mathematics

MDPI

Article

Multi-Scale TsMixer: A Novel Time-Series Architecture for Predicting A-Share Stock Index Futures

Zhiyuan Pei [1], Jianqi Yan [1], Jin Yan [1], Bailing Yang [1] and Xin Liu [2,*]

[1] School of Computer Science and Engineering, Faculty of Innovation Engineering, Macau University of Science and Technology, Macau 999078, China; 2109853eii30002@student.must.edu.mo (Z.P.); 2109853gii30001@student.must.edu.mo (J.Y.); 2109853gii30010@student.must.edu.mo (J.Y.); 3230002385@student.must.edu.mo (B.Y.)

[2] Macau Institute of Systems Engineering, Faculty of Innovation Engineering, Macau University of Science and Technology, Macau 999078, China

* Correspondence: xiliu@must.edu.mo

Abstract: With the advancement of deep learning, its application in financial market forecasting has become a research hotspot. This paper proposes an innovative Multi-Scale TsMixer model for predicting stock index futures in the A-share market, covering SSE50, CSI300, and CSI500. By integrating Multi-Scale time-series features across the short, medium, and long term, the model effectively captures market fluctuations and trends. Moreover, since stock index futures reflect the collective movement of their constituent stocks, we introduce a novel approach: predicting individual constituent stocks and merging their forecasts using three fusion strategies (average fusion, weighted fusion, and weighted decay fusion). Experimental results demonstrate that the weighted decay fusion method significantly improves the prediction accuracy and stability, validating the effectiveness of Multi-Scale TsMixer.

Keywords: deep learning; A-shares market; stock index futures; Multi-Scale TsMixer; component stock weighting; time-series prediction

MSC: 68T07; 91B84; 91G15

Academic Editors: Jian Luo, Zheming Gao and Xin Yan

Received: 22 March 2025
Revised: 19 April 2025
Accepted: 21 April 2025
Published: 25 April 2025

Citation: Pei, Z.; Yan, J.; Yan, J.; Yang B.; Liu X. Multi-Scale TsMixer: A Novel Time-Series Architecture for Predicting A-Share Stock Index Futures. *Mathematics* **2025**, *13*, 1415. https://doi.org/10.3390/math13091415

1. Introduction

Futures contracts are agreements to buy or sell a commodity at a future date at a price that is agreed upon today. They are important hedging tools for investors [1]. Stock index futures, derived from index values, are highly effective in minimizing investors' exposure to market volatility and addressing systemic risk. This is especially important in the Chinese market, where the difficulty of short-selling individual stocks heightens the significance of the short-selling mechanism in the futures market as a primary risk management strategy [2]. Acknowledging the significance of fostering financial market growth and employing effective risk management tools, China officially launched its first financial futures product, the Shanghai Shenzhen 300 (CSI300) index futures, in 2010 [3]. Subsequently, the Shanghai Stock Exchange 50 (SSE50) index futures and the China Securities 500 (CSI500) index futures were also launched. The launch of these stock index futures has facilitated an increase in trading volume and market liquidity, while emphasizing the importance of anticipating changes in the stock index futures market [4,5]. However, existing research indicates that the market does not fully align with the efficient market hypothesis [6,7]. As a result, predicting stock index futures prices is not only theoretically possible but also practical and valuable in real-world applications [8].

Long-term stock index futures prediction is a popular research area. Previous mainstream methods primarily rely on traditional econometric models [9] such as Logistic Regression (LOGIT) [10,11], autoregressive integrated moving average (ARIMA) [12,13], and generalized autoregressive conditional heteroscedasticity (GARCH) [14,15]. In addition, Zhe Lin [16] proposed an empirical analysis of the SSE Composite Index using GARCH-type models to examine its volatility, finding that the exponential GARCH (EGARCH) model outperforms others and offering suggestions for improving China's securities market stability. However, traditional econometric models struggle to effectively address the nonlinear relationships in financial data, resulting in lower prediction accuracy [17].

To address the limitations of traditional models, machine learning and deep learning have been increasingly applied in financial time-series forecasting, due to its ability to accurately capture complex nonlinear patterns in the data [18–21]. For example, Huang et al. investigated the predictability of financial movement direction. They used support vector machine (SVM) to forecast the weekly movement of the NIKKEI 225 index. The study demonstrated that SVM outperforms other methods [22]. Illa et al. proposed a stock price prediction methodology using random forest and SVM algorithms to forecast market trends, demonstrating that both models perform well, with SVM yielding the highest accuracy for time-series data [23]. Furthermore, Yang et al. focused on predicting stock market trends by analyzing emotions through news articles using natural language processing techniques and machine learning algorithms, finding that gradient boosting decision tree (GBDT) outperforms Adaboost, XGboost, decision tree, and logistic regression in accuracy [24].

Predicting stock index futures is essentially a time-series forecasting problem, and deep learning-based time-series models have become one of the hottest research topics in recent years [25]. For instance, Waswani et al. proposed the Transformer model, which replaced traditional recurrent networks with a self-attention mechanism, thus significantly improving parallelization and performance in sequence modeling tasks such as machine translation [26]. Building upon this, Zhou et al. developed Informer, a Transformer-based model tailored for long sequence time-series forecasting. Informer leveraged ProbSparse self-attention and a generative decoder to efficiently manage long-range dependencies, further enhancing prediction accuracy [27]. While these Transformer-based models showed strong performance, Zeng et al. questioned their effectiveness for long-term forecasting and presented DLinear, a simple linear model that outperformed complex Transformer models by more effectively capturing temporal relations with a single-layer architecture, suggesting that simpler approaches may be more suitable in some cases [28]. In contrast, Nie et al. introduced PatchTST, which enhanced Transformer models for multivariate time-series forecasting by employing a patching strategy to segment time series, thereby reducing computation and memory usage while improving the model's ability to capture long-term dependencies [29]. To address the complexity of temporal variations in time-series data, Wang et al. proposed TimeMixer, a multiscale mixing architecture that decomposes time series into seasonal and trend components, enabling more accurate predictions for both short-term and long-term forecasting [30]. Extending the idea of temporal variation modeling, Wu et al. developed TimesNet, which transformed time series into 2D tensors to capture both intra- and inter-period variations, achieving state-of-the-art performance in a range of time-series analysis tasks [31]. Moreover, Chen et al. presented TSMixer, an all-MLP architecture that efficiently captured both temporal and cross-variate dependencies using a novel mixing operation along both the time and feature dimensions, further contributing to advancements in time-series forecasting [32]. Due to its advantages, we will adopt this method.

Time-series forecasting, particularly in the financial domain, often faces the challenge of handling complex market volatility and the need for information across multiple time

scales [33]. Trading decisions in stock index futures are influenced not only by short-term fluctuations but also by long-term trends, such as weekly or monthly movements. However, most existing forecasting models primarily focus on short-term trends, neglecting cross-time scale information, which limits their effectiveness in addressing the complex dynamics of financial markets.

Moreover, many existing models rely on the historical data of stock index futures for overall modeling and forecasting [34]. This approach overlooks the fact that stock indices are composed of multiple weighted component stocks, resulting in models that capture only the overall trend of the index without accurately measuring the contribution of individual component stocks to index volatility. Due to the aggregation nature of stock indices, different component stocks may exhibit distinctly varying price changes and volatility characteristics at different times, which undermines the model's precision and accuracy.

To address these issues, this paper proposes three improved methods. First, based on the TSMixer model, this paper introduces a Multi-Scale module to better capture feature variations across different time granularities, such as 20-day and 60-day periods. This multi-path fusion strategy enables the model to retain daily details while effectively integrating weekly and monthly information, enhancing the accuracy and robustness of stock index futures forecasting. Second, considering that stock index futures are composed of multiple component stocks, this paper proposes using the historical data of all component stocks for training, and then inferring future trends of the stock index futures by weighting the forecast results of the individual component stocks during the prediction phase. This approach not only captures the unique characteristics of each component stock but also estimates their contributions to the overall index volatility, significantly enhancing the interpretability of the model. Third, to further explore the differentiated contributions of each component stock to the overall index in the prediction phase, three different weighted fusion methods are designed. These methods are average-based fusion, weighted-based fusion, and weighted-based decay fusion. In our experiments, the weighted-based decay fusion method considers both the weighted information of component stocks and the future expectation of the index at the same time and achieves the optimal predicted results.

