# Highlights

**Unleashing the Power of Deep Learning: A Performance Comparison of LSTM, GRU, and Transformer Modelsa**

Jagdish Chakole, Muktinath Vishwakarma, Nachiket N Doddamani, Manish Kurhekar,

- Research highlight 11
- Research highlight 2

# Unleashing the Power of Deep Learning: A Performance Comparison of LSTM, GRU, and Transformer Models

Jagdish Chakole[a,,*], Muktinath Vishwakarma[b], Nachiket N Doddamani[a], Manish Kurhekar[b],

[a] *Indian Institute of Information Technology Nagpur, Maharashtra, India*
[b] *Visvesvaraya National Institute of Technology Nagpur, Maharashtra, India*

**Abstract**

Metals, particularly non-ferrous metals, play a pivotal role as raw materials in numerous industries, serving as essential components for a nation's industrial and economic growth. The fluctuating prices of these metals pose a significant challenge for industries aiming to procure them at optimal costs. While various approaches, including machine learning, have been employed for metal price prediction, there remains room for improvement in the accuracy and efficiency of these models. Notably, recent advancements in deep learning have demonstrated impressive performance. This research delves into the application of state-of-the-art deep learning models tailored for sequential data to enhance the prediction accuracy of non-ferrous metal prices. The main contribution of this research work is the use of the Transformer model to develop a predictive model to predict the non-ferrous metal price and its comparison with the state-of-the-art deep learning models tailored for sequential data. Through empirical studies, we explore the efficacy of prominent deep learning architectures, including Long Short-Term Memory (LSTM), Gated Recurrent Unit (GRU), and Transformer models. The objective is to discern the model that exhibits superior performance in capturing the intricate patterns and temporal dynamics inherent in metal price time series data. Our experimental results show that the proposed Transformer-

*Corresponding author

*Email addresses:* jchakole@iiitn.ac.in (Jagdish Chakole), author2@example.com (Muktinath Vishwakarma), nachiketdoddamani@gmail.com (Nachiket N Doddamani), manishkurhekar@cse.vnit.ac.in (Manish Kurhekar)

based model outperformed the LSTM, and GRU-based models to predict non-ferrous metal prices including Aluminium, Copper, Zinc, and Nickel.

*Keywords:*

---

## 1. Introduction

Metals are the basic raw materials for many industries. Many metals are used to manufacture machines for diverse sectors like agriculture, automobile, construction, and more. Almost all industries directly or indirectly depend upon metals. So, the price of the finished product depends on the metals, that are required as raw materials to a product or as raw materials to equipment required to manufacture the product. The price of the metals depends on many diverse factors including demand and supply of the metal, geopolitical situation, etc. The price of the metals fluctuates over time. One of the objectives of the industry is to purchase metals at an optimal price as the price of their finished product is based on raw materials price. Governments and industries are in need to know the future price of these metals for decision-making.

The motivation for this work is related to the advancement in computer technology and the Data Science domain and its ubiquitous use. The majority of the global stock and commodity exchanges are working online because of the technological advancements. Historical and current trading data like price action, and volume data is easily available at affordable cost in electronic form [1]. The performance of data-driven machine learning and deep learning methods[2] has improved significantly. The availability of models and clean datasets have the potential to predict future prices or trends.

In this research work, we are interested in predicting the future price of non-ferrous metals as it is significant for the growth and development of the country. We are specifically interested in non-ferrous metals, as these metals are mostly useful in manufacturing industries because of their significant properties [3]. Non-ferrous metals are non-magnetic and non-iron based metals like Aluminium, Copper, Zinc, Nickel, etc.

This research makes two significant contributions. Firstly, it introduces a novel model leveraging the Transformer architecture for predicting metal prices, marking a departure from conventional approaches. Notably, prior studies have not explored the application of Transformer models for forecasting non-ferrous metal prices. Secondly, the study conducts a comprehensive

performance comparison between the proposed Transformer-based model and a state-of-the-art deep learning model designed for analyzing sequential data.

The time series dataset for the daily prices of non-ferrous metals exhibits a consistent time difference between consecutive data points, rendering it a sequential data format. Similarly, Natural Language Processing (NLP) data also follows a sequential structure. Recently, there has been a notable surge in the development of deep learning-based predictive models for sequential data within the NLP domain, demonstrating noteworthy performance. This research delves into the application of such models for predicting metal prices.

The London Metal Exchange (LME) stands as a prominent global platform facilitating the trade of metals. The pricing dynamics of numerous metals, particularly non-ferrous ones, are internationally influenced by the LME. Hence, the dataset sourced from the LME serves as the foundation for our research endeavors.

