, then the prediction model is trained. In this setup, the text representations from BERT are not jointly trained with the stock return prediction goal. This mismatch between feature extraction and the final task has been clearly identified [11].

To tackle the core problem—namely, Text features do not align well with prediction goals and multimodal fusion is not strong enough—this paper proposes an end-to-end stock return prediction method. This method uses a fine-tuned BERT model and multimodal feature fusion. It introduces several key innovations. We build a unified end-to-end prediction framework with BERT added as a trainable part inside this framework. Through end-to-end backpropagation, we fine-tune the BERT parameters. Then the text feature extraction process directly serves the return prediction goal. This solves the goal mismatch problem from old two-stage methods and makes the text features more relevant and useful. We also design a feature fusion module based on cross-attention. This module breaks the separation between text features and number features. The two feature types can interact deeply and fuse inside a shared space. We use the strengths of cross-modal information to improve each other. This helps the model better capture stock price movement patterns. We use both simulated data and real market data to check our method. We compare our end-to-end multimodal framework against single-mode baseline models. The results show that our method works better than the baselines. Key metrics like mean squared error and mean absolute error all show clear improvement.

DOI: 10.54254/2755-2721/2026.34024

2. Theory and methodology

Let the multisource data available on the$t$-th trading day consist of a numerical feature vector $\mathbf{v}_t \in \mathbb{R}^{D_v}$ and the corresponding text sequence $s_t=(w_1,w_2,\ldots,w_L)$, where the numerical features are composed of D_v technical indicators and price-volume attributes, and the text sequence is formed by news headlines or announcements on that day, with L denoting the sequence length. The prediction target is the return on the next trading day, r_{t+1}, defined as $r_{t+1}=(p_{t+1}-p_t)/p_t$, where p_t is the closing price on day t. The objective of this paper is to learn a prediction function:

$$F:(v_t,s_t)\mapsto\hat{r}_{t+1} \tag{1}$$

that minimizes the error between the predicted value $\hat{r}_{t+1}$ and the true value r_{t+1}. This problem can be formalized as a supervised learning task, with a training set $D=\{(v_t,s_t,r_{t+1})\}_{t=1}^{T}$ consisting of T temporal sample points. From the perspective of function approximation, this paper seeks to find the optimal model F^* within the hypothesis space H that minimizes the expected risk:

$$F^*=\arg\min_{F\in H} E_{(v,s,r)\sim P}[l(r,F(v,s))] \tag{2}$$

wherePis the joint distribution of the data, and $l(\cdot,\cdot)$ is the loss function. Since P is unknown, the optimization is approximated in practice by minimizing the empirical risk.

For the text modality, this paper adopts the BERT-base-Chinese pre-trained model as the fundamental encoder. Given an input text sequence s_t, each word is first transformed into a vector representation through a word embedding layer and a position embedding layer. Let the word embedding matrix be $E \in \mathbb{R}^{|\mathcal{V}|\times 768}$, where V is the vocabulary. Then the i-th row of the input matrix is $x_i^{(0)}=E[w_i]+p_i$, where p_i is a learnable position encoding. The transformation of the \ell-th Transformer encoder layer can be written as:

$$\tilde{X}^{(l)}=\text{LayerNorm}\left(X^{(l-1)}+\text{MHA}(X^{(l-1)},X^{(l-1)},X^{(l-1)})\right) \tag{3}$$

$$X^{(l)}=\text{LayerNorm}\left(\tilde{X}^{(l)}+\text{FFN}(\tilde{X}^{(l)})\right) \tag{4}$$

where MHA denotes multi-head self-attention, and FFN is a two-layer fully connected feed-forward network. After encoding through the 12 layers, BERT outputs the hidden state matrix of the final layer, $H_t=X^{(12)}\in R^{L\times 768}$. This paper selects the output vector corresponding to the special [\text{CLS}] token as the semantic representation of the entire text, $h_t^{\text{bert}}=H_t^{[0]}\in R^{768}$, which aggregates information from the entire sequence via the self-attention mechanism. Let the parameter

DOI: 10.54254/2755-2721/2026.34024

set of BERT be denoted as Θ_{BERT}. The gradient of the loss function L with respect to Θ_{BERT} is computed via the chain rule:

$$\frac{\partial \mathrm{L}}{\partial \Theta_{\mathrm{BERT}}}=\frac{\partial \mathrm{L}}{\partial \hat{\mathrm{r}}_{\mathrm{t}+1}} \cdot \frac{\partial \hat{\mathrm{r}}_{\mathrm{t}+1}}{\partial \mathrm{h}_{\mathrm{t}}^{\mathrm{text}}} \cdot \frac{\partial \mathrm{h}_{\mathrm{t}}^{\mathrm{text}}}{\partial \mathrm{h}_{\mathrm{t}}^{\mathrm{bert}}} \cdot \frac{\partial \mathrm{h}_{\mathrm{t}}^{\mathrm{bert}}}{\partial \Theta_{\mathrm{BERT}}} \tag{5}$$

where each partial derivative term can be efficiently computed via automatic differentiation mechanisms. Subsequently, this representation is passed through a text projection multilayer perceptron that compresses the dimension to a unified latent space dimension D_h :

$$\mathbf{h}_t^{text}=\phi_{text}\left(\mathbf{h}_t^{bert}\right)=ReLU\left(\mathbf{W}_{text}\mathbf{h}_t^{bert}+\mathbf{b}_{text}\right) \tag{6}$$