The main contributions of this paper are as follows: (1) A deep learning-based Multi-Scale TsMixer model is proposed, which integrates a Multi-Scale time-series data processing strategy. This model performs feature fusion across multiple time scales (short term, medium term, and long term) to more comprehensively capture the short-term fluctuations and long-term trends in stock index futures history. (2) A stock index futures trend forecasting method combining independent component stock predictions with weighted fusion is proposed. (3) Three different weighted fusion strategies (average-based fusion, weighted-based fusion, and weighted-based decay fusion) are designed and systematically evaluated for their forecasting performance. Experimental results demonstrate that the weighted-based decay fusion method significantly outperforms the other methods in terms of prediction accuracy and stability.

2. Methodology

In this work, we propose a method for predicting stock index futures price movements, comprising the three main components: (1) building baseline models and proposing a more advanced network architecture, (2) adopting an individual training strategy for each constituent stock, and (3) applying a weighted fusion of the prediction outputs. The integration of these three components yields superior predictive performance. This study addresses two distinct tasks: a regression task predicting the percentage change in

stock index futures prices and a binary classification task determining whether prices will increase or decrease.

2.1. Baselines and Proposed Model

We employ various advanced time-series models with different network architectures as baselines, including Transformer [26], Informer [27], DLinear [28], PatchTST [29], TimeMixer [30], TimesNet [31], and TSMixer [32], to address both regression and classification tasks in the experiments. To explore the effect of varying sequence lengths on model prediction accuracy, we configure the input data sequences for each model to lengths of 20, 60, and 80 during the baseline setup phase, thereby capturing different time scales. By comparing these sequence lengths, we then evaluate the models' predictive performance under varying amounts of historical information.

Among these baseline models, TsMixer is implemented based on a pure multilayer perceptron (MLP) architecture. Pure MLP architectures are frequently utilized as baseline models for various complex networks [35]. The primary advantages of pure MLPs include their simplicity, ease of implementation, parallelizability, and versatility. Additionally, when modeling high-dimensional or structured inputs, pure MLP architectures do not depend on specific structural priors, such as convolutional layers or self-attention mechanisms, yet they consistently deliver a stable performance. Consequently, we select TsMixer, which is founded on a pure MLP architecture, to innovate and enhance the network architecture in this work.

Typically, trading decisions for stock index futures require reference to longer timescales, such as weekly or monthly trends. To better capture feature variations across different temporal granularities (e.g., 5-day and 20-day), we incorporate a Multi-Scale module into TsMixer to enhance predictive accuracy. As illustrated in Figure 1, we refer to this enhanced architecture as the Multi-Scale TsMixer. Specifically, we apply average pooling to the daily data to obtain weekly feature sequences at a resolution of 5 and monthly feature sequences at a resolution of 20, ensuring the exclusion of weekends when forming weekly or monthly intervals.

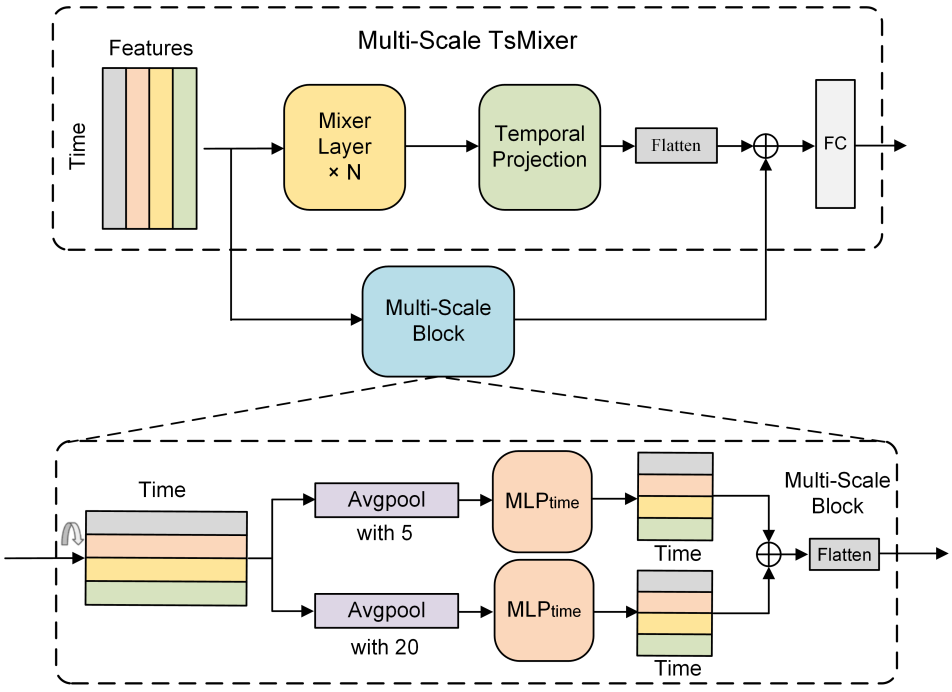

Figure 1. Illustration of our proposed Multi-Scale TsMixer network architecture.

Next, each weekly and monthly feature sequence passes through a small MLP for independent feature extraction, after which we concatenate these features with the original daily Tsmixer outputs. Consequently, we extend Tsmixer to embed Multi-Scale (daily, weekly, and monthly) information within the temporal dimension, thereby increasing the learnable input features. Finally, we project and fuse these Multi-Scale features to generate the final prediction. This multi-path fusion strategy not only preserves finer daily-level details but also integrates weekly and monthly information, aligning with the need to consider multiple time horizons in financial markets. As a result, the model achieves higher accuracy and robustness in capturing both short-term fluctuations and medium-to long-term trends in stock index futures.

2.2. Individual Training Strategy

In Section 2.1, we describe the network architecture of our proposed Multi-Scale TsMixer. In this section, we introduce a novel training strategy for the proposed model. In the field of stock index futures prediction, many previous studies directly utilize the historical data of stock index futures for overall modeling and prediction [19,36]. These approaches often overlook the fact that an index is composed of multiple constituent stocks with varying weights. As a result, the models are limited to capturing only the overall trend and are unable to accurately assess the differential contributions of individual constituent stocks to index fluctuations.

Due to the aggregative nature of indices, constituent stocks can exhibit significantly varying magnitudes of price changes and volatility characteristics across different periods. While stock index futures generally capture the overall trend, those approaches tend to obscure or attenuate the unique characteristics of individual stocks, thereby diminishing the precision and accuracy of the predictive models.

To address the aforementioned limitations, we utilize the historical data of all constituent stocks for training and predict stock index futures trends by weighting the predicted outcomes of individual constituent stocks during the prediction phase (detailed in Section 2.3). For example, when the model is trained exclusively on the index futures' historical data, it primarily learns a composite index curve. This approach hinders the accurate capture of each constituent stock's unique characteristics and weight variations. Conversely, by conducting parallel training on the constituent stocks, the model can learn each stock's distinct features and aggregate their predicted results using their respective weights to predict the overall index movement. This individual training strategy enables the explicit estimation of each stock's contribution to the index, thereby enhancing the interpretability and precision of the prediction results.

On the left side of Figure 2, we present the individual training strategy for constituent stocks. Taking the SSE50 Index as an example in Figure 2, we decompose the explicit features of stock index futures into 50 latent features, each representing an individual constituent stock. These latent features are then utilized as inputs to the time-series model, thereby capturing the unique characteristics of each constituent stock.