The rest of the paper is organized as Section 2 covers the motivation and background that discusses the various approaches and methods used to date to predict metal prices. Section 3 discusses the method used in this research work. The Transformer-based architecture of the proposed system is discussed in Section 4. In Section 5, we discuss the experimental dataset and performance evaluation measures used in this research work. The experimental results and their significance are discussed in Section 6. Section 7 concludes the paper.

## 2. Motivation and Background

In the subsequent section, we explore previous research endeavors focusing on forecasting metal prices conducted by various scholars. Kriechbaumer et al. [4] introduced an enhanced forecasting methodology for non-ferrous metal prices utilizing a Wavelet-ARIMA model. This model dissects the price time series into frequency and time domains, thereby refining forecasting precision. In another study, Kristjanpoller et al. [5] proposed a hybrid model integrating Artificial Neural Networks (ANN) and the Generalized Autoregressive Conditional Heteroskedasticity (GARCH) model to predict the volatility of copper, gold, and silver. Concerning short-term gold price prediction, Hajek et al. [6] introduced a fuzzy rule-based system incorporating financial price data and news sentiment data for training. They argue that news data is more relevant for predicting prices one day ahead, while historical trading data gains significance for forecasting prices five days in advance.

Guha et al. [7] employed the ARIMA model to forecast future gold prices in the Indian stock market using price series data. Additionally, Zhichao He et al. [8] proposed a hybrid forecasting model for non-ferrous metal prices. Their approach entails decomposing the price series into multiple subseries, utilizing the ARIMA model for forecasting low-frequency subseries, and subsequently aggregating them to reconstruct the original series. Notably, the authors underscored the optimal utilization of residual values from the series for effective forecasting.

Zhou et al. [9] predicted the prices of gold and palladium using the Deep Regularization Self-Attention Regression model, incorporating components such as CNN, LSTM, and self-attention. Their approach involved extracting spatial features through CNN and temporal features via LSTM. This model outperformed the ARIMA, SVR, CNN, and LSTM models. Addressing the instability of raw single series price data on the LME, Liu et al. [10] decomposed the data using variational mode decomposition into subseries, employing LSTM for forecasting each subseries and aggregating them with an aggregation function. Astudillo et al. [11] utilized support vector regression to forecast copper prices for the next 5 and 10 days, achieving an RMS of $\leq 2.2\%$ in their experimentation with London Metal Exchange data. Li et al. [12] proposed a model based on LSTM and Multivariate Mode Decomposition for metal price prediction, arguing that fusion models based on decomposition outperform single models.

Artificial Neural Network-based framework used in Méndez-Suárez et al.[13] to predict copper price five days ahead based on the current price data. Shao et al. [14] optimized the parameters of the LSTM model using improved particle swarm optimization to predict the nickel price. The experimental results concluded that the proposed model outperformed the conventional LSTM and ARIMA. The EWT-GBDT model, combining Empirical Wavelet Transform (EWT) and Gradient Boosting Decision Trees, was introduced in the work by Gu et al. [15]. This model aimed to forecast the future price of nickel on the London Metal Exchange. The process involves selecting input features based on their correlation with the nickel price and subsequently decomposing the chosen input feature series into subseries using EWT. The resulting decomposed features are then fed into the Gradient Boosting Decision Trees (GBDT) for the final price prediction. According to their experimental results, the proposed EWT-GBDT model demonstrated superior performance compared to both the standalone GBDT and Adaboost models. Also, Zhao et al.[16], used a decomposition-based model for predic-

tion.

We can outline that researchers initially experimented with statistical models such as the ARIMA model, which later evolved into variants or combinations of these models for metal price prediction. Subsequently, some researchers delved into machine learning techniques, followed by deep learning methods tailored for sequential data analysis. In our study, we investigate the effectiveness of contemporary deep learning models for sequential data in predicting metal prices.

Researchers have explored the versatility of the Transformer across various domains, demonstrating its effectiveness in many cases. Motivated by these successes, this work aims to investigate the Transformer architecture's applicability in predicting non-ferrous metal prices. Dosovitskiy et al. [17] delved into the application of the Transformer architecture in computer vision, revealing that the Vision Transformer outperformed traditional convolutional network-based models. Furthermore, the object detection Transformer was employed in the study by Carion et al. [18]. Additionally, Dong et al. [19] introduced the Speech-Transformer model for speech recognition tasks, while Payne et al. [20] utilized the Transformer for generating musical compositions. In the medical domain, Valanarasu et al. [21] leveraged the Transformer architecture for image segmentation tasks. These studies collectively highlight the broad spectrum of domains where the Transformer architecture has shown promising performance.

## 3. Methodology

The objective of this research work is to explore the capabilities of the Transformer for the prediction of the non-ferrous metal price. We have compared the performance of the proposed model with the classic deep learning methods for sequential data viz RNN, LSTM, and GRU. This section introduces the intuition and architectures of these methods.