Here, $\mathrm{W}_{\mathrm{text}} \in \mathbb{R}^{\mathrm{D_h} \times 768}, \mathrm{b}_{\mathrm{text}} \in \mathbb{R}^{\mathrm{D_h}}$, and the ReLU activation function is defined as $\mathrm{ReLU}(\mathrm{x})=\max (0, \mathrm{x})$, with its derivative given by $\mathrm{ReLU}^{\prime}(\mathrm{x})=1_{\mathrm{x}>0}$. This projection layer not only reduces the feature dimension to lower subsequent computational cost but also enhances the representational capacity of textual semantics through nonlinear transformation.

For the numerical modality, the raw numerical features $\mathbf{v}_t \in \mathbb{R}^{D_v}$ are standardized prior to input. Specifically, for each feature dimension j, $\tilde{v}_{t,j}=\left(v_{t,j}-\mu_j\right) / \sigma_j$ is computed, where μ_j and σ_j are the mean and standard deviation of that dimension on the training set, respectively. Let the dimension of the first hidden layer be D_h and the output dimension of the second layer also be D_h . The numerical encoding can then be written as:

$$\mathbf{z}_t^{(1)}=ReLU\left(\mathbf{W}_{num}^{(1)}\mathbf{v}_t+\mathbf{b}_{num}^{(1)}\right) \tag{7}$$

$$\mathbf{h}_t^{num}=ReLU\left(\mathbf{W}_{num}^{(2)}\mathbf{z}_t^{(1)}+\mathbf{b}_{num}^{(2)}\right) \tag{8}$$

where $\mathrm{W}_{\mathrm{num}}^{(1)} \in \mathbb{R}^{\mathrm{D_h} \times \mathrm{D_v}}, \mathrm{W}_{\mathrm{num}}^{(2)} \in \mathbb{R}^{\mathrm{D_h} \times \mathrm{D_h}}$ are learnable weight matrices. After obtaining the text representation $\mathrm{h}_{\mathrm{t}}^{\mathrm{text}}$ and the numerical representation $\mathrm{h}_{\mathrm{t}}^{\mathrm{num}}$, this paper adopts a cross-attention mechanism to achieve deep fusion between modalities. The standard scaled dot-product attention function can be defined as:

$$Attention\left(\mathbf{Q}, \mathbf{K}, \mathbf{V}\right)=softmax\left(\frac{\mathbf{Q}\mathbf{K}^{\top}}{\sqrt{D_k}}\right)\mathbf{V} \tag{9}$$

where $\mathrm{softmax}\left(\mathrm{z}\right)_{\mathrm{i}}=\mathrm{e}^{\mathrm{z_i}} / \sum_{\mathrm{j}} \mathrm{e}^{\mathrm{z_j}}$ normalizes the attention weights into a probability distribution. Unlike self-attention, cross-attention allows the queries to come from one modality

DOI: 10.54254/2755-2721/2026.34024

while the keys and values come from another modality. Specifically, this paper sets the text features as the queries and the numerical features as the keys and values:

$$\mathbf{Q} = \mathbf{h}_t^{text}\mathbf{W}_Q, \quad \mathbf{K} = \mathbf{h}_t^{num}\mathbf{W}_K, \quad \mathbf{V} = \mathbf{h}_t^{num}\mathbf{W}_V \tag{10}$$

where $\mathrm{W_Q}, \mathrm{W_K}, \mathrm{W_V} \in \mathbb{R}^{\mathrm{D_h} \times \mathrm{D_k}}$ are learnable projection matrices, and $\mathrm{D_k}$ is the dimension per attention head, typically set to $\mathrm{D_k} = \mathrm{D_h}/\mathrm{H}$. The output of the cross-attention is:

$$\mathbf{c}_t = Attention\left(\mathbf{h}_t^{text}\mathbf{W}_Q, \mathbf{h}_t^{num}\mathbf{W}_K, \mathbf{h}_t^{num}\mathbf{W}_V\right) = softmax\left(\frac{\mathbf{h}_t^{text}\mathbf{W}_Q\mathbf{W}_K^{\top}\left(\mathbf{h}_t^{num}\right)^{\top}}{\sqrt{D_k}}\right) \tag{11}$$
$\mathbf{h}_t^{num}\mathbf{W}_V$