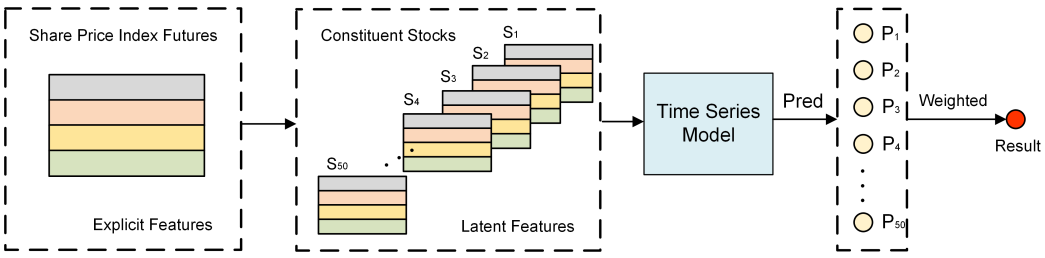

Figure 2. Workflow of predicting index futures with aggregating constituent stock's predicted results.

2.3. Fusion Method for Prediction

In the model prediction phase, to further investigate the differential contributions of individual constituent stocks to the overall index (as shown in the right side of Figure 2), we propose three different fusion methods. By weighting and integrating the prediction outcomes of each constituent stock, these methods effectively enhance the accuracy of predicting the rise and fall of stock index futures. The three fusion methods include average fusion, weighted fusion, and weighted decay fusion, are shown in Algorithm 1.

Algorithm 1 Predicting stock index futures using Multi-Scale TsMixer and fusion methods.

Input: Historical data of constituent stocks $\mathbf{S} = \{S_1, S_2, ..., S_n\}$, trained model $\mathcal{M}$, fusion method f_{fusion}, optional weights ω, optional decay factors α, current time step t.
Output: Predicted index futures value $\hat{Y}_{\mathbf{S},t+1}$.
Initialization: $\hat{Y}_{\mathbf{S},t+1} \leftarrow 0$
Step 1: Generate Individual Stock Predictions
for each stock $S_i \in \mathbf{S}$ **do**
    $\hat{y}_{S_i,t+1} \leftarrow \mathcal{M}(S_i, t)$ ▷ Predict next-day value
end for
Step 2: Apply Fusion Method
if $f_{\text{fusion}} =$ "average" **then**
    $\hat{Y}_{\mathbf{S},t+1} \leftarrow \frac{1}{n}\sum_{i=1}^{n} \hat{y}_{S_i,t+1}$ ▷ Simple average fusion
else if $f_{\text{fusion}} =$ "weighted" **then**
    $\hat{Y}_{\mathbf{S},t+1} \leftarrow \sum_{i=1}^{n} \omega_i \hat{y}_{S_i,t+1}$ ▷ Weighted fusion
else if $f_{\text{fusion}} =$ "weighted decay" **then**
    for each stock $S_i \in \mathbf{S}$ **do**
        $y_{S_i,\text{decay}} \leftarrow \sum_{k=1}^{n} \alpha_k \hat{y}_{S_i,t+k}$ ▷ Apply decay factor
    end for
    $\hat{Y}_{\mathbf{S},t+1} \leftarrow \sum_{i=1}^{n} \omega_i y_{S_i,\text{decay}}$ ▷ Decay-weighted fusion
end if
return $\hat{Y}_{\mathbf{S},t+1}$

2.3.1. Average-Based Fusion Method

Building on the independent predictions of each constituent stock, we first adopt the common average-based fusion method. Specifically, we compute the simple mean of the predicted expected values for all constituent stocks. For multiple constituent stocks $\mathbf{S} = \{S_1, S_2, \ldots, S_n\}$, the model's prediction for each stock on the next trading day can be expressed as:

$$\hat{y}_{S_i,t+1} = \mathcal{M}(S_i, t) \tag{1}$$

where $\hat{y}_{S_i,t+1}$ denotes the predicted outcome for stock S_i at time $t+1$, and $\mathcal{M}(S_i, t)$ represents the trained model that produces the next-day forecast based on information up to time t. We then combine these individual predictions by taking the simple mean across all stocks in to derive the average-based forecast for the index. The futures prediction obtained by the average-based fusion method can be formulated as follows:

$$\hat{Y}_{\mathbf{S},t+1} = \frac{1}{n}\sum_{i=1}^{n} \hat{y}_{S_i,t+1} \tag{2}$$

where n represents the number of constituent stocks.

However, because the average-based fusion method disregards differences in the constituent stocks' weights within the index, it fails to capture the more pronounced influence of heavily weighted constituents on the overall index. As a result, fluctuations in these key constituents may be diluted or overlooked.

2.3.2. Weighted-Based Fusion Method

To address the average-based fusion method's inability to account for the distinct influence of each constituent stock, we propose a weighted-based fusion method. This approach assigns weights to each stock's predicted outcome based on factors such as market capitalization and trading volume, and then aggregates these weighted predictions to produce the overall stock index futures forecast. By highlighting stocks with higher contributions to the index, it more accurately captures the structural relationships within the market. The weighted fusion formula is given by:

$$\hat{Y}_{\mathbf{S},t+1} = \sum_{i=1}^{n} \omega_i \hat{y}_{S_i,t+1} \tag{3}$$

where ω_i denotes the weighting coefficient of stock S_i and $\sum_{i=1}^{n} \omega_i = 1$.

However, there is a certain limitation in the weighted-based fusion method. Because an expectation gap exists between the spot market and the futures market, this method establishes only a relatively static weight mapping between each constituent stock and the index, and does not account for the potential dynamic linkage that may arise between stock index futures and the spot market over time.

2.3.3. Weighted-Based Decay Fusion Method

To better reflect the potential expectation gap and asynchrony between futures and spot markets, we further propose a weighted-based decay fusion method. In this approach, we incorporate the future price fluctuations of each constituent stock, and the corresponding weighting formula is given as follows:

$$\alpha_k = \begin{cases} 0.5^k, & \text{if } k = 1, 2, \ldots, (n-2) \\ 0.5^{n-1}, & \text{if } k = (n-1), n \end{cases} \tag{4}$$

where n represents the number of total future trading day and $\sum_{k=1}^{n} \alpha_k = 1$.

By using the weighted-based decay fusion method, the model more effectively captures short-term continuity or leading signals during training, highlighting the dynamic interactions between constituent stocks and futures in the merged outcomes. This approach not only retains the benefits of the weighted-based fusion method, but also incorporates a decaying mechanism to integrate short-to medium-term future information, thereby providing a more nuanced understanding of each constituent stock's influence on the overall index. Besides reflecting the dynamic relationships between constituent stocks and futures, it also enhances the synchronization and accuracy of predictions.

3. Experiments

3.1. Dataset

In this experiment, the dataset is sourced from the SSE50, CSI300, and CSI500 stock index futures and their corresponding constituent stock data. The historical composition of constituent stocks for each stock index future is obtained through the Tonghuashun platform. As is well known, the composition of constituent stocks in an index is updated over time.

The dataset covers daily trading data from 1 January 2016 to 31 October 2024. For each stock index future, there are 16 input features, including opening price, closing price, highest price, lowest price, trading volume, and others. Additionally, for each constituent stock within the stock index futures, the data include not only the same features as the index futures but also additional characteristics such as industry categories and market

capitalization, which are used as model inputs. A detailed description of the input features and calculation methods is presented in Table A1 of Appendix A.

We divide the dataset chronologically into three parts: the training set (from 1 January 2016 to 31 December 2020), the validation set (from 1 January 2020 to 31 December 2021), and the test set (from 1 January 2022 to 31 October 2024). Since CSI 500 futures were introduced on 16 April 2015, and our experiment incorporates a maximum output sequence length of 80, we reserve a portion of the historical data for model inputs and begin the training set on 1 January 2016. Additionally, to more accurately evaluate the performance of different methods and models, we utilize as much data as possible for validation and testing. Therefore, we use two years of data for the validation set and about two years of data for the test set.

