applied sciences

MDPI

Article

Multi-Step Multidimensional Statistical Arbitrage Prediction Using PSO Deep-ConvLSTM: An Enhanced Approach for Forecasting Price Spreads

Sensen Tu, Panke Qin *, Mingfu Zhu, Zeliang Zeng, Shenjie Cheng and Bo Ye

School of Software, Henan Polytechnic University, Jiaozuo 454003, China; 312120010520@home.hpu.edu.cn (S.T.); mingfuzhu@hpu.edu.cn (M.Z.); zengzeliang@home.hpu.edu.cn (Z.Z.); 212209020021@home.hpu.edu.cn (S.C.); 212309020094@home.hpu.edu.cn (B.Y.)

* Correspondence: qinpanke@hpu.edu.cn

Abstract: Due to its effectiveness as a risk-hedging trading strategy in financial markets, futures arbitrage is highly sought after by investors in turbulent market conditions. The essence of futures arbitrage lies in formulating strategies based on predictions of future futures price differentials. However, contemporary research predominantly focuses on projections of single indicators for the subsequent temporal juncture, and devising efficacious arbitrage strategies often necessitates the examination of multiple indicators across timeframes. To tackle the aforementioned challenge, our methodology leverages a PSO Deep-ConvLSTM network, which, through particle swarm optimization (PSO), refines hyperparameters, including layer architectures and learning rates, culminating in superior predictive performance. By analyzing temporal-spatial data within financial markets through ConvLSTM, the model captures intricate market patterns, performing better in forecasting than traditional models. Multistep forward simulation experiments and extensive ablation studies using future data from the Shanghai Futures Exchange in China validate the effectiveness of the integrated model. Compared with the gate recurrent unit (GRU), long short-term memory (LSTM), Transformer, and FEDformer, this model exhibits an average reduction of 39.8% in root mean squared error (RMSE), 42.5% in mean absolute error (MAE), 45.6% in mean absolute percentage error (MAPE), and an average increase of 1.96% in coefficient of determination (R2) values.

Keywords: convolutional long short-term memory model (CONVLSTM); particle swarm optimization (PSO); inter-commodity spread; multistep and multidimensional forecast

Citation: Tu, S.; Qin, P.; Zhu, M.; Zeng, Z.; Cheng, S.; Ye, B. Multi-Step Multidimensional Statistical Arbitrage Prediction Using PSO Deep-ConvLSTM: An Enhanced Approach for Forecasting Price Spreads. *Appl. Sci.* **2024**, *14*, 3798. https://doi.org/10.3390/app14093798

Academic Editor: Douglas O'Shaughnessy

Received: 25 March 2024
Revised: 23 April 2024
Accepted: 26 April 2024
Published: 29 April 2024

1. Introduction

In the dynamic arena of financial markets, futures arbitrage continues to be a pivotal subject for scrutiny among investors and scholars. This strategy is integral to capitalizing on profit avenues discerned through meticulous analysis and the tactical exchange of futures contracts, by capitalizing on the variances in pricing that occur in diverse markets over time [1]. To achieve successful arbitrage, it typically demands advanced forecasts of market trajectories and asset pricing volatilities. Price, which is a paramount factor in the ebbs and flows of trading activities, carries significant implications for the expansion of the market and the welfare of its participants. As a result, predicting price trends is an essential component of research in financial investment and a foundational element in devising strong arbitrage tactics.

Transitioning from the fundamental theory of futures arbitrage, early attempts to predict commodity futures prices were grounded in standard econometric methods. Historically, these methods were defined by the groundbreaking work of academics, such as the ARCH model introduced by Robert Engle in 1982, which explained time-varying volatility for economic time series data [2]. Following this innovation, a multitude of statistical models emerged, refining the approach to predict financial market behaviors. Notably, in

1986, Bollerslev presented the GARCH model, enhancing the ARCH concept and tailoring it more explicitly to financial datasets [3]. These econometric models, with the GARCH model in particular, were not only used in alignment with traditional regression techniques, but they also uniquely accounted for fluctuating error variance, facilitating a more nuanced understanding of market volatility—a key component in investor decision-making. A significant milestone was achieved with Morana's study in 2001, which successfully applied GARCH-based methods for short-term crude oil price forecasts [4]. Despite their empirical successes, challenges have persisted with these traditional models, primarily due to the intricacy and dynamic nature of financial data, which present nonlinear characteristics and time-sensitive elements that traditional mathematical models struggle to encapsulate fully.

The advent of machine learning techniques in the financial sector has ushered in a new era for algorithmic trading. Currently, researchers in related fields are primarily concentrating on two aspects: the extraction of Alpha factors and the optimization of synthetic models.

Factor mining, which is based on artificial intelligence algorithms, has evolved beyond the traditional approach of establishing clear investment logic relationships to mine and screen factors. Instead, it has become more adaptable, extracting and learning valuable information from relational events and sentiment data to enhance the efficiency of investment decisions. For instance, deep learning models can analyze unstructured data such as news articles and social media posts by natural language processing (NLP) technology. It allows models to capture market sentiment and potential influencing factors and enhance the precision of spread predictions. Recent literature corroborates that deep learning models excel at distilling crucial features from complex data and adeptly applying these insights to new contexts [5].

An accumulation of research underscores the efficacy of machine learning techniques in anticipating futures prices. Early initiatives, such as Grudnitski's 1993 study [6], employed neural network (NN) algorithms for predicting gold futures prices, showing a marked improvement over traditional time series approaches. The competitive edge of NNs was further highlighted by Moshiri and Foroutan's 2006 comparison [7], which found NNs to outperform ARMA and GARCH models in crude oil price forecasting. Extending this success, Diana Osorio et al., (2017) [8] applied neural networks to project S&P and gold futures prices, yielding promising outcomes. Hailei Zhao's research in 2021 [9] implemented machine learning techniques grounded on the fundamental factors of agricultural product futures, enhancing forecast precision and providing key contributions to the field. Tsantekidis et al. (2017) [10] chose convolutional neural networks for anticipating stock prices and discovered they surpassed traditional multilayer perceptrons and support vector machines in their predictive prowess. Dixon et al. implement a deep neural network (DNN) in 2016 [11] for various commodity futures prices and reported uniquely accurate results. In the vein of leveraging sophisticated AI, Long et al. (2018) [12] incorporated LSTM, BPNN, and CNN in creating arbitrage models, with experiments on coking coal, iron ore, and rebar futures demonstrating LSTM's superior performance. Sheng Y and Ma D's comparison in 2022 [13] among LASSO, Xgboost, BPNN, and LSTM for arbitrage effects underlined the outstanding forecasting abilities of deep learning models. Zhou et al., (2021) [14] proposed the Informer model based on the Transformer model and demonstrated the effectiveness of the model in enhancing the prediction capacity of long sequence time-series forecasting (LSTF) through experiments. Liu et al. (2022) [15] designed a SCINet model based on TCN, which achieved higher accuracy on public datasets across multiple fields. Moreover, neural network-based generative techniques have also begun to be applied in the field of financial time series. Zhang et al. [16] proposed a generative adversarial network (GAN) structure based on LSTM and got a promising performance in the closing price prediction on the real data. It is becoming increasingly clear that AI, especially through deep learning and machine learning, offers tremendous promise for futures market prediction and navigation. These advanced methodologies adeptly tame the nonlinearity of financial markets, elevate

forecasting accuracy to new heights, and furnish investors with sophisticated tools for crafting strategic decisions.

Nonetheless, current predictive endeavors within financial studies commonly focus on the immediate future, aiming to determine the upcoming value or direction for a single metric, essentially for univariate time series projection. Researchers often design models that are calibrated solely to forecast a single point or a unidimensional trend in subsequent time steps. This traditional approach, however, does not suffice for arbitrage strategy construction, where a broader temporal and dimensional perspective is essential for evaluating multiple indicators over an upcoming time span. To navigate these complexities, this study introduces a convolutional long short-term memory (ConvLSTM) model geared toward predicting futures price spreads. This advanced model boasts capabilities for multistep and multidimensional forecasting, which aligns more closely with the nuanced demands of practical trading scenarios.

However, network architectures like ConvLSTM pose challenges due to their complex decision-making process and limited transparency in how predictions are influenced [17], which can affect confidence and trust in their results. Additionally, choosing the right hyperparameters often depends on previous research or the subjective judgment of practitioners. Selecting the appropriate hyperparameters is essential for enhancing the network's structure, which improves its ability to generalize and fit data accurately. A significant scholarly effort is directed toward developing systematic methods to reduce subjective bias and identify the best set of hyperparameters.

This study presents an innovative forecasting approach tailored for the futures arbitrage domain, utilizing a PSO Deep-ConvLSTM model—a sophisticated, multidimensional, multistep predictor that seamlessly integrates PSO with the ConvLSTM network. The primary aim is to enhance the accuracy of futures price spread forecasts, thereby amplifying the potential for profitable arbitrage strategies. By harnessing real historical futures price spread data, the study endeavors to ascertain the efficacy of the PSO Deep-ConvLSTM through a comprehensive comparative analysis against alternative forecasting models.