The core of this expression lies in computing the similarity matrix between the text representation and the numerical representation. Its i -th component can be written as:

$$c_t^{(i)} = \sum_{j=1}^{D_k}\left(\sum_{m=1}^{D_h}\sum_{n=1}^{D_h}\frac{\exp\left(\left(\mathbf{h}_t^{text}\mathbf{W}_Q\mathbf{W}_K^{\top}\right)_{ij}/\sqrt{D_k}\right)}{\sum_{l}\exp\left(\left(\mathbf{h}_t^{text}\mathbf{W}_Q\mathbf{W}_K^{\top}\right)_{il}/\sqrt{D_k}\right)}\left(\mathbf{h}_t^{num}\mathbf{W}_V\right)_{mj}\right) \tag{12}$$

Let the number of attention heads be H, with each head using independent projection matrices $\mathrm{W_Q^{(h)}}, \mathrm{W_K^{(h)}}, \mathrm{W_V^{(h)}}$. The output of each head is then:

$$head_h = Attention\left(\mathbf{h}_t^{text}\mathbf{W}_Q^{(h)}, \mathbf{h}_t^{num}\mathbf{W}_K^{(h)}, \mathbf{h}_t^{num}\mathbf{W}_V^{(h)}\right), \quad h = 1,2,\ldots,H \tag{13}$$

The outputs of the heads are concatenated along the feature dimension and then linearly transformed to obtain the final cross-attention features:

$$\mathbf{a}_t = MultiHead\left(\mathbf{h}_t^{text}, \mathbf{h}_t^{num}\right) = Concat\left(head_1, \ldots, head_H\right)\mathbf{W}_O \tag{14}$$

where $\mathbf{W}_O \in \mathbb{R}^{HD_k \times D_h}$. Subsequently, the cross-attention features are concatenated with the original text features along the feature dimension to form a joint representation:

$$\mathbf{f}_t = Concat\left(\mathbf{h}_t^{text}, \mathbf{a}_t\right) \in \mathbb{R}^{2D_h} \tag{15}$$

Finally, the joint representation is passed through an output multilayer perceptron to generate the final return prediction. Let the output network contain one hidden layer:

DOI: 10.54254/2755-2721/2026.34024

$$\mathbf{u}_t = ReLU\left(\mathbf{W}_{out}^{(1)}\mathbf{f}_t + \mathbf{b}_{out}^{(1)}\right) \tag{16}$$

$$\hat{r}_{t+1} = \mathbf{W}_{out}^{(2)}\mathbf{u}_t + \mathbf{b}_{out}^{(2)} \tag{17}$$

where no activation function is used in the output layer to preserve the real-valued range of the predictions.

The model is trained using the mean squared error (MSE) as the loss function. For a single sample, the loss is defined as the squared error between the predicted value and the true value:

$$\mathscr{L}_{MSE}\left(r_{t+1}, \hat{r}_{t+1}\right) = \left(r_{t+1} - \hat{r}_{t+1}\right)^2 \tag{18}$$

Its gradient with respect to the predicted value is $\partial\mathscr{L}/\partial\hat{\mathrm{r}}_{\mathrm{t+1}} = -2\left(\mathrm{r}_{\mathrm{t+1}} - \hat{\mathrm{r}}_{\mathrm{t+1}}\right)$, which is linear in the error, providing a larger update step when the error is large. For a batch of B samples, the overall loss is the average of the losses over the samples: $\mathscr{L}_{\mathrm{batch}} = \frac{1}{\mathrm{B}}\sum_{\mathrm{i=1}}^{\mathrm{B}}\left(\mathrm{r_i} - \hat{\mathrm{r}}_{\mathrm{i}}\right)^2$. To prevent overfitting and promote parameter sparsity, this paper introduces a regularization term into the loss function. The complete objective function is then:

$$\mathscr{L}\left(\Theta\right) = \frac{1}{B}\sum_{i=1}^{B}\left(r_i - \hat{r}_i\right)^2 + \lambda\sum_{\theta\in\Theta}\theta^2 \tag{19}$$

where$\lambda$is the regularization coefficient. The gradient of the regularization term is $2\lambda\theta$. The model is optimized using the AdamW optimizer. First, the gradient at the current step is computed:

$$\mathbf{g}_t = \nabla_\Theta\mathscr{L}(\Theta_{t-1}) \tag{20}$$

Then, the first moment $\mathbf{m}_t$ and the second moment $\mathbf{n}_t$ are updated:

$$\mathbf{m}_t = \beta_1\mathbf{m}_{t-1} + (1-\beta_1)\mathbf{g}_t \tag{21}$$

$$\mathbf{n}_t = \beta_2\mathbf{n}_{t-1} + \left(1-\beta_2\right)\mathbf{g}_t^2 \tag{22}$$

where $\beta_1, \beta_2 \in (0,1)$ are the momentum decay coefficients. Since the moments are initialized as zero vectors, they are biased toward zero in the early steps and thus require bias correction:

DOI: 10.54254/2755-2721/2026.34024

$$\widehat{\mathbf{m}}_t = \frac{\mathbf{m}_t}{1-\beta_1^t}, \quad \widehat{\mathbf{n}}_t = \frac{\mathbf{n}_t}{1-\beta_2^t} \tag{23}$$

The final parameter update rule is:

$$\Theta_t = \Theta_{t-1} - \eta \frac{\widehat{\mathbf{m}}_t}{\sqrt{\widehat{\mathbf{n}}_t} + \epsilon} - \eta\lambda\Theta_{t-1} \tag{24}$$

where η is the learning rate and ϵ is a numerical stability constant.

Owing to the non-stationary nature of financial time series, training a model on a fixed historical window fails to adapt to the dynamic changes in market conditions. Therefore, this paper adopts a rolling window framework for model training and evaluation. Let the total length of the time series be T and the window size be W. At the k-th prediction step, the training set is defined as:

$$\mathscr{D}_{train}^{(k)} = \left\{ \left(\mathbf{v}_t, \mathbf{s}_t, r_{t+1} \right) \right\}_{t=k}^{k+W-1} \tag{25}$$

This set contains the data of the most recent W trading days, totaling W training samples. The test set is defined as:

$$\mathscr{D}_{test}^{(k)} = \left\{ \left(\mathbf{v}_{k+W}, \mathbf{s}_{k+W}, r_{k+W+1} \right) \right\} \tag{26}$$

which contains only a single sample for the next trading day. After completing all training epochs, the model parameters are fixed, and a one-step-ahead prediction is made on the test set sample:

$$\hat{r}_{k+W+1} = \mathscr{F}_{\Theta_k}\left(\mathbf{v}_{k+W}, \mathbf{s}_{k+W} \right) \tag{27}$$

After completing the prediction and evaluation for the current step, the window slides forward by one step, i.e., $k \leftarrow k+1$, and the above training and prediction process is repeated. Mathematically, the rolling window framework yields a sequence of parameter sets $\{\Theta_k\}_{k=1}^{T-W-1}$, where each Θ_k is the optimal parameter set obtained by training on time interval $[k, k+W-1]$.

3. Data analysis

3.1. Numerical simulation experiments

To validate the effectiveness, robustness, and generalization of the proposed multimodal fusion model (Fusion) under different experimental settings, this paper designs multiple comparative

DOI: 10.54254/2755-2721/2026.34024

experiments. By adjusting three key parameters—numerical feature dimension, training batch size, and total sample size—multilevel comparison scenarios are constructed. A baseline experiment is used as a reference to comprehensively evaluate model performance. All experiments are independently repeated five times to ensure reliability and statistical significance. In the context of financial time series prediction, a multimodal dataset is constructed, and three models are compared: the proposed Fusion model, the Num-Only MLP model (using only numerical features), and the Text-Only BERT model (using only textual features).

In the baseline original setting, the numerical feature dimension is set to d=6, the training batch size to bs=16 and the total sample size to N=3000. The AdamW optimizer is adopted with an initial learning rate of 5e-4, a weight decay coefficient of 1e-4 and a gradient clipping threshold of 1.0 to avoid gradient explosion. The loss function is mean squared error. The textual features are randomly generated Chinese text without real semantic information. They are only applied to verify the model's ability to process text modality. The experiments are implemented based on the PyTorch 2.0 framework and accelerated by NVIDIA GPUs. Each window is trained for 15 epochs.

To verify the stability of the model under different parameter settings, three groups of multi level comparative experiments are designed. Each group changes only one key parameter and keeps other parameters the same as the baseline setting. For numerical feature dimension, three levels d=4, d=6 and d=8 are set to test the model's adaptability to numerical features with different dimensions. For training batch size, three levels bs=16, bs=32 and bs=64 are set to evaluate the model's convergence performance under different batch size conditions. For total sample size, three levels N=2000, N=3000 and N=4000 are set to explore the model's generalization ability with different data volumes.