3.2. Experimental Setting

To ensure the fairness of our experiments, all models are compared under the same dataset and evaluation metrics, using a consistent random seed. The batch size is set to 128 in the training part. The initial learning rate is fixed at 0.0005, and parameter updates are performed via the Adam optimizer. The mean squared error (MSE) is adopted as the loss function, defined by the following formula:

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \tag{5}$$

where y_i represents the true value and $\hat{y}_i$ denotes the predicted value for the ith observation. In order to flexibly adjust the learning rate, we adopt a dynamic decay strategy under which the learning rate is reduced to one-fifth of its former value if the validation loss remains unimproved for 15 consecutive epochs. We also employ early stopping to prevent overfitting, halting training whenever the validation performance does not improve for 25 consecutive epochs. These hyperparameter settings and techniques are uniformly applied to all baseline methods as well as our proposed Multi-Scale TsMixer model, thereby ensuring experimental consistency. Finally, we evaluate performance for each trained model on the SSE50, CSI300, and CSI500 stock index datasets.

3.3. Evaluation Criteria

In our regression task, we aim to predict the percentage increase or decrease in stock index futures prices. To evaluate model performance under this framework, we employ several metrics commonly used in regression, including mean absolute error (MAE), mean absolute percentage error (MAPE), and the coefficient of determination (R^2). Their definitions are given by:

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}\left|y_i - \hat{y}_i\right| \tag{6}$$

$$\text{MAPE} = \frac{1}{n}\sum_{i=1}^{n}\left|\frac{y_i - \hat{y}_i}{y_i}\right| \tag{7}$$

$$R^2 = 1 - \frac{\sum_{i=1}^{n}\left(y_i - \hat{y}_i\right)^2}{\sum_{i=1}^{n}\left(y_i - \bar{y}\right)^2} \tag{8}$$

where y_i represents the true value, $\bar{y}$ represents the mean value, and $\hat{y}_i$ denotes the predicted value for the ith observation. Moreover, in practical stock index futures trading, we typically initiate long or short positions based on the model's predicted trends. Because these

regression metrics do not directly reflect profit or loss, we further treat the outputs as a binary classification. To achieve a more intuitive assessment of trading performance, we adopt precision, specificity, accuracy, and the Sharpe ratio:

$$\begin{aligned} \text{Precision} &= \frac{\text{TP}}{\text{TP}+\text{FP}} \\ \text{Specificity} &= \frac{\text{TN}}{\text{TN}+\text{FP}} \\ \text{Accuracy} &= \frac{\text{TP}+\text{TN}}{\text{TP}+\text{TN}+\text{FP}+\text{FN}} \end{aligned} \tag{9}$$

where TP, FP, TN, and FN refer to the numbers of true positives, false positives, true negatives, and false negatives, respectively.

4. Experimental Results and Analysis

In this section, we comprehensively compare and analyze the performance of each model in both regression and binary classification tasks, employing multiple metrics (MAPE, MAE, R^2, precision, specificity, and accuracy) across different sequence lengths and indices. To streamline the discussion, we provide the regression and classification results for the CSI500 stock index in Tables 1 and 2, respectively, while supplementary findings for the CSI300 and SSE50 indices appear in Appendixes B and C (Tables A2–A5).

Table 1. Performance comparison of baseline models and Multi-Scale TsMixer on the CSI500 regression task for different input sequences (upper arrows indicate that larger values are better; lower arrows vice versa).

	Model	Days	MAPE (↓)	MAE (↓)	R^2 (↑)
Baselines	TSMixer	20	0.9742	55.2110	0.9802
		60	0.9736	55.1976	0.9803
		80	0.9728	55.1477	0.9803
	Informer	20	0.9804	55.6039	0.9801
		60	0.9942	56.4421	0.9796
		80	0.9714	55.1211	0.9802
	DLinear	20	0.9730	**55.1466**	**0.9803**
		60	**0.9718**	**55.0922**	0.9803
		80	0.9697	54.9805	0.9803
	PatchTST	20	1.2085	69.2901	0.9738
		60	1.1883	68.0490	0.9741
		80	1.0850	61.7578	0.9778
	Transformer	20	0.9828	55.7572	0.9801
		60	**0.9718**	55.1575	0.9802
		80	0.9748	55.2661	0.9803
	TimesNet	20	0.9881	56.1015	0.9798
		60	0.9793	55.5010	0.9802
		80	0.9694	54.9557	0.9803
	TimeMixer	20	0.9768	55.3756	0.9802
		60	0.9731	55.2106	0.9802
		80	0.9736	55.2130	0.9803
Multi-Scale TsMixer (ours)		20	**0.9729**	55.1629	**0.9803**
		60	0.9724	55.1209	**0.9804**
		80	**0.9682**	**54.9034**	**0.9804**

Table 2. Performance comparison of baseline models and Multi-Scale TsMixer on CSI500 classification task for different input sequences (upper arrows indicate that larger values are better; lower arrows vice versa).

	Model	Days	Precision (↑)	Specificity (↑)	Accuracy (↑)
Baselines	TSMixer	20	0.4821	0.5465	0.5297
		60	0.4906	0.5530	0.5336
		80	0.5109	0.5521	0.5439
	Informer	20	0.4889	0.5637	0.5292
		60	0.4850	0.5678	0.5234
		80	0.4912	0.5698	0.5307
	DLinear	20	0.4930	0.5625	0.5336
		60	0.5031	0.5773	0.5424
		80	0.5123	0.5764	0.5497
	PatchTST	20	0.4448	0.5251	0.4868
		60	0.4583	0.5379	0.5044
		80	0.4825	0.5583	0.5234
	Transformer	20	0.4881	0.5656	0.5222
		60	0.4980	0.5686	0.5395
		80	0.4891	0.5791	0.5336
	TimesNet	20	0.4909	0.5674	0.5307
		60	0.4939	0.5702	0.5338
		80	0.5079	0.5804	0.5468
	TimeMixer	20	0.4952	0.5691	0.5321
		60	0.5121	0.5763	0.5439
		80	0.5100	0.5703	0.5395
Multi-Scale TsMixer (ours)		20	**0.4972**	**0.5707**	**0.5365**
		60	**0.5138**	**0.5788**	**0.5482**
		80	**0.5203**	**0.5918**	**0.5526**

This is a detailed explanation of the tables (see Tables 1, 2, A2–A5). In all the tables mentioned, upper arrows indicate that larger values are better; lower arrows vice versa. Green, red, and black represent the results that stand out for the 20, 60, and 80-day input sequences, respectively. Bold numbers indicate the best results, and double-underlined numbers indicate the suboptimal results.

4.1. Regression Results

In this regression task, we systematically evaluate the performance of each baseline model (TSMixer, Informer, DLinear, PatchTST, Transformer, TimesNet, TimeMixer) alongside the proposed Multi-Scale TsMixer under input sequences of 20, 60, and 80 days (see Table 1).

Generally, as the input sequence length increases, models capture more historical data, resulting in steady improvements in prediction performance. This trend is particularly evident with longer sequences (60 and 80 days). For instance, from Table 1, with DLinear, the MAE error is 55.147 for a 20-day sequence. However, when the sequence length increases to 60 and 80 days, the error decreases by 0.055 and 0.167, respectively.

It is noteworthy that the proposed Multi-Scale TsMixer demonstrates consistent stability across all input lengths, progressively leveraging the advantages of its Multi-Scale features as the input sequence length increases. For instance, at 20 and 60 days, the performance of Multi-Scale TsMixer and DLinear is relatively close, but at an 80-day input, Multi-Scale TsMixer obviously outperforms the other baseline models in terms of MAPE, MAE, and R^2. This further validates the efficacy of the Multi-Scale (weekly, monthly) fea-

ture fusion strategy in capturing longer-term market trends and subtle fluctuations. These findings not only highlight the importance of utilizing multi-level information in long sequences, but also demonstrates that the proposed method can effectively accommodate varying time scales, offering greater applicability and robustness in predicting stock index futures price changes.