In addressing the inherent challenges associated with the opaque decision-making of traditional ConvLSTM models, this research pioneers the utilization of the PSO algorithm for optimizing hyperparameters, with the overarching goal of enhancing the model's transparency and augmenting its performance. This systematic tuning process is designed to mitigate reliance on subjective expertise and historical precedents, thus transitioning toward an objective and replicable methodology capable of reliably guiding the network's learning process.

Coupling the ConvLSTM's aptitude for capturing complex data patterns with PSO's strengths in parameter optimization, the proposed PSO Deep-ConvLSTM model promises to be a powerful tool in futures arbitrage. By advancing this synergy of methodologies, the framework aims to provide actionable insights, decision support, and investment strategies for market participants. Findings from this research are expected to offer a strong endorsement for the practical application of the PSO Deep-ConvLSTM model in arbitrage decision-making and risk management, thereby illuminating a pathway toward more sophisticated and precise financial market analyses.

In short, we summarize the key contributions of this work as follows:

1. We employ the ConvLSTM model to prognosticate multiple pertinent indicators in the future timeline of the futures price spread data.
2. By introducing the PSO algorithm, we optimize the ConvLSTM network. This approach rectifies the shortfall related to the inaccurate acquisition of initial connection weights and hyperparameters intrinsic to the ConvLSTM model. Consequently, it fortifies the objectivity of hyperparameter selection and enables a more accurate prediction of futures price spread data.
3. To evaluate the predictive performance of our PSO Deep-ConvLSTM model, we conducted comparative experiments with existing models, such as the FEDformer. Our

research findings demonstrate that our model exhibits marginally superior accuracy compared with state-of-the-art approaches.

The remainder of this paper is structured to facilitate clear comprehension and logical flow. Section 2 delineates the problem and engages in an in-depth analysis of the dataset. It aims to articulate the objective function configured for our peculiar study needs and to confirm the empirical dataset's reliability and accessibility. In Section 3, we expound on both the COVLSTM and the innovative PSO Deep-ConvLSTM frameworks. Section 4 is devoted to a detailed experimental evaluation and analytical comparison between the PSO Deep-ConvLSTM and established benchmark models, focusing specifically on their applicability to forecasting inter-commodity spread. Section 5 concludes the paper and discusses future perspectives.

2. Problem Statement and Data Analysis

In this section, we initially offer a concise overview of the problem that our research seeks to address. Furthermore, we provide a clear and succinct description of the data through correlation analysis and Engle–Granger (EG) cointegration tests, thereby verifying its validity and applicability.

2.1. Problem Statement

For predicting futures markets, engaging in multistep forecasting across diverse dimensions presents a more significant practical impact than single-step prediction or forecasting within isolated dimensions. This multifaceted approach garners the interest of both financial practitioners and researchers. The primary objective of multistep forecasting is to analyze historical data and project values for forthcoming time periods. In contrast to single-step forecasting, multistep forecasting grapples with heightened uncertainty, which may precipitate a decline in the predictive model's effectiveness due to cumulative errors during the modeling process. In response to this challenge, we propose the PSO Deep-ConvLSTM model as a viable solution. To demonstrate and validate our approach, we selected futures contracts for rebar (RB) and hot-rolled coil (HC) listed on the Shanghai Futures Exchange to formulate a pair trading strategy. We then trained and backtested the predictive model using the fitted spread data derived from this arbitrage investment portfolio.

More specifically, our approach entails the use of actual price spread data via the ConvLSTM model to forecast and generate an array of price spread variations, such as closing price, opening price, and lowest and highest price spreads, over an ensuing period. The end goal of these forecasts is to inform and guide us in formulating an appropriate futures arbitrage strategy. Therefore, with the intention of elevating the returns of our pair trading strategy, our foremost objective is to enhance the prediction accuracy of the ConvLSTM model. This objectivity is primarily achieved through the optimization of the model's hyperparameters. Guided by our sophisticated enhanced heuristic algorithm, the hyperparameter search within the ConvLSTM model is treated as a black-box optimization task. The corresponding objective is defined as follows:

$$\underset{\Theta}{minimize} \sum_{t \in \Omega_{test}} \| y_t - model_{\Theta}(X_t) \|_F^2 \tag{1}$$

where Θ denotes the hyperparameter set of our model, Ω_{test} is the set of time stamps used for testing, F is the Frobenius norm, and $model_{\Theta}$ is the predictive model.

2.2. Data Structure

The Shanghai Futures Exchange proffers a snapshot-based order feed implemented via the CTP (Comprehensive Transaction Platform). The feed accumulates changes occurring within the preceding 500 milliseconds, encompassing multiple fields that encapsulate trade and order-book information. Specifically for our analysis, we capitalized on the price field to compute the spread between rebar and hot-rolled coil. The dataset utilized in this study was sourced from Choice Financial Software. Given the inherent noisy characteristics of

financial data, we transformed the 500-millisecond spread data, ranging from 21:01 on 15 July 2020, to 10:50 on 18 May 2022, into 1 min K-line data. This transformation effectively mitigated the impact of noise, thereby enhancing our model's capacity to capture the temporal dependencies inherent in the data. Additionally, among the extensive array of contracts listed annually, we narrowed our focus to the historical data derived from January, May, and October contracts.

Subsequently, all adjusted data points were integrated, taking into account the trading volume. This comprehensive process culminated in a comprehensive dataset consisting of 153,440 historical trading data points, spanning a period of 447 days. Each data point comprises the following features:

1. OPEN/HIGH/LOW/CLOSE: the first/highest/lowest/last value in 1-min spread data.
2. Exponential moving average (EMA):

$$EMA = Close * Weight + EMA_{pre} * (1 - Weight)$$

3. Difference (DIF): $DIF_i = EMA(CLOSE, 12) - EMA(CLOSE, 26)$
4. Differential exponential average (DEA): $DEA_i = EMA(DIF_i, 9)$
5. Moving average convergence and divergence (MACD):

$$MACD_i = 2 \times (DIF_i - DEA_i)$$

6. The price spread fluctuation.

2.3. Data Analysis

Commodity contracts for paired transactions often need to have a long-term and stable cointegration relationship, such as the combination of rebar and hot-rolled coil. This paper makes a prediction study based on the real price difference data of this combination. This section demonstrates the effectiveness of using data.

To ascertain the presence of a long-term stable cointegration relationship among the selected futures contracts, we utilized EViews10 software to conduct a cointegration analysis on the original price data.

A close examination of the contract time series plot shown in Figure 1 reveals that the closing price data for both RB and HC display comparable fluctuation patterns. This initial observation indicates a potential correlation between the price data of these two commodity futures. A comprehensive quantitative analysis of this correlation is furnished in Table 1. The correlation coefficients computed for the opening price, closing price, highest price, and lowest price collectively suggest a significant correlation between the two commodities.

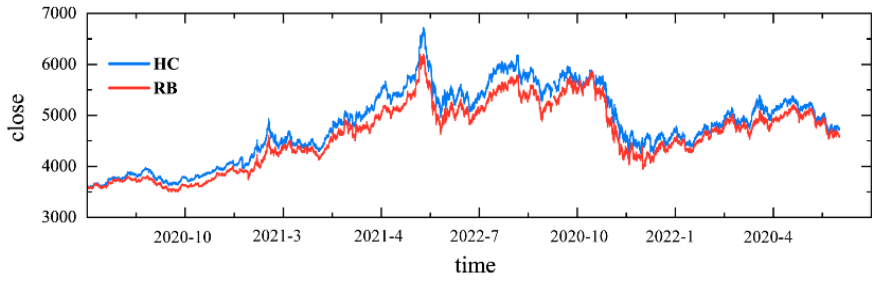

Figure 1. Time series plot of the closing prices for RB and HC.

Table 1. Correlation test of the time series.

CORR	Open	Close	High	Low
RB-HC	0.990999	0.991001	0.991005	0.990997

A correlation analysis of two time series, RB and HC, was conducted based on tick data and fitted into 1 min candlesticks where four price indicators are correspondingly analyzed for correlation.

While most variables within financial data are nonstationary, research into their correlations reveals intrinsic linkages and a stable equilibrium relationship over the long term. In this study, we performed stationarity tests on the selected RB and RH price series, employing the commonly used augmented Engle–Granger (ADF) method to test the stationarity of the time series. Tests were conducted for the series at 0th and 1st order unit root, with results presented in Table 2. As the data in the table elucidate, for the price series of RB and HC, we could not reject the null hypothesis at a 5% confidence level in the 0-order unit root test. Therefore, all variables were nonstationary. However, after differentiating the series at the 1st order, the absolute values of the t-statistics of each variable were greater than any critical value, with accompanying probabilities of zero, indicating each of them was stationary under a first-order differential. Consequently, it can be inferred that the price series of the principal futures contracts of both RB and HC are integrated into order one.

Table 2. ADF test on the price data of RB and HC.