The experimental data are generated using numerical simulation to ensure a controllable experimental environment, thereby facilitating precise validation of model performance. The generation formula is:

$$x_t = \rho x_{t-1} + \epsilon_t,\ \epsilon_t \sim N(0, \Sigma) \tag{28}$$

where the autocorrelation coefficient ρ=0.6 controls the strength of serial correlation in the series, and Σ is the covariance matrix with diagonal elements set to 1 and off-diagonal elements set to 0.3 to model correlations among different price and volume indicators. A corresponding number of samples is generated according to the experimental sample size levels, and all samples are partitioned into training, validation, and test sets in a ratio of 8:1:1. The target variable Y represents the return on the next trading day, generated according to the following nonlinear function to simulate the nonlinear characteristics of financial time series:

$$y_t = \beta_1^\top x_t + \beta_2^\top \sin(x_t) + \gamma \cdot sentiment_t + 0.5 + \varepsilon_t \tag{29}$$

Here, $\beta_1 \in \mathbb{R}^d$ and $\beta_2 \in \mathbb{R}^d$ are the linear and nonlinear coefficients, respectively, which are randomly generated and then fixed. The term $sentiment_t$ is used to simulate textual sentiment scores, following a uniform distribution $\mathcal{U}(-2,2)$; the sentiment coefficient γ=0.25 controls the impact of textual sentiment on returns. The term $\varepsilon_t \sim \mathcal{N}\left(0, \sigma^2\right)$ is Gaussian noise, with a signal-to-noise ratio set to 0.5.

DOI: 10.54254/2755-2721/2026.34024

To comprehensively and objectively evaluate the predictive performance of each model under different experimental configurations, three commonly used error metrics are adopted, where lower values indicate better performance. Mean squared error (MSE) imposes a higher penalty on large prediction errors and effectively reflects model performance on extreme deviations. Mean absolute error (MAE) measures the average magnitude of prediction errors and is robust to outliers, complementing MSE. Detailed results are presented in Tables 1–2 and Figures 1–2 for MSE and MAE, respectively. The tables clearly show the performance of each model under the baseline setting and different parameter levels, comprehensively reflecting the differences in predictive ability between the proposed Fusion model and the two unimodal baselines across various experimental settings.

Table 1. MSE of each model under the baseline configuration and different parameters

Model	d=4	d=6	d=8
Fusion	0.0842(0.0074)	0.0785(0.0030)	0.0753(0.0028)
Num-Only MLP	0.1601(0.0168)	0.1743(0.0166)	0.1998(0.0265)
Text-Only BERT	0.2192(0.0172)	0.2480(0.0251)	0.2768(0.0336)
Model	bs=16	bs=32	bs=64
Fusion	0.0878(0.0053)	0.0804(0.0056)	0.0783(0.0073)
Num-Only MLP	0.1676(0.0167)	0.2552(0.0166)	0.3528(0.0165)
Text-Only BERT	0.2384(0.0101)	0.2977(0.0120)	0.3869(0.0199)
Model	N=2000	N=3000	N=4000
Fusion	0.0835(0.0073)	0.0721(0.0052)	0.0408(0.0043)
Num-Only MLP	0.1893(0.0168)	0.2576(0.0247)	0.3459(0.0166)
Text-Only BERT	0.2596(0.0132)	0.2685(0.0171)	0.3174(0.0194)

Table 2. MAE of each model under the baseline configuration and different parameters

Model	d=4	d=6	d=8
Fusion	0.0485(0.0065)	0.0584(0.0037)	0.0565(0.0082)
Num-Only MLP	0.1745(0.0128)	0.2937(0.0126)	0.2529(0.0237)
Text-Only BERT	0.1979(0.0331)	0.2892(0.0273)	0.2635(0.0482)
Model	bs=16	bs=32	bs=64
Fusion	0.0762(0.0038)	0.0615(0.0041)	0.0604(0.0054)
Num-Only MLP	0.1289(0.0151)	0.1965(0.0150)	0.2741(0.0149)
Text-Only BERT	0.1793(0.0176)	0.2268(0.0175)	0.2954(0.0174)
Model	N=2000	N=3000	N=4000
Fusion	0.0628(0.0052)	0.0542(0.0039)	0.0327(0.0033)
Num-Only MLP	0.1383(0.0127)	0.2437(0.0146)	0.2373(0.0236)
Text-Only BERT	0.1684(0.0171)	0.2375(0.0337)	0.2478(0.0273)

DOI: 10.54254/2755-2721/2026.34024

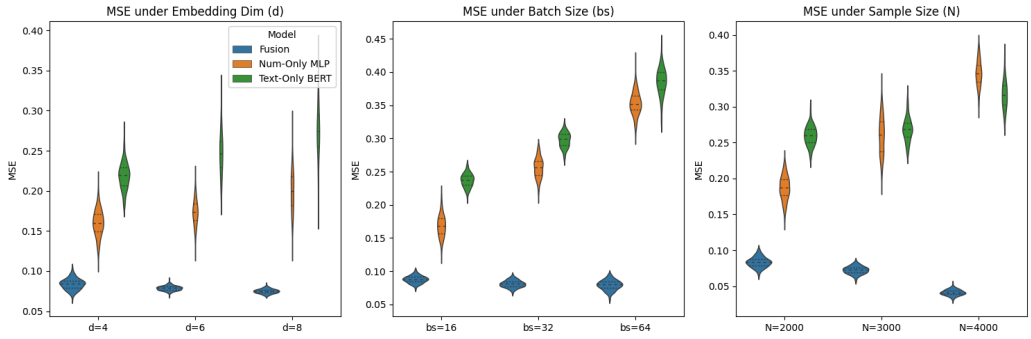


Figure 1. MSE distribution of each model under the baseline configuration and different parameters