To comprehensively evaluate the performance of our model, we further compare its results across different indices. The experimental results for the CSI300 and SSE50 indices are detailed in Appendixes B and C, respectively. The findings reveal that the proposed Multi-Scale TsMixer model consistently delivers the optimal performance across all these indices, demonstrating the robustness of our proposed method and its capability to yield accurate predictions across diverse market conditions.

4.2. Classification Results

Additionally, we further validate the model's effectiveness in trading strategies through the binary classification task (up/down) and evaluate its accuracy during simulated trading using metrics such as precision, specificity, and accuracy. As illustrated in Table 2 and Figure 3, the performance trends of most models remain consistent. Interestingly, we find that improvements in regression metrics (MAE, MAPE) are often accompanied by higher classification accuracy. Our proposed model consistently achieves the highest prediction accuracy across various indices. Figure 3 presents a comparison of accuracy variations across different input sequence lengths for multiple models. In the classification task, our proposed model consistently outperforms all baseline models, regardless of the input sequence length.

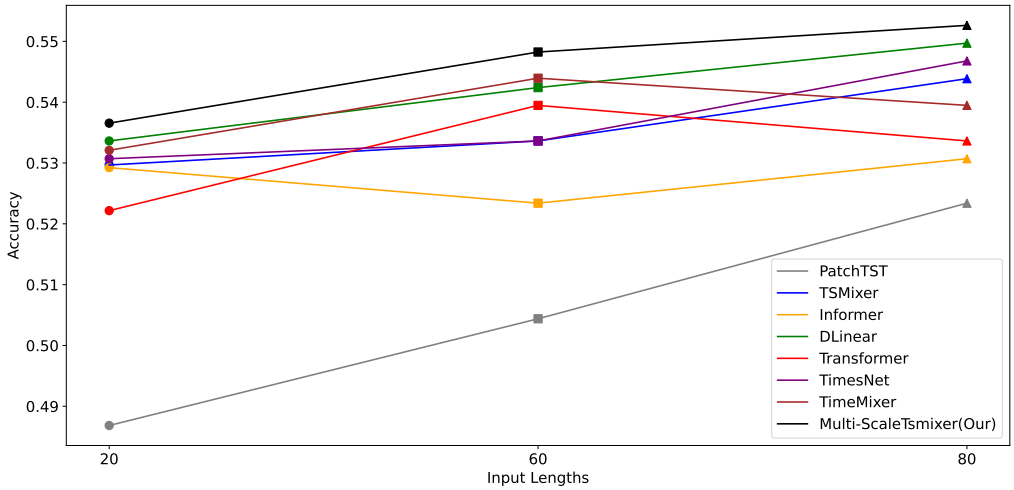

Figure 3. Comparison of the accuracy of different baseline models and our proposed model across various input lengths.

4.3. Weighted Fusion Results

After validating the model with overall index data, we turn our attention to the internal structure of the index, performing more detailed predictions at the constituent stock level. Specifically, we independently model each constituent stock and apply different weighting fusion strategies during the prediction phase to aggregate multiple stock forecasts into a composite prediction for the overall index. This approach enables the explicit differentiation of the contribution that each constituent stock makes to the index's movement. By integrating this method with the previously proposed Multi-Scale TsMixer network structure, we further improve the model's ability to capture both short-term fluctuations and long-term trends in stock index futures.

In this phase, we use the best-performing Multi-Scale TsMixer with an 80-day sequence length from the previous subsection as the baseline. We then compare several fusion methods and their performance. In the experiments, we compare three fusion methods, including average-based fusion, weighted-based fusion, and weighted-based decay fusion methods. In Table 3, the experimental results show that the weighted-based decay fusion method delivers the best results across all evaluation metrics. This indicates the weighted-based decay fusion method's superiority in capturing the dynamic relationships between constituent stocks and stock index futures.

Table 3. Classification performance of different fusion methods in the Multi-Scale TsMixer on three indices with 80-day input lengths (bold indicates best performance; upper arrows indicate that larger values are better; lower arrows vice versa).

Index	Fusion Method	Precision (↑)	Specificity (↑)	Accuracy (↑)
CSI500	Average-based Fusion	0.5463	0.6091	0.5787
	Weighted-based fusion	0.5461	0.6156	0.5856
	Weighted-based decay fusion	**0.5867**	**0.6414**	**0.6185**
CSI300	Average-based fusion	0.5367	0.5989	0.5707
	Weighted-based fusion	0.5449	0.6126	0.5857
	Weighted-based decay fusion	**0.5745**	**0.6326**	**0.6075**
SSE50	Average-based fusion	0.5398	0.5959	0.5739
	Weighted-based fusion	0.5434	0.6095	0.5800
	Weighted-based decay fusion	**0.5685**	**0.6287**	**0.6002**

The experimental results also show that the average-based fusion method, by failing to account for the varying importance of constituent stocks in the index composition, results in moderate accuracy. The weighted-based fusion enhances the alignment with the index structure by applying the weights based on market capitalization or liquidity, which improves predictive accuracy, but it overlooks the potential expectation gap between futures and spot markets, leading to the insufficient capture of dynamic relationships.

In contrast, the weighted-based decay fusion method achieves a more significant breakthrough. By incorporating a short-term decay factor into the training labels or prediction process, it more accurately reflects the expectation gap between futures and spot markets, resulting in superior performance across accuracy and other metrics compared to the other methods.

The further analysis of the performance of these fusion methods across various indices (CSI500, CSI300, and SSE50) shows that the weighted-based decay fusion method outperforms the others in all indices. For instance, in the CSI500 index, the accuracy (0.61848) and specificity (0.64136) of the weighted-based decay fusion method significantly exceed those of the average-based and weighted-based fusion methods. In Table 3, the weighted-based decay fusion method also shows notable improvements in accuracy and specificity on the CSI300 and SSE50 indices. Particularly for the SSE50 index, the weighted-based decay fusion method achieves an accuracy of 0.60018, which is approximately 4% higher than the accuracy of the average-based (0.57394) and weighted-based fusion methods (0.57998).

In conclusion, introducing the weighted decay mechanism significantly enhances the prediction synchronization and accuracy in the dynamic relationship between constituent stocks and stock index futures. The weighted-based decay fusion method demonstrates distinct advantages across different indices and fusion strategies, particularly in capturing the dynamic relationships and short-term fluctuations between constituent stocks and stock index futures, highlighting its broad potential for stock index futures prediction.

Finally, we backtest the performance of the Multi-Scale TsMixer in both independently predicting index futures and merging forecasts from individual components on the test set over the period from 1 January 2022 to 31 October 2024. In the return calculation, we incorporate both the predicted and actual signals. Specifically, let $\hat{y}_t$ denote the predicted signal on day t and y_t denote the actual signal on day t (with 1 corresponding to an upward movement and 0 corresponding to a downward movement). We define the indicator function as

$$I(t) = \begin{cases} 1, & \text{if } \hat{y}_t = y_t, \\ 0, & \text{if } \hat{y}_t \neq y_t. \end{cases}$$

Then, we introduce the conversion factor δ_t, defined as

$$\delta_t = 2\, I(t) - 1.$$

Thus, when the predicted signal matches the actual signal, $\delta_t = 1$; otherwise, $\delta_t = -1$. The cumulative return over the entire backtesting period is calculated as

$$R_{\text{total}} = \prod_{t=1}^{T} (1 + r_t) - 1,$$

where the net return on day t, r_t, is defined as

$$r_t = \delta_t \cdot \frac{|\text{price}_t - \text{price}_{t-1}|}{\text{price}_{t-1}} - f,$$

where price_t denotes the closing price of the futures on day t, and f is the transaction fee, set at 0.000023 (i.e., 0.23 per 10,000). Figure 4 shows the return comparison between the single-future input and the constituent-stock weighted approach for three index futures with an 80-day feature length.