Order	Series		Test Statistic	Critical Value			*p* Value	Stationarity (5%)
				1%	5%	10%		
Zero	RB	CLOSE	−1.889	−3.431	−2.862	−2.567	0.337	no
		OPEN	−1.872				0.345	no
		HIGH	−1.891				0.337	no
		LOW	−1.934				0.334	no
	HC	CLOSE	−1.927	−3.431	−2.862	−2.567	0.312	no
		OPEN	−1.933				0.317	no
		HIGH	−1.925				0.337	no
		LOW	−2.067				0.258	no
One	RB	CLOSE	−67.71	−3.431	−2.862	−2.567	0.000	yes
		OPEN	−227.87				0.000	yes
		HIGH	−45.67				0.000	yes
		LOW	−48.87				0.000	yes
	HC	CLOSE	−64.35	−3.431	−2.862	−2.567	0.000	yes
		OPEN	−148.11				0.000	yes
		HIGH	−55.819				0.000	yes
		LOW	−49.22				0.000	yes

Subsequently, we can initiate the Engle–Granger cointegration test, beginning with the formulation of the cointegration equation as follows:

$$hc\_close = c \cdot rb\_close + et \quad (2)$$

where et denotes the cointegration residual, or simply the residual. The parameter c represents the cointegration coefficient.

Table 3 presents the results of the Engle–Granger cointegration test. At the 1% confidence level, the ADF test statistic of the residual series is smaller than the critical value. As a result, we reject the null hypothesis and consider the series to be stationary. In accordance with Engle–Granger's cointegration theory, we deduce that the price data for the main contracts of RB and HC exhibit a cointegration relationship. Consequently, these data are suitable for pair trading.

Table 3. ADF test on the residual series.

Residual	ADF Test Statistic	Critical Value			*p*-Value	Conclusion
		1%	5%	10%		
et	−4.52	−3.431	−2.862	−2.567	0.0001	cointegration

To ensure the effectiveness of our fitting process, we conducted a stationarity test on the fitted spread data. Table 4 demonstrates that the fitted spread data are a stationary time series at a 5% confidence level, indicating that our fitting process is robust and reliable. The time series plot of the fitted data for the closing price spreads is displayed in Figure 2.

Table 4. ADF test on the fitted spread series.

Series	ADF Test Statistic	Critical Value			*p*-Value	Stationarity (5%)
		1%	5%	10%		
CLOSE	−4.609	−3.431	−2.862	−2.567	0.0001	yes
OPEN	−4.825				0.0000	
HIGH	−4.785				0.0001	
LOW	−4.607				0.0000	

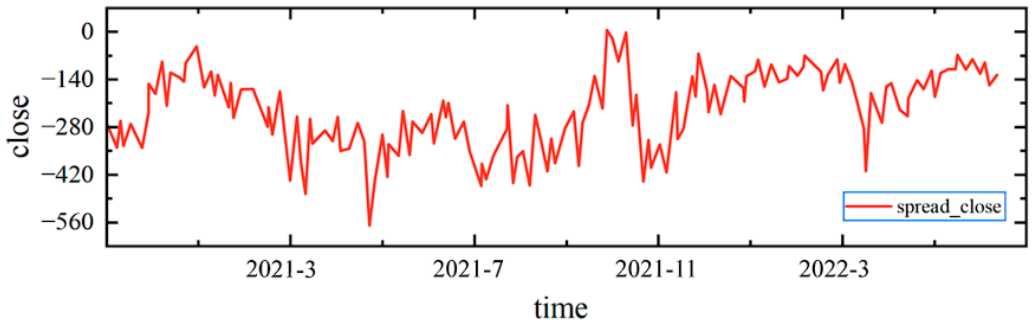

Figure 2. Time series plot of the closing prices difference.

3. Methodology

3.1. Convolutional Long Short-Term Memory (ConvLSTM)

Originally introduced in 2015 by Shi et al. [18], ConvLSTM is a neural network architecture that ingeniously combines convolutional neural networks (CNN) and long short-term memory networks (LSTM). This special blend of architectures allows ConvLSTM to leverage the benefaction of convolution operations in data feature extraction and characterize the recurrent information propagation inherent in LSTM.

In comparison with conventional LSTM networks (the structure of the ConvLSTM cell is shown in Figure 3), ConvLSTM initially employs convolution operations to extract data features before training to better predict numerical trends. ConvLSTM, fundamentally a recurrent neural network, uses the output of the previous recurrent unit as the input to the subsequent one and hosts three LSTM gating units. With the inclusion of convolution operations, ConvLSTM exhibits the ability to capture spatial variations during object motion. Consequently, ConvLSTM has found extensive applications across various spheres like video prediction [19], image compression [20], and other general algorithmic tasks [18].

The internal structure of ConvLSTM comprises a fully connected layer where the input, cell output, and state vectors are all one-dimensional (1D). The entire input data, cell outputs, hidden layers, and all three gating units of ConvLSTM, however, maintain a representation in 3D tensors. The state of a unit at a specific moment is determined by the convolution operator in the ConvLSTM network, with the convolution kernel size outlining the speed of the detected action. Contrasting with traditional fully connected LSTM, ConvLSTM effectively handles the tokenization of pure numerical data. ConvLSTM is better suited for handling the tokenization of raw numerical data. Through the convolutional operation between data and convolution kernels, kernel parameters are shared across input

data, enhancing the extraction of more generalized features. Equation (3) provides the mathematical representation of a single ConvLSTM network unit:

$$\begin{cases} i_t = \sigma(W_{xi} \cdot X_t + W_{hi} \cdot H_{t-1} + W_{ci} \odot C_{t-1} + b_i) \\ f_t = \sigma(W_{xf} \cdot X_t + W_{hf} \cdot H_{t-1} + W_{cf} \odot C_{t-1} + b_f) \\ C_t = f_t \odot C_{t-1} + i_t \odot \tan h(W_{xc} \cdot X_t + W_{hc} \cdot H_{t-1} + b_c) \\ o_t = \sigma(W_{xo} \cdot X_t + W_{ho} \cdot H_{t-1} + W_{co} \odot C_t + b_o) \\ H_t = o_t \odot \tan h(C_t) \end{cases} \tag{3}$$

where X_t represents the input data and hidden state at sequential time t, while f_t, i_t and o_t denote the forget gate, input gate, and output gate, respectively. The function is a sigmoid function defined as $\sigma(x) = 1/(1 + e^{-x})$. $\odot$ signifies an element-wise dot product. W and b represent weight matrices and bias vectors, respectively. The forget gate layer determines the extent to which previous information, gathered from H_{t-1} combined with X_t. If f_t equals 0, the model completely discards prior information; if f_t equals 1, the model retains all previous information. The input gate layer establishes which information to store in the cell state c_t, in conjunction with the c_t vector. Last, the output layer identifies which information to transmit, incorporating the cell state c_t and filtered input o_t in the process.

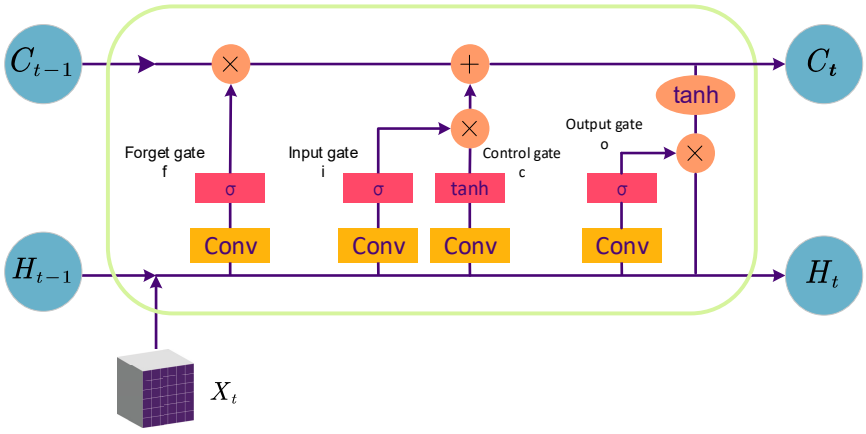

Figure 3. Structure of a ConvLSTM cell.

In application to futures spread prediction, the ConvLSTM network method considers the correlation of influencing factors. Samples can be perceived as "T images (n, K)" where T corresponds to the time step, n to the predicted step, and K to the number of selected influencing factors. This design allows ConvLSTM not only to handle time series autocorrelation effectively but also to consider the correlation between various factors. This holistic approach facilitates addressing multicollinearity problems in regression models, yielding more accurate futures spread predictions. Thus, through the application of ConvLSTM, incorporating both time series autocorrelation and inter-factor correlation into account, the accuracy of predictions is enhanced.

3.2. PSO Deep-ConvLSTM Network

Although ConvLSTM models have demonstrated their aptitude for handling time series data, determining optimal hyperparameters remains a formidable challenge. Undoubtedly, these hyperparameters substantially dictate the resulting network model's topology. Consequently, different hyperparameter configurations can lead to varying degrees of predictive performance. Therefore, the judicious selection of model hyperparameters is of paramount importance [21]. However, current research predominantly relies on subjective means, relying on intuition and trial-and-error methods for hyperparameter selection. Such an approach obscures the accuracy of determining the model's

optimal network structure hyperparameters, resulting in suboptimal prediction accuracy and inefficient training procedures.