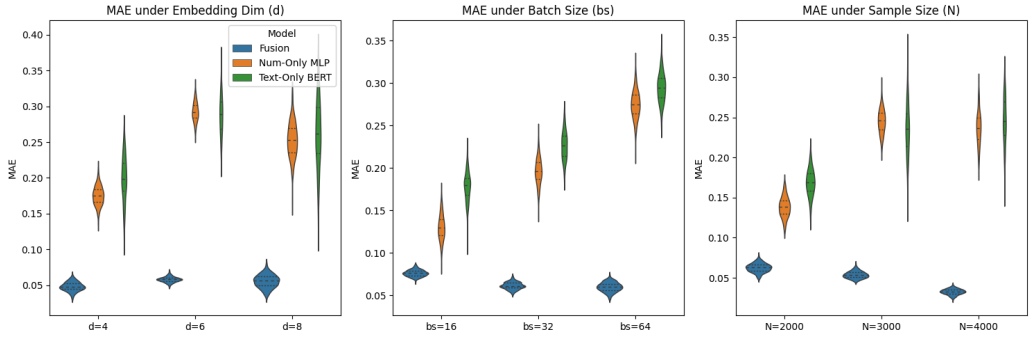


Figure 2. MAE distribution of each model under the baseline configuration and different parameters

As shown by the MSE and MAE values in Tables 1 and 2 and the violin plots in Figures 1 and 2, the proposed Fusion model achieves significantly better prediction performance and higher stability than the Num-Only MLP and Text-Only BERT models. Across all settings of embedding dimension d, batch size bs, and sample size N, Fusion consistently yields the lowest MSE and MAE, with a minimum MSE of 0.0408 and a minimum MAE of 0.0327. Its standard deviations are mostly below 0.008, much lower than those of the unimodal models, indicating more stable predictions. In contrast, Text-Only BERT has the highest error rates (maximum MSE of 0.3869 and maximum MAE of 0.2954), with Num-Only MLP performing in between. These results confirm that multimodal fusion effectively integrates numerical and textual features, compensates for the limitations of unimodal representations, and improves prediction accuracy.

The three models respond differently to parameter variations, further verifying the advantages of the Fusion model. As embedding dimension d rises, the Fusion model's MSE decreases continuously, showing that higher-dimensional embeddings strengthen its feature fusion ability, while both unimodal models show rising errors, implying overfitting. With increasing batch size bs, the Fusion model achieves steadily lower error and better training stability, whereas the unimodal models suffer from large error increases, indicating high sensitivity to batch size. As sample size N grows, the Fusion model achieves the most pronounced performance gains with sharply reduced MSE and MAE, reflecting strong data utilization efficiency. By contrast, the unimodal models show either rising errors or only minor improvements, revealing bottlenecks in feature learning that prevent effective performance gains from larger datasets.

DOI: 10.54254/2755-2721/2026.34024

3.2. Real data analysis

3.2.1. Data sources and experimental setup

The data used in this study comprise two types: numerical market data and unstructured textual data. The numerical data are sourced from the Wind financial database, a widely used high-quality data source in the domestic financial sector that provides multidimensional financial data including stocks, futures, and macroeconomics. The textual data are derived from the daily commodity market reviews and market summaries provided by the Wind database, covering supply and demand analyses, price dynamics, inventory changes, and market sentiment assessments for ferrous products such as rebar, iron ore, coke, stainless steel, ferrosilicon, and silicomanganese.

Table 3. Composition of the indicator system for real-world data analysis

ID	Feature Name	Unit	Definition and Description
return1	1-period lagged return	%	Logarithmic return of the closing price of black commodity contracts on the current day relative to the previous day's closing price.
open	Opening price	yuan/ton	Opening price of black commodity contracts on the current day.
high	Daily high price	yuan/ton	Highest price of black commodity contracts on the current day.
low	Daily low price	yuan/ton	Lowest price of black commodity contracts on the current day.
close	Closing price	yuan/ton	Closing price of black commodity contracts on the current day.
volume	Trading volume	lot	Total trading volume of black commodity contracts on the current day.
amount	Trading turnover	ten thousand yuan	Total trading amount of black commodity contracts on the current day.
up_down	Daily price change	%	Daily price fluctuation range of black commodity contracts on the current day.
hold	Open interest	lot	Market participation depth of black commodity contracts on the current day.
emo	Analyst sentiment index	—	Analyst sentiment indicator of black commodity contracts on the current day.
date	Date	—	The calendar date corresponding to the trading day.