### *3.1. RNN*

A feedforward Artificial Neural Network (ANN) is suitable for data with spatial features but it is not suitable for data having temporal features i.e. sequential data. In sequential data, there is a dependency between data at time instant $t$ and at time instant $t-1$. The ANN is not able to handle this dependency as it treats every data point independently. Recurrent Neural Networks (RNN) is a variant of ANN to handle sequential data [22]. As

shown in Fig. 1, in RNN the hidden layers are treated as a state (hidden state). The hidden state at time step $t$ is based on the hidden state at time step $t-1$ and input data (features) at time instant $t$ as indicated in Equation 1. A hidden state acts as a memory that stores information till the previous data point. The output of RNN at time $t$ is based on hidden state $h_t$ at time $t$ as shown in Equation 2.

$$h_t = \tanh(x_t, h_{t-1}) \tag{1}$$

$$O_t = f(h_t) \tag{2}$$

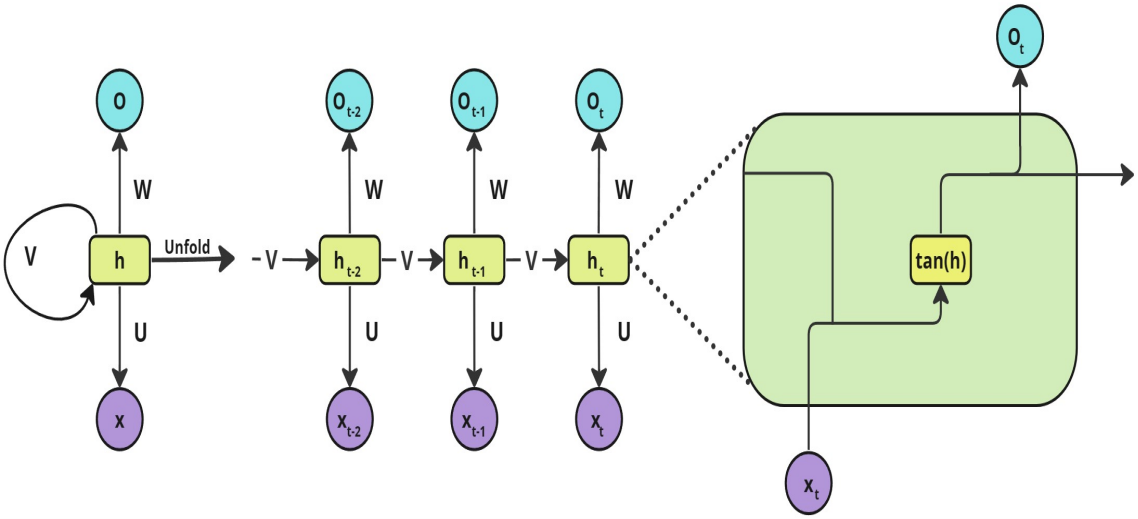


Figure 1: Recurrent Neural Network Architecture

Fig. 1, shows hidden states of RNN at different time instances. These different time instances share the same parameters. The RNN is capable of handling data dependencies in the sequential data but has some issues like it may have a Vanishing and Exploding gradient problem. The hidden state in RNN has limited capability so, it is not able to handle data dependencies if data is separated by many time steps.

*3.2. Long Short-Term Memory (LSTM)*

The RNN can handle the dependency between the data if the two relevant data points are not far apart but if there is a long-term dependency between data then the RNN suffers from vanishing gradient problem [23]. The LSTM model is a variant of the RNN model and it can handle this long-term dependency problem [24].

As shown in Fig. 2, in addition to hidden state $h_t$ the LSTM has one more state called cell state $C_t$. The hidden state is short-term memory and the cell state is long-term memory. The cell state $C_t$ is like a convey belt that passes information from the current time step to the next time step. Unlike RNN the LSTM does not remember all previous information. It only remembers relevant information, passes only necessary information to the next time step, and takes only fruitful information from the input using three gates viz. forget gate, output gate, and input gate. In Fig. 2, The forget gate $f_t$ govern by the following equation, decides which part of the input cell state $C_{t-1}$ is forgeted.

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \tag{3}$$

The input gate decides this part of the input added to the cell state $C_t$, govern by following equations as $i_t * \tilde{C}_t$.

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \tag{4}$$

$$\tilde{C}_t = \tanh(W_c \cdot [h_{t-1}, x_t] + b_c) \tag{5}$$

The cell state $C_t$ is computed as

$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t \tag{6}$$

The output $h_t$ is computed as

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) \tag{7}$$

$$h_t = o_t * \tanh(C_t) \tag{8}$$

In the above equations, $W$ is the weight vector and $b$ is the bais of the respective gate.