To mitigate these challenges, this study introduces a pioneering integration of the particle swarm optimization (PSO) algorithm with ConvLSTM, culminating in the PSO Deep-ConvLSTM model. The model's primary goal is to facilitate an automated search for the optimal network structure hyperparameters of ConvLSTM by leveraging the PSO algorithm. Drawing inspiration from avian foraging behaviors, PSO stands as an intelligent optimization algorithm [22]. By simulating the collaborative interactions of birds, the algorithm steers a swarm toward their food source. This is achieved through iteratively updating their positions based on individual best positions and relative swarm positions, eventually forming an optimal configuration [23].

The PSO algorithm further boasts an array of appealing features, including ease of implementation, accelerated convergence rates, and the capacity to search for global optima. Therefore, the integration of the PSO algorithm enables swift, accurate searching for optimal ConvLSTM neural network structural hyperparameters. The proposed framework of the PSO Deep-ConvLSTM model is illustrated in Figure 4.

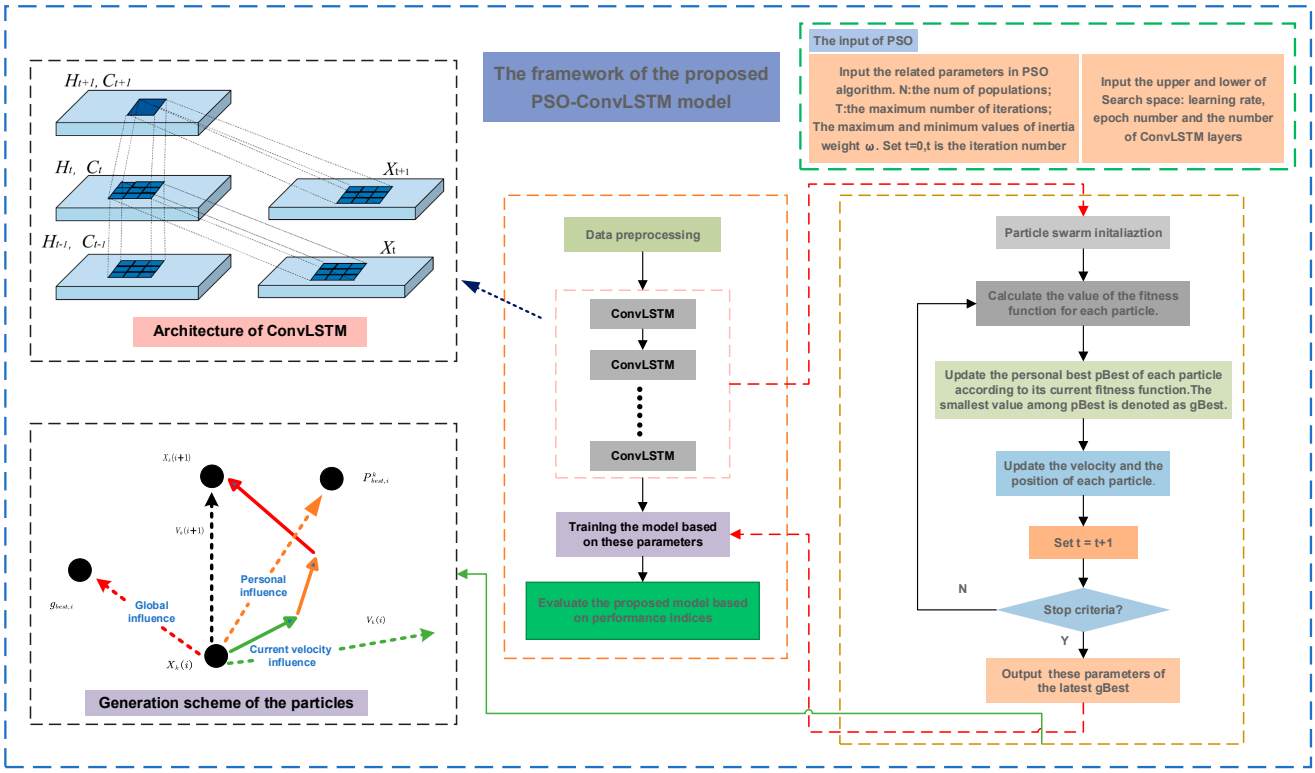

Figure 4. PSO Deep-ConvLSTM model architecture.

3.2.1. Dynamic Adjustment

In the PSO algorithm, for each particle k in the swarm, the velocity and position are updated at each $(i+1)^{th}$ iteration using the following Equation (4):

$$\begin{cases} V_k = \omega V_k(i+1) + c_1 r_1 (p^k_{best,i} - X_k(i)) + c_2 r_2 (g_{best,i} - X_k(i)) \\ X_k(i+1) = X_k(i) + V_k(i+1) \end{cases} \quad (4)$$

where $p^k_{best,i}$ is the individual best position of the particle k in the i^{th} iteration; $g_{best,i}$ is the global best position of any particle in the i^{th} iteration; ω is the inertia weight, which is a non-negative number and is used to adjust the search range of the solution space; c_1 and c_2 are the real acceleration coefficients that control how much the global and individual

best positions should influence the particle's velocity; r_1 and r_2 are uniformly distributed random numbers in the range of 0 and 1, used to maintain an adequate level of diversity. Algorithm 1 delineates the particle swarm optimization algorithm in pseudocode format.

Algorithm 1: The pseudocode of the algorithm is as follows

INPUT: Fitness function, lower bound, upper bound is part of the problem, i.e., it will be given in the problem. N (population size), and T (the maximum number of iterations) are to be chosen by the user. c_1, c_2 and ω update by the given equation.

1. Initialize a random population (P) and velocity (v) within the bounds;
2. Evaluate the objective function value (f) of P;
3. Assign p_{best} as p and f_{best} as f;
4. Identify the solution with the best fitness and assign that solution as g_{best} and fitness as f_{gbest}

For iteration:
for i = 1 to T:
for k = 1 to N:
Determine the velocity (v_k) of k^{th} particle;
Determine the new position (X_k) of k^{th} particle;
Bound X_k;
Evaluate the objective function value f_k of k^{th} particle;
Update the population by including X_k and f_k;
Update p^k_{best} and f_{pbest} if $f_k < f_{p^k_{best}}$ then $\begin{cases} p^k_{best} = X_k \\ f_{pbest} = f_k \end{cases}$
Update g_{best} and f_{gbest} if $f_{p^k_{best}} < f_{gbest}$ then $\begin{cases} g_{best} = p^k_{best} \\ f_{gbest} = f_{p^k_{best}} \end{cases}$
end
end

The inertia weight denoted as ω in the PSO algorithm, plays a crucial role in managing the inertia of particles and endorsing an expansive exploration of the search space. It functions as an integral parameter in striking a balance between the algorithm's global and local search capabilities. An established method of allocating inertia weights involves a linearly decreasing approach, offering ease of implementation. However, with a linear increase in iteration numbers, this strategy may subsequently shrink the inertia weight excessively, dampening the algorithm's global search capability and precipitating a potential entrapment in local optima. Moreover, as the algorithm advances into its late stages, the swarm's diversity is at risk of being diminished, reflecting a decelerated convergence speed [24].

In the PSO algorithm's purview, the learning factors c_1 and c_2 primarily mediate the step size adjustment for the particle's trajectory toward the individual's and flock's optimal positions. To hasten the search process during the preliminary iterations and enhance the global search capability, a decrease in c_1's value coupled with an increase in c_2's value is typically advocated, thereby fostering local refinement searches during the algorithm's later iterations. However, while present optimization research seldom espouses the optimization of learning factors, traditional PSO algorithms tend to fixate $c_1 = c_2$ at constant values, potentially failing to cater to pragmatic application needs.

Therefore, in light of the aforementioned rationales and extensive simulation experiments, this paper subsequently embraces the 'Dynamic Adjustment Strategy', as proposed by Huang Jianhua et al. [24]. This strategy dynamically modulates both the inertia weight and the learning factors, as modeled in the following implementation Equation (5):

$$\begin{cases} \omega = \omega_{\min} + [\omega_{\max} - \omega_{\min}] \cos\left(\frac{\pi t}{2T_{\max}}\right) + \sigma \times betarnd(a, b) \\ c_1 = 2\sqrt{1 - \left|\cos\left(\cos\left(\frac{\pi}{2} \times \frac{t}{T_{\max}}\right)\right)\right|} \\ c_2 = 2\sqrt{\left|\cos\left(\frac{\pi}{2} \times \frac{t}{T_{\max}}\right)\right|} \end{cases} \quad (5)$$

Among them $\omega_{\max}$ and $w_{\min}$ are the maximum and minimum values of inertia weights, respectively; $T_{\max}$ is the maximum number of iterations for particles; σ is the inertia adjustment factor; beta generates random numbers that follow a beta distribution where a and b are the two parameters of the beta distribution.

3.2.2. Structure of PSO Deep-ConvLSTM

The network architecture of PSO Deep-ConvLSTM consists of an input layer, multiple ConvLSTM layers, and an output layer. The specific number of ConvLSTM layers, as well as hyperparameters such as learning rate, epoch number, and convolution kernel size, serve as the objectives to be optimized through PSO. After defining the range of values for hyperparameters optimization and setting the parameters for the particle swarm, the positions and velocities of each particle are randomly initialized.