The sample period ranges from January 1, 2023 to December 31, 2025, covering three full years of daily data. This time window is chosen for two main reasons. First, it includes several typical market stages such as post pandemic economic recovery, real estate policy adjustments and

DOI: 10.54254/2755-2721/2026.34024

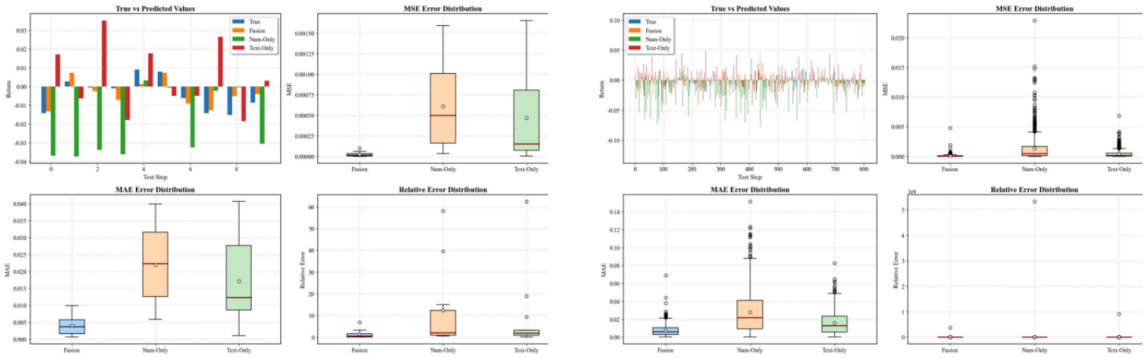
commodity price fluctuations. Second, the three year period provides about 750 trading days of samples. It meets the basic data demands of deep learning models and avoids market structural discontinuities that may appear in longer periods. A total of 22 numerical features are selected in this study, with detailed information listed in Table 3. The textual data come from daily ferrous commodity market summaries released by the Wind database. Each text consists of 300 to 800 Chinese characters and covers five core dimensions. The first dimension is supply and demand analysis, which mainly focuses on production, inventory and apparent consumption changes of rebar, iron ore and other products. The second is price dynamics, including spot and futures price trends as well as basis variations. The third is policy information, involving macroeconomic policies such as real estate regulations, environmental production restrictions and import and export tariffs. The fourth is market sentiment, including trader attitudes, winter stockpiling progress and pre holiday inventory replenishment. The fifth is international factors, covering shipment volumes of Australia and Brazil, freight changes and updates of international mining production.

The raw data include numerical and textual features. There are 22 numerical features: open price, high price, low price, close price, price change percentage, trading volume, trading amount, open interest, analyst sentiment, date, and 1-step to 12-step ahead returns. To reduce scale effects on model training, each numerical feature is standardized using Z-score normalization, The textual data are daily summaries of the ferrous commodity market, with lengths between 100 and 800 characters. To improve training efficiency, a pre-encoding strategy is used: all texts are encoded into `input_ids` and `attention_mask` in one pass using the BERT-base-Chinese tokenizer, with the maximum sequence length truncated or padded to 128. The target variable (next-day return of ferrous commodities) is also normalized, and its parameters are saved for later inverse transformation. A rolling window framework is adopted for model training and evaluation to eliminate look-ahead bias in time-series forecasting. Let the total length of the time series be T (total number of trading days), and the rolling window size be W. At the k -th prediction step ($k = 0,1,\ldots,T - W - 1$), the training set is defined as $\mathscr{D}_{\text{train}}^{(k)} = \left(xi, S_i, y_i\right), i = k^{k+W-1}$, comprising the data of the most recent W trading days, and the test set is defined as $\mathscr{D}test^{(k)} = \left(xk + W, S_{k+W}, y_{k+W}\right)$, consisting of the single sample for the next trading day. In the real-world data experiment, the total sample size is T = 728 trading days, and the rolling window size is set to W = 90 . Accordingly, 638 independent training and prediction steps are performed. In each step, the model is trained on the latest 90 days of data to predict the return on the 91st day. The window then rolls forward one day until all test samples are processed. All numerical features are Z-score normalized separately within each training window, using only current window training data to prevent future information leakage.

3.2.2. Analysis of experimental results

In the stock return prediction task, the three models exhibit significant differences in predictive performance and error distribution. A preliminary experiment using the first 100 trading days of data, shown in Figure 3(a), indicates that the proposed multimodal fusion model achieves substantially lower prediction error than the unimodal baselines. These results suggest that even with a limited sample size, the cross-attention mechanism effectively captures complementary information between numerical market signals and text-based sentiment features, leading to more accurate and stable predictions.