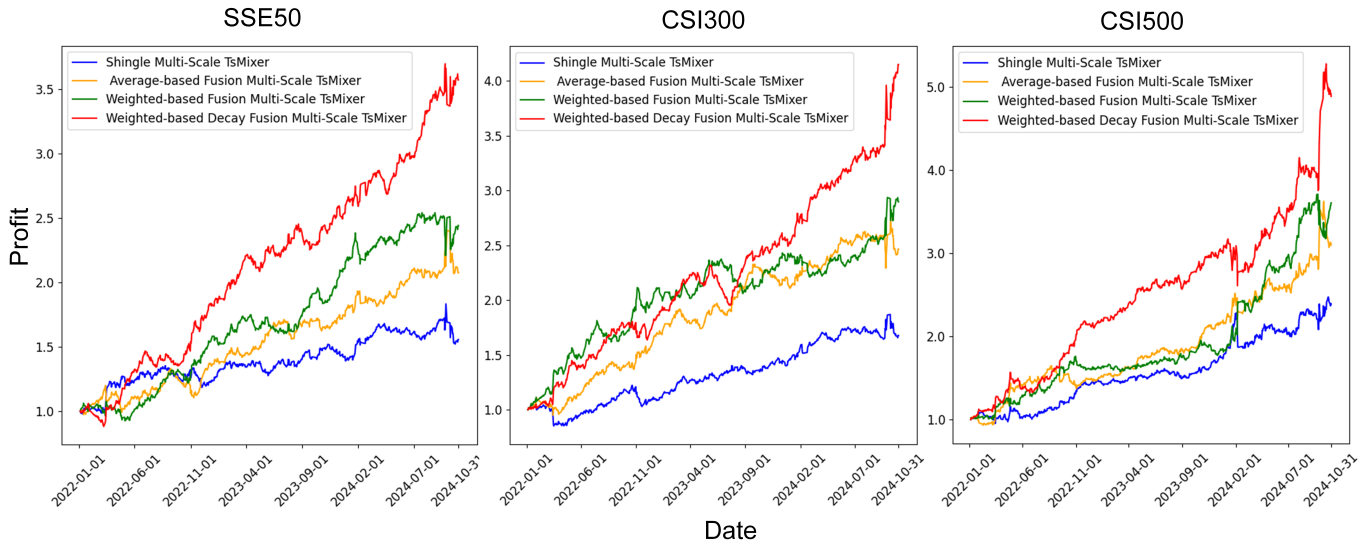

Figure 4. A comparative comparison of profits between the single-future and three fusion methods on three indices.

Table 4 demonstrates that our weighted decay fusion method consistently achieves superior returns across various stock index futures. In addition, the approach achieves a notably lower maximum drawdown and enhanced profitability compared to single-future based model, underscoring the remarkable efficacy and robustness of the proposed fusion

strategy. Moreover, continuous iterations and optimizations of the fusion method lead to a steady increase in returns across the three stock index futures.

Table 4. Comparison of profitability and maximum drawdown for different index futures using various fusion methods (bold indicates best performance; upper arrows indicate that larger values are better; lower arrows vice versa).

Index	Method	Profit (%) (↑)	Maximum Drawdown (%) (↓)
SSE50	Single-future	55.52	17.05
	Average-based fusion	107.33	16.38
	Weighted-based fusion	144.06	**14.00**
	Weighted-based decay fusion	**257.10**	14.89
CSI300	Single-future	67.73	19.21
	Average-based fusion	146.47	13.99
	Weighted-based fusion	189.55	**13.26**
	Weighted-based decay fusion	**314.81**	15.74
CSI500	Single-future	139.18	18.83
	Average-based fusion	210.02	16.52
	Weighted-based fusion	260.58	**14.40**
	Weighted-based decay fusion	**388.71**	17.74

5. Conclusions and Future Works

This paper presents an innovative stock index futures prediction framework, Multi-Scale TsMixer.Experimental results demonstrate that the Multi-Scale TsMixer model excels in long time-series data, effectively capturing both short-term fluctuations and long-term trends. In addition, we present a deep learning-based approach for stock index futures prediction, integrating the independent modeling of constituent stocks with Multi-Scale time-series fusion. By designing three fusion methods, particularly introducing a weighted decay mechanism, the proposed approach significantly enhances both predictive accuracy and stability.

While the proposed method delivers promising results, there remains room for further optimization. We will provide a more comprehensive assessment of the model's generalizability on major international indices such as the DJI30, S&P500, and FTSE100 in future work.Future research will focus on enhancing the model's ability to handle market volatility, improving computational efficiency, and incorporating additional market factors to further strengthen predictive performance.

Author Contributions: Conceptualization, Z.P. and J.Y. (Jianqi Yan); Methodology, Z.P.; Supervision, X.L.; Software, Z.P.; Validation, B.Y.; Writing—original draft preparation, Z.P., J.Y. (Jianqi Yan) and J.Y. (Jin Yan); Writing—review and editing, X.L., Z.P., J.Y. (Jianqi Yan), J.Y. (Jin Yan) and B.Y. All authors have read and agreed to the published version of the manuscript.

Funding: This research is supported by the joint research and Development fund of Wuyi University, Hong Kong and Macao (2019WGALH20), Macao Science and Technology Development Fund (No. 0013/2021/ITP).

Data Availability Statement: Data available upon request from the authors.

Conflicts of Interest: The authors declare no conflicts of interest.

Appendix A

Table A1. List of feature names, corresponding equations, and descriptions for financial explanations of each equation.

Feature Names	Equations	Description
`close_ratio`	$\frac{\text{close}_t - \text{close}_{t-1}}{\text{close}_{t-1}}$	The ratio of the change in the closing price from the previous day to the previous day's closing price.
`open_ratio`	$\frac{\text{open}_t - \text{close}_{t-1}}{\text{close}_{t-1}}$	The ratio of the change in the opening price from the previous day's closing price.
`high_ratio`	$\frac{\text{high}_t - \text{close}_{t-1}}{\text{close}_{t-1}}$	The ratio of the change in the highest price from the previous day's closing price.
`low_ratio`	$\frac{\text{low}_t - \text{close}_{t-1}}{\text{close}_{t-1}}$	The ratio of the change in the lowest price from the previous day's closing price.
`swing_volatility_5_0`	$\frac{\sum_{i=1}^{5} \lvert \text{price}_i - \text{price}_{i-1} \rvert}{5}$	A measure of volatility over the last 5 periods, capturing swings in price.
`volume_ratio`	$\frac{\text{volume}_t - \text{volume}_{t-1}}{\text{volume}_{t-1}}$	The ratio of the change in trading volume from the previous period's volume.
`ma5_ratio`	$\frac{\text{MA5}_t - \text{MA5}_{t-1}}{\text{close}_{t-1}}$	The change in the 5-period moving average from the previous day's 5-period moving average, normalized by the previous day's closing price.
`ma20_ratio`	$\frac{\text{MA20}_t - \text{MA20}_{t-1}}{\text{close}_{t-1}}$	The change in the 20-period moving average from the previous day's 20-period moving average, normalized by the previous day's closing price.
`ma60_ratio`	$\frac{\text{MA60}_t - \text{MA60}_{t-1}}{\text{close}_{t-1}}$	The change in the 60-period moving average from the previous day's 60-period moving average, normalized by the previous day's closing price.
`rsi_tt_3`	$100 - \frac{100}{1 + \frac{\sum_{i=1}^{3} \text{gain}_i}{\sum_{i=1}^{3} \text{loss}_i}}$	The 3-period relative strength index (RSI), which measures overbought or oversold conditions over a short time frame.
`rsi_tt_6`	$100 - \frac{100}{1 + \frac{\sum_{i=1}^{6} \text{gain}_i}{\sum_{i=1}^{6} \text{loss}_i}}$	The 6-period RSI, similar to the 3-period but calculated over a longer window.
`rsi_tt_12`	$100 - \frac{100}{1 + \frac{\sum_{i=1}^{12} \text{gain}_i}{\sum_{i=1}^{12} \text{loss}_i}}$	The 12-period RSI, commonly used to evaluate market trends over a medium time span.
`rsi_tt_14`	$100 - \frac{100}{1 + \frac{\sum_{i=1}^{14} \text{gain}_i}{\sum_{i=1}^{14} \text{loss}_i}}$	The 14-period RSI, the standard time period for calculating RSI in financial analysis.
`day_of_week`	$\text{day}_t \in \{1, 2, 3, 4, 5, 6, 7\}$	The day of the week, represented numerically from Monday = 1 to Sunday = 7.
`day_of_month`	$\text{day}_t \in \{1, 2, 3, \ldots, 31\}$	The day of the month, ranging from 1 to 31, representing the specific day in a given month.
`day_of_year`	$\text{day}_t \in \{1, 2, 3, \ldots, 365\}$	The day of the year, ranging from 1 to 365, representing the position of the day within the year.