Within the learning model, both the loss function for ConvLSTM and the fitness function for PSO are selected as the "mean squared error (MSE)". The fitness values are computed and ranked, aiming to identify a set of hyperparameters that minimize the ConvLSTM error. The formula for the fitness function can be expressed as follows:

$$MSE = \frac{1}{n}\sum_{1}^{n}(\hat{y}_i - y_i)^2 \tag{6}$$

In this equation, the variable n represents the sample size of the validation dataset, y_i denotes the actual value, and $\hat{y}_i$ represents the predicted value.

The optimal value of the target parameter is output when the optimization process satisfies the convergence condition. Otherwise, it will repeat the process of computing the particle's fitness, updating the position and velocity of each particle, and updating the $p^k_{best,i}$ and $g_{best,i}$ until satisfying the convergence conditions.

3.2.3. Steps of the PSO Deep-ConvLSTM

The flowchart illustrating the process is depicted in Figure 5. The specific procedure for optimizing the parameters of the ConvLSTM network using PSO can be methodically outlined in six steps as follows:

Step 1: Preprocessing. The data preprocessing comprises several stages: conversion of the data into a format compatible with CONVLSTM, data normalization, sorting, and further normalization processing. Subsequently, we proportionally divide the data into the training and test sets.

Step 2: Parameters Initialization. Initial settings for particle swarm parameters, such as particle swarm size, the number of algorithm iterations, and the range of iteration steps, are established. Afterward, we randomly initialize the position and velocity of each particle.

Step 3: Fitness Value Calculation. MSE (Mean Square Error) serves as the fitness function in our method. By training the CONVLSTM neural network model, we can assess the fitness of each particle. Consequently, the personal historical best value (pbest) and the global best value (gbest) are deduced based on the current fitness values.

Step 4: Particle Speed and Position Update. Particles' velocities are dynamically adjusted according to the individual historical best position and the global best position of the population. The particle's position is influenced by both its previous position and the current particle velocity. In particular, the update of each particle's velocity and position can be executed as per Equations (4) and (5).

Step 5: Iteration Loop Evaluation. If the iteration is completed, the best value is outputted; otherwise, the process returns to Step 3.

Step 6: Prediction. The optimal hyperparameters are revealed and the model is trained based on these hyperparameters. Following this, the model's predictive capabilities are utilized to forecast future data.

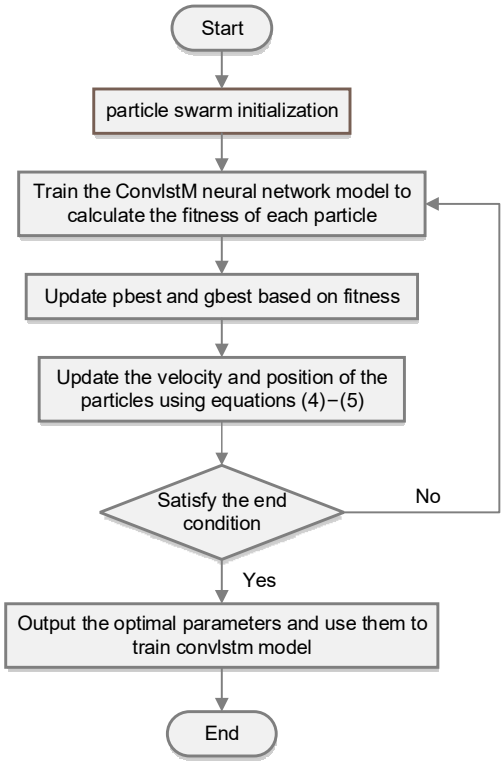

Figure 5. Flowchart of PSO Deep-ConvLSTM.

The PSO Deep-ConvLSTM model we finally built is based on the algorithm-optimized parameters, which determine the structure of the model and give a final decision-making ability to the model. The process of algorithmic optimization search demonstrates the origin of our parameter settings and model building while maximizing the elimination of human factors and enhancing the interpretability of the model structure and parameters.

4. Experiment

4.1. Experimental Environment

The hardware and software configurations used for this experiment are shown in Table 5. The network was built under the Pytorch deep learning framework, and training and testing of the network were conducted based on this framework.

Table 5. Experimental environment.

Items	Python Version	Memory	GPU	System	CUDA
Parameter	Python3.8	32GB	Tesla V100	Centos7	CUDA11.7

4.2. The Processing of Data

To verify the performance of the proposed prediction model, this paper adopts one-minute interval k-line fitting price spread data as the experimental data. Each term of the price spread data is composed of eight features, including the opening price spread, highest price spread, lowest price spread, closing price spread, MACD, DEA, DIF, and price spread fluctuation. In the experiment, we allocate 70% of the dataset as the training set for the comparative model, with the remaining 30% serving as the test set. For the PSO Deep-ConvLSTM model, the first 70% of the dataset is utilized for the optimization algorithm to search for optimal parameters and train the model, with the final 30% deployed as the test set to assess the model's generalization error. Herein, the data are processed as follows: Prior to feeding the feature data into the artificial neural network, the data

undergo normalization, being effectively converted into a [0, 1] range. This approach not only minimizes the impact of noise, enhancing the predictive accuracy of the model, but also expedites model convergence, ensuring efficient parameter updates within the neural network. Equation (7) provides the formula for normalization:

$$\overline{x}_i = \frac{x_i - x_{\min}}{x_{\max} - x_{\min}} \tag{7}$$

where $x_{\min}$ and $x_{\max}$, respectively, represent the minimum and maximum values of the entire training set.

As the data are subject to normalization during the model training phase, the output of the test set can be reverse normalized using Equation (8).

$$x_i = x_t' \cdot (x_{\max} - x_{\min}) + x_{\min} \tag{8}$$

where x_t' is the output value of the forecasting model.

4.3. Judgement Criteria

To assess the performance of the prediction model, this study employs the following four metrics to measure the model's predictive accuracy: mean absolute percentage error (MAPE), which takes into account the error between predicted and actual values as well as the proportionality of the error to the actual values; root-mean-square error (RMSE), a measure of the discrepancy between observation values and actual values; mean absolute error (MAE), an indicator reflecting the real error in predicted values; and the coefficient of determination (R^2), a gauge of the predictive power of a statistical model. The respective formulas are as follows:

$$\begin{cases} e_{MAPE} = \frac{1}{n}\sum_{1}^{n}\frac{|\hat{y}_i - y_i|}{y_i} \\ e_{RMSE} = \sqrt{\frac{1}{n}\sum_{1}^{n}(\hat{y}_i - y_i)^2} \\ e_{MAE} = \frac{1}{n}\sum_{1}^{n}|\hat{y}_i - y_i| \\ R^2 = \frac{\sum_{1}^{n}(\hat{y}_i - \overline{y})^2}{\sum_{1}^{n}(y_i - \overline{y})^2} \end{cases} \tag{9}$$

In these equations, y_i and $\hat{y}_i$ denote the actual and predicted values of the price spread data at time I, respectively; n stands for the sample size of the test dataset; and $\overline{y}$ represents the average value of the dataset. Typically, the smaller the values of MAPE, RMSE, and MSE, the less the deviation between the predicted and actual values. The value of R^2, which falls within the range [0, 1], is utilized to measure the accuracy of predictions by a statistical model. Ordinarily, a higher R-squared implies superior modeling performance.

4.4. Optimizing Network Parameters by the PSO

In the pursuit of effectively updating the weight of the neural network, this study has employed the Adam optimizer for the optimization of model parameters, with a batch size established at 128. The ConvLSTM model's compatibility with our dataset was assured via the implementation of the particle swarm optimization (PSO) algorithm for ConvLSTM hyperparameter optimization.

The PSO algorithm, by mimicking the communal behaviors exhibited by birds when locating food, leverages a combination of individual and collective experiences to progressively approach the target of interest. Continuous position updates, shaped by their personal optimum position in tandem with the overall flock's optimum position, consequently get

folded into an optimal configuration [25]. PSO operates in iterations, allowing for swarm updates through alterations in each component's velocity and position in every cycle. These updates fundamentally depend on personal best values (pbest) and global best values (gbest). Therefore, the accurate tuning of the PSO parameters is of utmost importance.

Eberhart and Shi's research [22] posits that a decent solution success rate is achievable when a PSO algorithm deploys a particle range of 20 to 50. However, our experiments have illustrated that employing a particle size of 6 allows for quicker convergence of the PSO algorithm with fewer computational resource requirements. A comprehensive review of PSO-centric literature [26–29] guided us in setting the particle size at 6 and iteration count at 10, with the upper and lower bounds of W being determined as 0.9 and 0.4, respectively.

Furthermore, an examination of academic literature surrounding the application of the ConvLSTM network in prediction problems [30,31], as well as relevant works, highlighted the pivotal role of the number of ConvLSTM layers and the size of the convolution kernel corroborated by the results of our experiments. Consequently, we selected the learning rate, size of the convolution kernel, the number of ConvLSTM layers, and the epoch as targets for optimization. A logical search range is essential to preventing issues such as excessive resource consumption associated with expansive search ranges during the search process. Following the analysis of related research, definitions for the search ranges were formulated as being [0.00001, 0.0005] for the learning rate, [1, 9] for convolution kernel size, and [2, 7] for layers of ConvLSTM. Besides, a few random epoch values were tested while keeping other parameters constant, revealing a suboptimal model performance when the epoch is less than 100 and an improved performance when it exceeds 300. However, to account for potential randomness, the epoch range was set between [1, 400].