DOI: 10.54254/2755-2721/2026.34024



(a) sample size=100 (b) sample size=728

Figure 3. Comparative analysis of prediction performance across different models

Subsequently, we extend the experiment to the full dataset, as illustrated in Figure 3(b). The overall performance trends are consistent with the preliminary results: our model consistently outperforms all baseline methods across key evaluation metrics. The comparison between true and predicted values, shown in the upper-left panel of Figure 3(b), indicates that the predicted values of the Fusion model closely align with the actual returns in terms of both trend and fluctuation magnitude, achieving the best predictive accuracy and consistency. In contrast, the two unimodal baseline models, Num-Only and Text-Only, both exhibit pronounced systematic biases.

4. Conclusions

We face a core challenge in quantitative finance: merging number-based market data with unstructured text meaning. To tackle this challenge, this paper proposes an end-to-end stock return prediction method based on BERT and multimodal feature fusion. We use BERT-base-Chinese as the text encoder to pull out semantic features, and a multilayer perceptron to process number-based features. A cross-attention mechanism allows deep interaction and adaptive fusion between the two data types. We train all parts together in an end-to-end way inside a unified rolling window training framework.

We test our method with both numerical simulations and real-data backtesting. The real dataset comes from the ferrous commodity market and covers January 2023 to December 2025, giving us 728 trading days. Inside the rolling window framework, we run 638 separate training and prediction steps. The results show that our Fusion model works better than single-mode baselines on all three metrics: MSE, MAE, and relative error. Fusion gets much lower MSE and MAE than the numbers-only model, and it also beats the text-only model by a clear margin. This shows high prediction accuracy and strong robustness.

These results are promising, but we still have several directions to explore. First, this study only mixes text data and number data, but financial markets also have other rich information. This includes financial report images, earnings call audio, social media sentiment, and macroeconomic data. Future work could add more data types to our framework to build a fuller multimodal financial prediction system.Second, our rolling window framework is an offline backtesting method, which is different from real trading settings. Later research could put the prediction model into real quantitative trading systems with capital management and risk control modules. This would allow more realistic trading tests and help check the model's profit power and risk resistance.Third, BERT has many parameters and high inference delay, so meeting the low-delay needs of high-frequency trading is hard. Future work could use model compression methods like knowledge distillation and

DOI: 10.54254/2755-2721/2026.34024

model pruning. These methods could cut computing time and cost while keeping prediction accuracy. Then the model could work in tougher real-time trading settings.

References

[1] Boakes, J. (2024). An analysis of the performance and interpretability of machine learning classification algorithms to predict long-term share returns on the JSE. Master's Thesis, University of Cape Town.
[2] Moodi, F., Jahangard Rafsanjani, A., Zarifzadeh, S., & Zare Chahooki, M. A. (2024). Advanced Stock Price Forecasting Using a 1D-CNN-GRU-LSTM Model.Journal of AI and Data Mining, 12(3), 393-408.
[3] Araci, D. (2019). FinBERT: Financial Sentiment Analysis with Pre-trained Language Models.arXiv preprint, arXiv: 1908.10063.
[4] Yuntao Zhang, Zheng Dong, Wenrui Xu. (2025). Integrative stock price trend prediction via hierarchical LLM text processing and patch-based transformer with co-attention.Expert Systems with Applications.
[5] Akyildirim, E., Bariviera, A. F., Nguyen, D. K., & Sensoy, A. (2022). Forecasting high-frequency stock returns: a comparison of alternative methods.Annals of Operations Research, 313(2), 639-690.
[6] Shi et al. (2025). A Hybrid Long Short-Term Memory-Graph Convolutional Network Model for Enhanced Stock Return Prediction: Integrating Temporal and Spatial Dependencies.Mathematics, 13(7), 1142-1154.
[7] Chi-Yao Peng. (2025). Information-Processing Entropy and Heterogeneous Sentiment Reaction Windows: Evidence from S&P 500 Stocks.Entropy, 27(12), 1234.
[8] Bryan Lim, Sercan Ö. Arık, Nicolas Loeff, Tomas Pfister. (2025). TIC-FusionNet: A multimodal deep learning framework with temporal decomposition and attention-based fusion for time series forecasting.PLoS ONE, 20(10), e0333379.
[9] Chen, P., & Corizzo, R. (2024). A deep fusion model for stock market prediction with news headlines and time series data.Neural Computing and Applications, 36, 21229-21271.
[10] Ruan, L., & Jiang, H. (2025). Stock Price Prediction Using FinBERT-Enhanced Sentiment with SHAP Explainability and Differential Privacy.Mathematics, 13(17), 2747.
[11] Tu, P. L., & Chen, J. L. (2025). Mitigating market volatility impact in stock prediction: a Wasserstein GAN approach with GRU and CNN.ICASI 2025, IET Conference Proceedings.