Appendix B

Table A2. Performance comparison of baseline models and Multi-Scale TsMixer on CSI300 regression task for different input sequences.

	Model	Days	MAPE (↓)	MAE (↓)	R^2 (↑)
Baseline	TSMixer	20	0.8522	33.0073	0.9822
		60	0.8542	33.0702	0.9822
		80	0.8469	32.8142	0.9822
	Informer	20	0.8510	32.9610	0.9820
		60	0.8479	32.8465	0.9821
		80	0.8454	32.7390	0.9821
	DLinear	20	0.8522	**32.7635**	**0.9823**
		60	0.8542	32.9006	0.9823
		80	0.8469	32.8755	0.9824
	PatchTST	20	0.9334	36.0531	0.9805
		60	1.0275	39.6457	0.9774
		80	1.0189	39.2741	0.9775
	Transformer	20	0.8519	32.9921	0.9820
		60	0.8491	32.8979	0.9821
		80	0.8475	32.7897	0.9823

Table A2. *Cont.*

	Model	Days	MAPE (↓)	MAE (↓)	R^2 (↑)
Baseline	TimesNet	20	**0.8490**	32.8955	0.9821
		60	0.8465	32.7864	0.9822
		80	0.8464	32.7728	0.9823
	TimeMixer	20	0.8496	32.9093	0.9820
		60	0.8506	32.9579	0.9818
		80	0.8458	32.7553	0.9823
Multi-Scale TsMixer (ours)		20	0.8500	32.899	**0.9823**
		60	**0.8411**	**32.577**	**0.9824**
		80	**0.8304**	**32.024**	**0.9825**

Table A3. Performance comparison of baseline models and Multi-Scale TsMixer on SSE50 regression task for different input sequences.

	Model	Days	MAPE (↓)	MAE (↓)	R^2 (↑)
Baseline	TSMixer	20	0.8463	22.2496	0.9805
		60	0.8384	22.0446	0.9805
		80	0.8385	22.0575	**0.9806**
	Informer	20	0.8497	22.3443	0.9802
		60	0.8448	22.2235	0.9804
		80	0.8423	22.1587	0.9804
	DLinear	20	0.8371	22.0097	**0.9806**
		60	0.8357	21.9767	**0.9806**
		80	0.8368	22.0054	**0.9806**
	PatchTST	20	0.9314	24.4489	0.9781
		60	1.0374	27.1900	0.9747
		80	0.9647	25.3007	0.9768
	Transformer	20	0.8419	22.1472	0.9804
		60	0.8401	22.1039	0.9803
		80	0.8361	21.9948	**0.9806**
	TimesNet	20	0.8395	22.0832	0.9803
		60	**0.8337**	**21.9292**	0.9805
		80	0.8355	21.9880	0.9805
	TimeMixer	20	0.8463	22.2616	0.9804
		60	0.8439	22.1849	0.9805
		80	0.8382	22.0440	0.9805
Multi-Scale TsMixer (ours)		20	**0.8350**	**21.9561**	**0.9806**
		60	0.8380	22.0314	**0.9806**
		80	**0.8310**	**21.7311**	**0.9806**

Appendix C

Table A4. Performance comparison of baseline models and Multi-Scale TsMixer on CSI300 classification task for different input sequences.

	Model	Days	Precision (↑)	Specificity (↑)	Accuracy (↑)
Baselines	TSMixer	20	0.4871	0.5513	0.5283
		60	0.4912	0.5550	0.5324
		80	0.5083	0.5592	0.5409
	Informer	20	0.4839	0.5621	0.5262
		60	0.4882	0.5658	0.5298
		80	0.4905	0.5701	0.5323
	DLinear	20	0.4912	0.5705	0.5343
		60	0.5041	0.5759	0.5403
		80	0.5128	0.5783	0.5471
	PatchTST	20	0.4452	0.5284	0.4893
		60	0.4591	0.5406	0.5061
		80	0.4814	0.5579	0.5228
	Transformer	20	0.4878	0.5670	0.5281
		60	0.4903	0.5789	0.5345
		80	0.4987	0.5722	0.5372

Table A4. *Cont.*

	Model	Days	Precision (↑)	Specificity (↑)	Accuracy (↑)
Baselines	TimesNet	20	0.4941	**0.5723**	**0.5347**
		60	0.4961	0.5734	0.5389
		80	0.5082	0.5792	0.5451
	TimeMixer	20	**0.4943**	0.5602	0.5318
		60	0.5094	0.5727	0.5382
		80	0.5118	0.5753	0.5436
Multi-Scale TsMixer (ours)		20	**0.4956**	0.5664	0.5303
		60	**0.5152**	**0.5821**	**0.5501**
		80	**0.5219**	**0.5902**	**0.5548**

Table A5. Performance comparison of baseline models and Multi-Scale TsMixer on SSE50 classification task for different input sequences.

	Model	Days	Precision (↑)	Specificity (↑)	Accuracy (↑)
Baselines	TSMixer	20	0.4845	0.5487	0.5279
		60	0.4893	0.5532	0.5328
		80	0.5067	0.5586	0.5393
	Informer	20	0.4863	0.5612	0.5287
		60	0.4876	0.5645	0.5311
		80	0.4897	0.5687	0.5343
	DLinear	20	**0.4976**	0.5593	0.5312
		60	0.5027	0.5742	0.5394
		80	0.5104	0.5738	0.5462
	PatchTST	20	0.4435	0.5263	0.4882
		60	0.4576	0.5389	0.5053
		80	0.4808	0.5567	0.5225
	Transformer	20	0.4861	0.5668	0.5229
		60	0.4973	0.5715	0.5367
		80	0.4951	0.5772	0.5384
	TimesNet	20	**0.4949**	0.5659	0.5301
		60	**0.5147**	**0.5808**	**0.5483**
		80	0.5073	0.5781	0.5446
	TimeMixer	20	0.4905	**0.5699**	**0.5318**
		60	0.4934	0.5712	0.5335
		80	0.5087	0.5719	0.5373
Multi-Scale TsMixer (ours)		20	0.4924	**0.5721**	**0.5362**
		60	0.5116	0.5775	0.5428
		80	**0.5209**	**0.5895**	**0.5531**