With the search range of the target optimization parameters, we will obtain the optimal parameters by the PSO. As delineated in Table 6, the optimal parameters for the ConvLSTM model, as yielded by application of the PSO, stand at a learning rate of $5.7399496072129 \times 10^5$ an epoch count of 365, ConvLSTM layer featuring 6 neurons, and kernel-size of 6. Figure 6 shows the evolution of the loss function of the PSO Deep-ConvLSTM model during training and testing. Utilizing this optimal parameter configuration should potentialize the performance of the ConvLSTM model, particularly in the domain of arbitrage spread prediction.

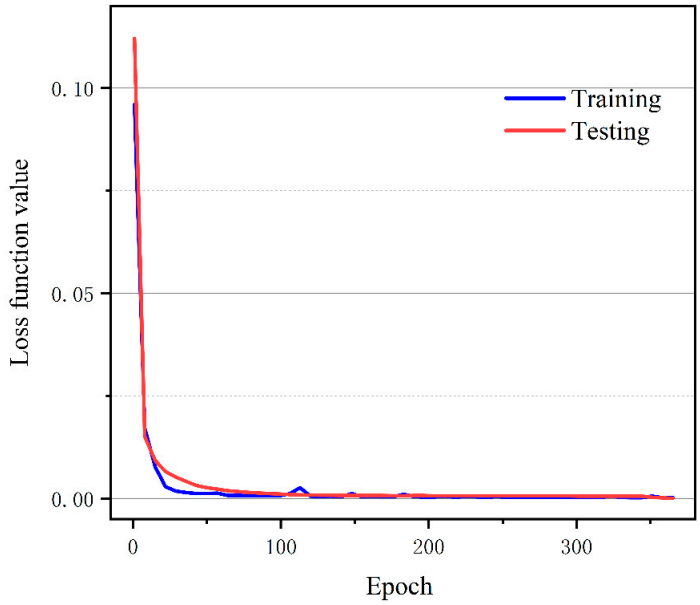

Figure 6. Evolution of loss function in PSO Deep-ConvLSTM model during training and testing.

Table 6. Results of hyperparametric optimization.

Parameter	Search Range	Optimal Value
Learning rate	[0.00001, 0.0005]	$5.7399496072129 \times 10^5$
Epoch	[1, 400]	365
ConvLSTM layers numbers	[2, 7]	6
Kernel size	[1, 9]	3

4.5. Experimental Results and Analysis

In this section, as we employ our proposed model to conduct multistep, multidimensional predictions on inter-commodity futures price spread data, it is imperative to validate the forecasting effectiveness of different models. According to the study by Kline et al. [32], multistep predictions can be realized through either iterative or independent methods.

1. The iterative method uses a single step-ahead model to iteratively generate forecasts.
2. The independent method uses a dedicated network to forecast each forecast horizon.

For comparative experimentation purposes, we have selected the following models: iterative LSTM, iterative GRU, Transformer, and FEDformer. The first two models belong to the iterative category, while our employed ConvLSTM, Transformer, and FEDtransformer adopt the independent prediction approach.

Specifically, recurrent neural networks (RNNs) have been appropriated in the realm of financial forecasting due to their distinct advantages in handling time series problems. However, they do present certain drawbacks, particularly the issue of gradient explosion. LSTM networks, capable of remedying the setbacks of RNNs, offer enhanced feature extraction and generalization abilities. The GRU model is another variant of the LSTM model. We have made minor modifications to these models enabling them to perform iterative predictions, thereby accomplishing multistep predictions.

Additionally, the Transformer [33] is an encoder-decoder structure primarily encompassing position encoding, position embedding, and a self-attention module. The multihead attention feature in the Transformer allows for parallel computation, thereby reducing the training duration. Since its introduction in 2017, the Transformer has been widely applied to various data types, including text and image data. The FEDformer is a variant of the Transformer model [34] that combines the Transformer with the seasonal-trend decomposition method and uses randomly selected Fourier components to maintain a compact representation of time series. This not only overcomes the Transformer's high computation cost and inability to capture a global view of time series, but it also further enhances prediction accuracy. Thus, we have also selected these two models for comparison. By evaluating our proposed model against these models, we can ascertain its performance and effectiveness in predicting cross-commodity futures price spreads, supplying valuable references for further research and application.

In the experiments, the batch size for all models was set to 128. For LSTM and GRU models, a two-layer structure was adopted with a learning rate of 0.0001 over 100 training epochs. The number of neurons was set to 100 and 20 for the first and second layers, respectively, while the Transformer and FEDformer models utilized the official default parameters. To comprehensively evaluate the performance of our proposed model and validate its superiority under different prediction horizons, we selected prediction settings of one, four, and eight steps. Specifically, the one-step prediction was primarily employed for benchmarking against LSTM-based models, highlighting the competitive edge of our proposed model in single-step forecasting. On the other hand, the four-step and eight-step predictions were utilized to assess the model's refinement capabilities in handling multistep forecasting tasks. For ease of reference, the models were assigned numerical identifiers: the GRU model was Model-1, the LSTM was Model-2, the Transformer was Model-3, FEDforemer-W was Model-4, FEDformer-F was Model-5, and our proposed PSO Deep-ConvLSTM was Model-6.

Table 7 provides a detailed characterization of each model's performance indicators for one-step, four-step, and eight-step predictions across the "open", "high", "low", and "close" dimensions. To clearly observe that the PSO Deep-ConvLSTM model can approximately predict the trend, we have plotted a comparison between the model's eight-step predictions for the "close" dimension and the actual values (as shown in Figure 6). Additionally, Figure 7 illustrates the prediction errors of the PSO Deep-ConvLSTM model compared with the actual values, providing further insight into its predictive accuracy.

Table 7. Evaluation results of the model for one-step, four-step, and eight-step predictions across four dimensions.

Model	Metrics	OPEN			HIGH			LOW			CLOSE		
		One	Four	Eight	One	Four	Eight	One	Four	Eight	One	Four	Eight
Model-1	RMSE	2.508	10.420	19.331	2.778	8.927	20.148	2.623	8.708	20.194	2.832	9.898	18.960
	MAE	2.104	8.483	15.409	2.296	7.190	16.187	2.367	6.934	16.273	2.316	7.999	15.086
	MAPE	1.782	6.195	12.565	1.942	5.389	13.567	2.187	4.912	12.992	2.268	5.817	12.351
	R^2	0.994	0.950	0.873	0.991	0.964	0.861	0.993	0.966	0.862	0.992	0.955	0.878
Model-2	RMSE	2.334	11.211	13.401	2.800	11.270	13.657	2.506	11.739	14.345	2.260	11.126	13.980
	MAE	1.904	8.762	10.113	2.141	8.781	10.469	2.186	9.268	11.062	1.871	8.736	10.782
	MAPE	1.642	6.279	7.706	1.439	6.402	8.223	1.843	6.511	8.367	1.700	6.261	8.275
	R^2	0.995	0.941	0.939	0.994	0.940	0.936	0.992	0.935	0.930	0.993	0.942	0.933
Model-3	RMSE	4.833	3.892	4.500	4.513	3.792	4.721	5.582	4.907	4.529	5.013	4.359	4.394
	MAE	4.204	2.825	3.277	3.811	2.641	3.496	4.910	3.881	3.279	4.292	3.275	3.136
	MAPE	3.397	2.099	2.439	3.069	1.897	2.692	3.928	3.002	2.387	3.390	2.488	2.266
	R^2	0.992	0.995	0.992	0.993	0.995	0.992	0.990	0.992	0.992	0.992	0.994	0.993
Model-4	RMSE	4.613	4.829	4.583	6.613	5.036	4.713	5.715	4.304	4.532	6.462	4.225	4.512
	MAE	3.738	3.691	3.251	5.359	3.883	3.369	4.556	3.090	3.058	5.251	3.123	3.192
	MAPE	2.796	2.647	2.270	4.143	2.825	2.409	3.347	2.152	2.094	3.848	2.224	2.228
	R^2	0.993	0.992	0.993	0.985	0.991	0.992	0.989	0.994	0.993	0.986	0.994	0.993
Model-5	RMSE	2.534	3.974	4.321	2.874	3.828	4.310	2.810	3.792	4.410	2.808	3.833	4.455
	MAE	1.751	2.763	3.008	1.978	2.606	2.944	1.893	2.617	3.026	1.989	2.645	3.038
	MAPE	1.245	1.963	2.094	1.438	1.854	2.085	1.312	1.793	2.071	1.410	1.844	2.137
	R^2	0.998	0.995	0.994	0.997	0.995	0.994	0.997	0.995	0.993	0.997	0.995	0.993
Model-6	RMSE	1.437	2.387	4.296	1.598	2.569	4.542	1.691	2.591	4.643	1.307	2.767	4.598
	MAE	1.215	1.633	3.103	1.240	1.830	3.304	1.359	1.782	3.411	0.922	1.953	3.353
	MAPE	0.910	1.162	2.218	0.949	1.311	2.447	1.006	1.258	2.405	0.669	1.411	2.405
	R^2	0.999	0.998	0.994	0.999	0.998	0.993	0.999	0.998	0.993	0.999	0.997	0.993

The GRU model was Model-1, the LSTM was Model-2, the Transformer was Model-3, FEDforemer-W was Model-4, FEDformer-F was Model-5, and our proposed PSO Deep-ConvLSTM was Model-6.

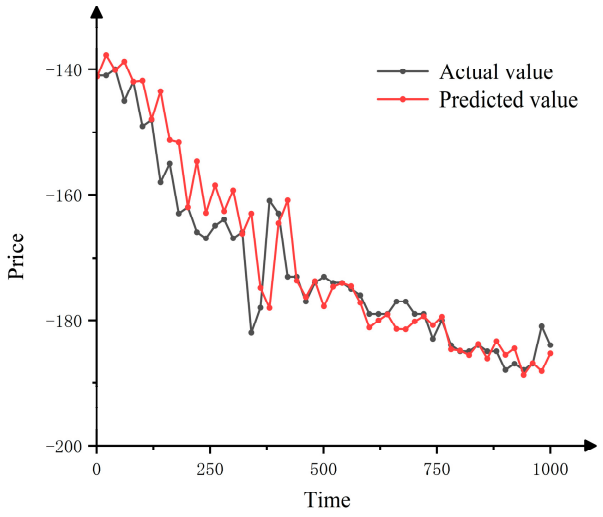

Figure 7. Comparison of predictions with real values.

From these data and figures, we can obtain an approximate understanding of the relative merits of each model. To enable a more intuitive and precise comparison of predictive effectiveness among the models and to underscore the superior performance of the PSO Deep-ConvLSTM model, we compiled the average of these indicators across all dimensions.

Figure 6 clearly illustrates that, for one-step-ahead predictions, the LSTM model has a significant advantage over other models, such as the Transformer. The PSO Deep-ConvLSTM model yields RMSE, MAE, and MAPE values of 1.508, 1.184, and 0.884, respectively. Compared with the traditional GRU and LSTM models, our proposed model has achieved a reduction of 43.8% and 39.1% in RMSE, 47.9% and 41.6% in MAE, and 56.8% and 46.6% in MAPE, demonstrating its superior prediction performance in single-step forecasting. From Figure 7, we observe that in four-step-ahead predictions, the LSTM and GRU models start to falter, while the Transformer-type models begin to show their strengths. The performance of LSTM and GRU deteriorates significantly, whereas the proposed model attains RMSE, MAE, and MAPE values of 2.579, 1.800, and 1.286, still delivering satisfactory results. When compared with advanced Transformer and FEDformer models, the PSO Deep-ConvLSTM model exhibits a decrease in RMSE by 39.1%, 43.9%, and 33.1%; in MAE by 43.0%, 47.8%; and 32.3%, and in MAPE by 45.8%, 47.8%, and 31.0%. This implies that the discrepancy between predicted and actual values for the four-step-ahead forecast is smaller with higher prediction accuracy. Furthermore, the R2 value is closer to 1, indicating a strong fitting capability. In Figure 8, in the case of eight-step-ahead predictions, the performance of LSTM and GRU models is quite poor, which also justifies their advantage in short-term forecasting. As the number of prediction steps increases, the error metrics exhibit substantial deterioration. However, the proposed model, with RMSE, MAE, and MAPE values of 4.520, 3.293, and 2.369, can alleviate this issue to some extent. In comparison with the long-term forecasting-focused Transformer model, our model reduces the various metrics by 0.4%, 0.1%, and 3.1%, respectively. Even when compared with the FEDformer model, an expert in long-term forecasting, our model surpasses it under the wavelet exchange mode and remains fairly competitive under the Fourier exchange mode. Figures 9–11 display the average evaluation indicators for single-step and multiple-step predictions of the six models.

In summary, the PSO Deep-ConvLSTM model achieves satisfactory prediction results in both single-step and multistep forecasting.

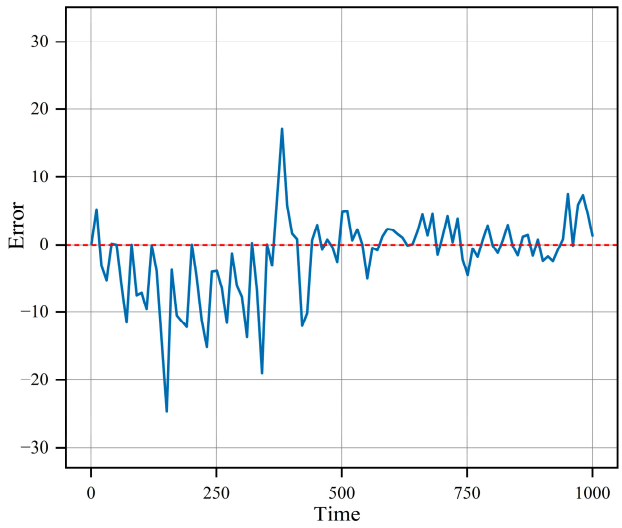

Figure 8. Prediction errors.

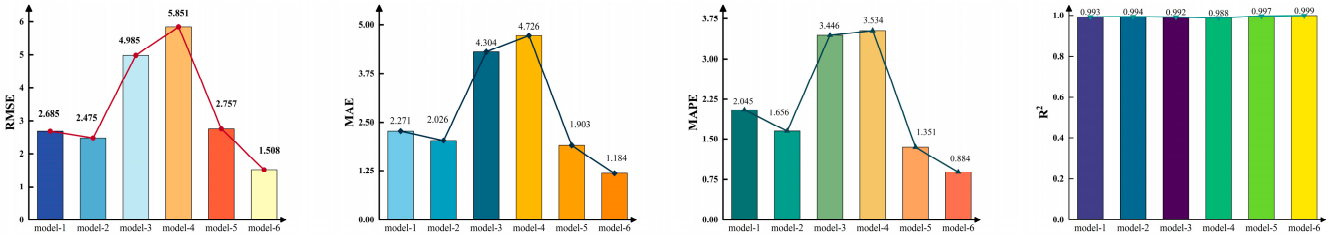

Figure 9. Forecasting performance of various models for one-step-ahead prediction of indicator values.

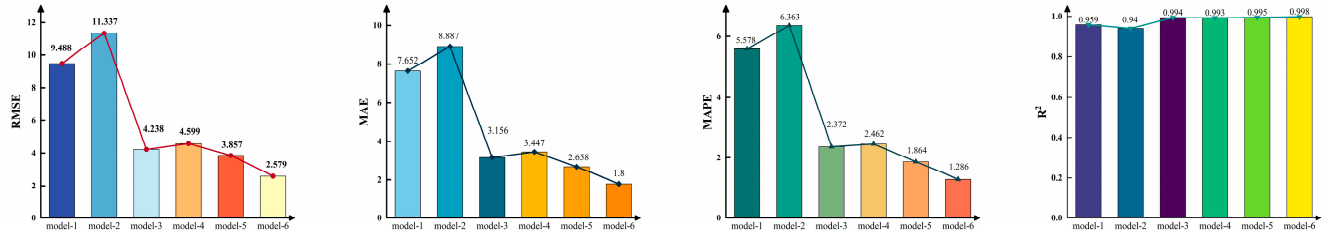

Figure 10. Forecasting performance of various models for four-step-ahead prediction of indicator values.

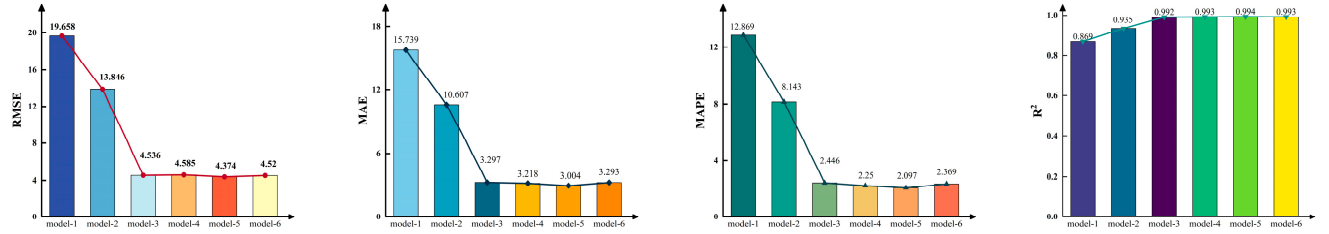

Figure 11. Forecasting performance of various models for eight-step-ahead prediction of indicator values.

5. Conclusions and Future Perspectives

In this study, we propose the PSO Deep-ConvLSTM futures price spreads prediction model, aiming to offer a superior predictive tool to aid investors in devising efficient arbitrage strategies within futures markets. The model's efficacy was tested using real historical futures data and assessed by juxtaposing the data with alternate time series models. The empirical observations revealed the PSO Deep-ConvLSTM model's distinct precision and superiority in both single-step and multistep predictions for futures price spread forecasting, particularly its augmented capacity to apprehend the nonlinear attributes embedded in the data. Additionally, we employed multiple data sets to further enhance the model's confidence stability and generalization ability. The model exhibited comprehensive analytical competencies, considering multiple market indicators, thereby affirming its reliability in handling the complexities and volatility inherent in futures markets.