References

1. Gorton, G.; Rouwenhorst, K.G. Facts and fantasies about commodity futures. *Financ. Anal. J.* **2006**, *62*, 47–68. [CrossRef]
2. Fang, W.; Zhang, S.; Xu, C. Improving prediction efficiency of Chinese stock index futures intraday price by VIX-Lasso-GRU Model. *Expert Syst. Appl.* **2024**, *238*, 121968. [CrossRef]
3. Hu, G.X.; Wang, J. A review of China's financial markets. *Annu. Rev. Financ. Econ.* **2022**, *14*, 465–507. [CrossRef]
4. Wang, X.; Wang, X.; Li, B.; Bai, Z. The nonlinear characteristics of Chinese stock index futures yield volatility: Based on the high frequency data of CSI300 stock index futures. *China Financ. Rev. Int.* **2020**, *10*, 175–196. [CrossRef]
5. Carpenter, J.N.; Lu, F.; Whitelaw, R.F. The real value of China's stock market. *J. Financ. Econ.* **2021**, *139*, 679–696. [CrossRef]
6. Chen, R.; Pan, B. Chinese stock index futures price fluctuation analysis and prediction based on complementary ensemble empirical mode decomposition. *Math. Probl. Eng.* **2016**, *2016*, 3791504. [CrossRef]
7. Li, Y.; Shen, D.; Wang, P.; Zhang, W. Does intraday time-series momentum exist in Chinese stock index futures market? *Financ. Res. Lett.* **2020**, *35*, 101292. [CrossRef]
8. Wang, C.P.; Lin, S.H.; Huang, H.H.; Wu, P.C. Using neural network for forecasting TXO price under different volatility models. *Expert Syst. Appl.* **2012**, *39*, 5025–5032. [CrossRef]
9. Sezer, O.B.; Gudelek, M.U.; Ozbayoglu, A.M. Financial time series forecasting with deep learning: A systematic literature review: 2005–2019. *Appl. Soft Comput.* **2020**, *90*, 106181. [CrossRef]

10. Zhou, F.; Zhang, Q.; Sornette, D.; Jiang, L. Cascading logistic regression onto gradient boosted decision trees for forecasting and trading stock indices. *Appl. Soft Comput.* **2019**, *84*, 105747. [CrossRef]
11. Asghar, M.Z.; Rahman, F.; Kundi, F.M.; Ahmad, S. Development of stock market trend prediction system using multiple regression. *Comput. Math. Organ. Theory* **2019**, *25*, 271–301. [CrossRef]
12. Junior, P.R.; Pamplona, E.D.O.; Salomon, F.L.R. ARIMA: An applied time series forecasting model for the Bovespa stock index. *Appl. Math.* **2014**, *5*, 3383. [CrossRef]
13. Wang, C. Forecast on price of agricultural futures in China based on ARIMA model. *Asian Agric. Res.* **2016**, *8*, 9–16.
14. Byun, S.J.; Cho, H. Forecasting carbon futures volatility using GARCH models with energy volatilities. *Energy Econ.* **2013**, *40*, 207–221. [CrossRef]
15. Arnerić, J.; Poklepović, T. Nonlinear extension of asymmetric garch model within neural network framework. *Comput. Sci. Inf. Technol.* **2016**, *6* , 101–111.
16. Lin, Z. Modelling and forecasting the stock market volatility of SSE Composite Index using GARCH models. *Future Gener. Comput. Syst.* **2018**, *79*, 960–972. [CrossRef]
17. Tang, Y.; Song, Z.; Zhu, Y.; Yuan, H.; Hou, M.; Ji, J.; Tang, C.; Li, J. A survey on machine learning models for financial time series forecasting. *Neurocomputing* **2022**, *512*, 363–380. [CrossRef]
18. Yan, H.; Ouyang, H. Financial time series prediction based on deep learning. *Wirel. Pers. Commun.* **2018**, *102*, 683–700. [CrossRef]
19. Cao, J.; Li, Z.; Li, J. Financial time series forecasting model based on CEEMDAN and LSTM. *Phys. A Stat. Mech. Its Appl.* **2019**, *519*, 127–139. [CrossRef]
20. Lazcano, A.; Herrera, P.J.; Monge, M. A combined model based on recurrent neural networks and graph convolutional networks for financial time series forecasting. *Mathematics* **2023**, *11*, 224. [CrossRef]
21. Olorunnimbe, K.; Viktor, H. Ensemble of temporal Transformers for financial time series. *J. Intell. Inf. Syst.* **2024**, *62*, 1087–1111. [CrossRef]
22. Huang, W.; Nakamori, Y.; Wang, S.Y. Forecasting stock market movement direction with support vector machine. *Comput. Oper. Res.* **2005**, *32*, 2513–2522. [CrossRef]
23. Illa, P.K.; Parvathala, B.; Sharma, A.K. Stock price prediction methodology using random forest algorithm and support vector machine. *Mater. Today Proc.* **2022**, *56*, 1776–1782. [CrossRef]
24. Yang, J.; Zhao, C.; Yu, H.; Chen, H. Use GBDT to predict the stock market. *Procedia Comput. Sci.* **2020**, *174*, 161–171. [CrossRef]
25. Lei, M.; Wang, X. DWT-TimesNet: An Improved Model for Wheat Price Long-Time Series Forecasting. In Proceedings of the 2023 IEEE International Conference on Big Data (BigData), Sorrento, Italy, 15–18 December 2023; pp. 3431–3438.
26. Waswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.; Kaiser, L.; Polosukhin, I. Attention is all you need. In Proceedings of the NIPS, Long Beach, CA, USA, 4–9 December 2017.
27. Zhou, H.; Zhang, S.; Peng, J.; Zhang, S.; Li, J.; Xiong, H.; Zhang, W. Informer: Beyond efficient transformer for long sequence time-series forecasting. In Proceedings of the AAAI Conference on Artificial Intelligence, Vancouver, BC, Canada, 2–9 February 2021; Volume 35, pp. 11106–11115.
28. Zeng, A.; Chen, M.; Zhang, L.; Xu, Q. Are transformers effective for time series forecasting? In Proceedings of the Thirty-Seventh AAAI Conference on Artificial Intelligence and Thirty-Fifth Conference on Innovative Applications of Artificial Intelligence and Thirteenth Symposium on Educational Advances in Artificial Intelligence, Washington, DC, USA, 7–14 February 2023; AAAI Press: Washington, DC, USA, 2023.
29. Nie, Y.; Nguyen, N.H.; Sinthong, P.; Kalagnanam, J. A Time Series is Worth 64 Words: Long-term Forecasting with Transformers. In Proceedings of the Eleventh International Conference on Learning Representations, Kigali, Rwanda, 1–5 May 2023.
30. Wang, S.; Wu, H.; Shi, X.; Hu, T.; Luo, H.; Ma, L.; Zhang, J.Y.; Zhou, J. TimeMixer: Decomposable Multiscale Mixing for Time Series Forecasting. In Proceedings of the Twelfth International Conference on Learning Representations, Vienna, Austria, 7–11 May 2024.
31. Wu, H.; Hu, T.; Liu, Y.; Zhou, H.; Wang, J.; Long, M. TimesNet: Temporal 2D-Variation Modeling for General Time Series Analysis. In Proceedings of the Eleventh International Conference on Learning Representations, Kigali, Rwanda, 1–5 May 2023.
32. Chen, S.A.; Li, C.L.; Arik, S.O.; Yoder, N.C.; Pfister, T. TSMixer: An All-MLP Architecture for Time Series Forecasting. *Trans. Mach. Learn. Res. arXiv* **2023**, arXiv:2303.06053.
33. Jiang, W. Applications of deep learning in stock market prediction: Recent progress. *Expert Syst. Appl.* **2021**, *184*, 115537. [CrossRef]
34. Bhandari, H.N.; Rimal, B.; Pokhrel, N.R.; Rimal, R.; Dahal, K.R.; Khatri, R.K. Predicting stock market index using LSTM. *Mach. Learn. Appl.* **2022**, *9*, 100320. [CrossRef]

35. Tolstikhin, I.O.; Houlsby, N.; Kolesnikov, A.; Beyer, L.; Zhai, X.; Unterthiner, T.; Yung, J.; Steiner, A.; Keysers, D.; Uszkoreit, J.; et al. Mlp-mixer: An all-mlp architecture for vision. *Adv. Neural Inf. Process. Syst.* **2021**, *34*, 24261–24272.
36. Wang, C.; Chen, Y.; Zhang, S.; Zhang, Q. Stock market index prediction using deep Transformer model. *Expert Syst. Appl.* **2022**, *208*, 118128. [CrossRef]