The model's supremacy is reflected in two primary facets. Initially, the application of PSO methodology enhances the weight initialization and precision of parameters within the ConvLSTM network model, thus augmenting the objectivity of parameter selection. Moreover, its multidimensional and multistep predictive capacities pave the way for constructing arbitrage strategies based on forecast accuracy and comprehensiveness, elements of paramount importance to financial market participants.

To conclude, while the PSO Deep-ConvLSTM model has demonstrated significant performance potential, opportunities for amplification remain. These can include the integration of strategic statistical combinations and the addition of more futures market traits. Additionally, extending the model's application to other financial markets might further quantify its generalizability and adaptability. Anticipated future research may build upon this groundwork, proffering an expanded range of applications within the financial sector, thus presenting an array of profound possibilities.

Author Contributions: Conceptualization, S.T.; methodology, S.T.; software, S.T., Z.Z. and S.C.; validation, S.T., P.Q. and B.Y.; formal analysis, P.Q., investigation, S.T. and M.Z.; resources, S.T.; data curation, S.T.; writing—original draft preparation, S.T.; writing—review and editing, S.T.; visualization, S.T.; supervision, S.T. and P.Q.; project administration, S.T. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The original contributions presented in the study are included in the article; further inquiries can be directed to the corresponding author.

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

References

1. Girma, P.B.; Paulson, A.S. Risk arbitrage opportunities in petroleum futures spreads. *J. Futures Mark.* **1999**, *19*, 931–955. [CrossRef]
2. Engle, R.F. Autoregressive conditional heteroscedasticity with estimates of the variance of United Kingdom inflation. *Econom. J. Econom. Soc.* **1982**, *50*, 987–1007. [CrossRef]
3. Bollerslev, T. Generalized autoregressive conditional heteroskedasticity. *J. Econom.* **1986**, *31*, 307–327. [CrossRef]
4. Morana, C. A Semi-parametric Approach to Short-term Oil Price Forecasting. *Energy Econ.* **2001**, *23*, 325–338. [CrossRef]
5. Hsieh, T.J.; Hsiao, H.F.; Yeh, W.C. Forecasting stock markets using wavelet transforms and recurrent neural net-works: An integrated system based on artificial bee colony algorithm. *Appl. Soft Comput.* **2011**, *11*, 2510–2525. [CrossRef]
6. Grudnitski, G.; Osburn, L. Forecasting S&P and gold futures prices: An application of neural networks. *J. Futures Mark.* **1993**, *13*, 631–643.
7. Moshiri, S.; Foroutan, F. Forecasting nonlinear crude oil futures prices. *Energy J.* **2006**, *27*, 81–96. [CrossRef]
8. Osorio, D.; Giron, L.; Sierra, L. ¿Es El Mercado De Metales Eficiente? (Is the Metals Market Efficient?). Available online: https://ssrn.com/abstract=2956657 (accessed on 10 April 2017).
9. Zhao, H. Futures price prediction of agricultural products based on machine learning. *Neural Comput. Appl.* **2021**, *33*, 837–850. [CrossRef]
10. Tsantekidis, A.; Passalis, N.; Tefas, A.; Kanniainen, J.; Gabbouj, M.; Iosifidis, A. Forecasting Stock Prices from the Limit Order Book Using Convolutional Neural Networks. In Proceedings of the 2017 IEEE 19th Conference on Business Informatics (CBI), Thessaloniki, Greece, 24–27 July 2017.
11. Dixon, M.; Klabjan, D.; Bang, J.H. *Classification-Based Financial Markets Prediction Using Deep Neural Networks*; Social Science Electronic Publishing: Rochester, NY, USA, 2016.
12. Long, Z.; Lu, Y.; Ma, X.; Dong, B. Pde-net: Learning pdes from data. In Proceedings of the International Conference on Machine Learning (PMLR), Stockholm, Sweden, 10–15 July 2018; pp. 3208–3216.
13. Shen, Y.; Li, W.; Zeng, Y.; Li, Z.; Chen, Y.; Zhang, J.; Zhao, H.; Feng, L.; Ma, D.; Mo, X.; et al. Chromosome-level and haplotype-resolved genome provides insight into the tetraploid hybrid origin of patchouli. *Nat. Commun.* **2022**, *13*, 3511. [CrossRef] [PubMed]
14. 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, Virtually, 2–9 February 2021; Volume 35, pp. 11106–11115.
15. Liu, M.; Zeng, A.; Chen, M.; Xu, Z.; Lai, Q.; Ma, L.; Xu, Q. Scinet: Time series modeling and forecasting with sample convolution and interaction. *Adv. Neural Inf. Process. Syst.* **2022**, *35*, 5816–5828.
16. Zhang, K.; Zhong, G.; Dong, J.; Wang, S.; Wang, Y. Stock Market Prediction Based on Generative Adversarial Network. *Procedia Comput. Sci.* **2019**, *147*, 400–406. [CrossRef]
17. Weigend, A.; Gershenfeld, N. *Times Series Prediction: Forecasting the Future and Understanding the Past*; Addison-Wesley: Reading, MA, USA, 1994.
18. Shi, X.; Chen, Z.; Wang, H.; Yeung, D.Y.; Wong, W.; Woo, W. Convolutional LSTM Network: A Machine Learning Approach for Precipitation Nowcasting. *Adv. Neural Inf. Process. Syst.* **2015**, *28*, 802–810.
19. Wu, X.; Wu, Z.; Zhang, J.; Ju, L.; Wang, S. Salsac: A video saliency prediction model with shuffled attentions and correlation-based convlstm. In Proceedings of the AAAI Conference on Artificial Intelligence, New York, NY, USA, 7–12 February 2020; Volume 34, pp. 12410–12417.
20. Guan, S.; Li, H.; Zheng, W.S. Unsupervised learning for optical flow estimation using pyramid convolution lstm. In Proceedings of the 2019 IEEE International Conference on Multimedia and Expo (ICME), Shanghai, China, 8–12 July 2019; pp. 181–186.
21. Peiguang, L.; Jiaqian, Z.; Yulian, W. SCONV: A Financial Market Trend Forecast Method Based on Emotional Analysis. *J. Comput. Res. Dev.* **2020**, *57*, 1769–1778.

22. Kennedy, J.; Eberhart, R. Particle swarm optimization. In Proceedings of the ICNN'95-International Conference on Neural Networks, Perth, WA, Australia, 27 November–1 December 1995; Volume 4, pp. 1942–1948.
23. Jain, M.; Saihjpal, V.; Singh, N.; Singh, S.B. An overview of variants and advancements of PSO algorithm. *Appl. Sci.* **2022**, *12*, 8392. [CrossRef]
24. Jianhua, H.; Min, Z.; Qingchun, H.U. LSTM Stock Prediction Model Based on Improved Particle Swarm Optimization. *J. East China Univ. Sci. Technol.* **2022**, *48*, 696–707.
25. Kennedy, J. Swarm intelligence. In *Handbook of Nature-Inspired and Innovative Computing: Integrating Classical Models with Emerging Technologies*; Springer: Boston, MA, USA, 2006; pp. 187–219.
26. Pant, M.; Thangaraj, R.; Abraham, A. Particle swarm optimization using adaptive mutation. In Proceedings of the 2008 19th International Workshop on Database and Expert Systems Applications, Turin, Italy, 1–5 September 2008; pp. 519–523.
27. Shi, Y. Particle swarm optimization: Developments, applications and resources. In Proceedings of the 2001 Congress on Evolutionary Computation (IEEE Cat. No. 01TH8546), Seoul, Republic of Korea, 27–30 May 2001; Volume 1, pp. 81–86.
28. Marini, F.; Walczak, B. Particle swarm optimization (PSO). A tutorial. *Chemom. Intell. Lab. Syst.* **2015**, *149*, 153–165. [CrossRef]
29. Garcia-Gonzalo, E.; Fernandez-Martinez, J.L. A brief historical review of particle swarm optimization (PSO). *J. Bioinform. Intell. Control* **2012**, *1*, 3–16. [CrossRef]
30. Kim, S.; Hong, S.; Joh, M.; Song, S.K. Deeprain: Convlstm network for precipitation prediction using multichannel radar data. *arXiv* **2017**, arXiv:1711.02316.
31. Kelotra, A.; Pandey, P. Stock market prediction using optimized deep-convlstm model. *Big Data* **2020**, *8*, 5–24. [CrossRef] [PubMed]
32. Kline, D.M. Methods for multi-step time series forecasting neural networks. In *Neural Networks in Business Forecasting*; IGI Global: Hershey, PA, USA, 2004; pp. 226–250.
33. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, Ł.; Polosukhin, I. Attention is all you need. *Adv. Neural Inf. Process. Syst.* **2017**, *30*. [CrossRef]
34. Zhou, T.; Ma, Z.; Wen, Q.; Wang, X.; Sun, L.; Jin, R. Fedformer: Frequency enhanced decomposed transformer for long-term series forecasting. In Proceedings of the International Conference on Machine Learning, PMLR, 2022, Baltimore, MD, USA, 17–23 July 2022; pp. 27268–27286.