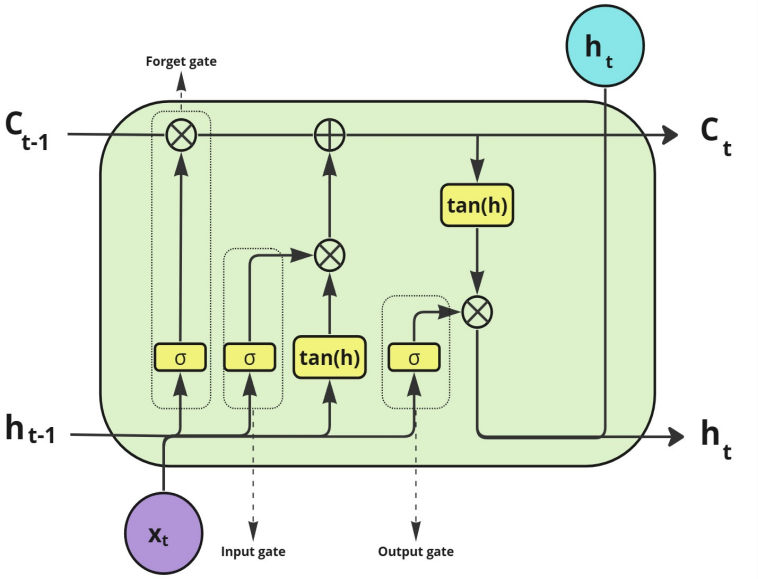


Figure 2: LSTM architechture

### *3.3. GRU*

Gated Recurrent Unit (GRU) is introduced in [25], like LSTM it is also a type of RNN that addresses the vanishing gradient problem. Like LSTM it also has gates but only two gates instead of three gates in LSTM. These two gates are namely the reset gate and the update gate as shown in Fig. 3. Unlike LSTM there is only one state. It has only a hidden state no cell state. The GRU architecture is less complex compared to LSTM. It is computationally less complex compared to LSTM. LSTM can capture better long-term dependency. The reset gate equation is as follows. The vector $r_t$ depends on the previous hidden state $h_{t-1}$ and current input $x_t$

$$r_t = \sigma(W_r \cdot [h_{t-1}, x_t] + b_r) \tag{9}$$

The update gate is as follows.

$$u_t = \sigma(W_u \cdot [h_{t-1}, x_t] + b_u) \tag{10}$$

The equation for candidate hidden state $\tilde{h_t}$ is as follows. where $\odot$ is elementwise multiplication.

$$\tilde{h_t} = \tanh(W_h \cdot [r_t \odot h_{t-1}, x_t] + b_h) \tag{11}$$

The current hidden state $h_t$ is computed as follows.

$$h_t = (1 - u_t) \odot h_{t-1} + u_t \odot \tilde{h}_t \tag{12}$$

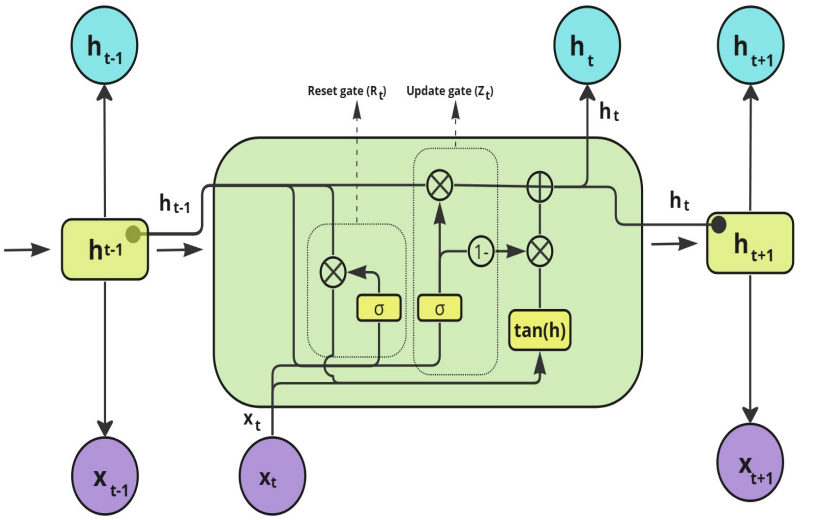


Figure 3: GRU architechture

### *3.4. Transformer*

Unlike LSTM, GRU, and other primitive RNN-based models that rely on sequential processing, the Transformer on the other hand adopts a localized and hierarchical processing approach inspired by Convolutional Neural Networks (CNN). The key innovation lies in the integration of attention-based mechanisms which allows parallel computation and makes it possible to capture long-range dependencies thereby making it more efficient as shown in Fig. 4.

- **Attention Mechanism**
  It is a mechanism that assigns different levels of importance to different parts of the input sequence, enabling the model to selectively focus on different parts of the input sequence when producing an output, thereby

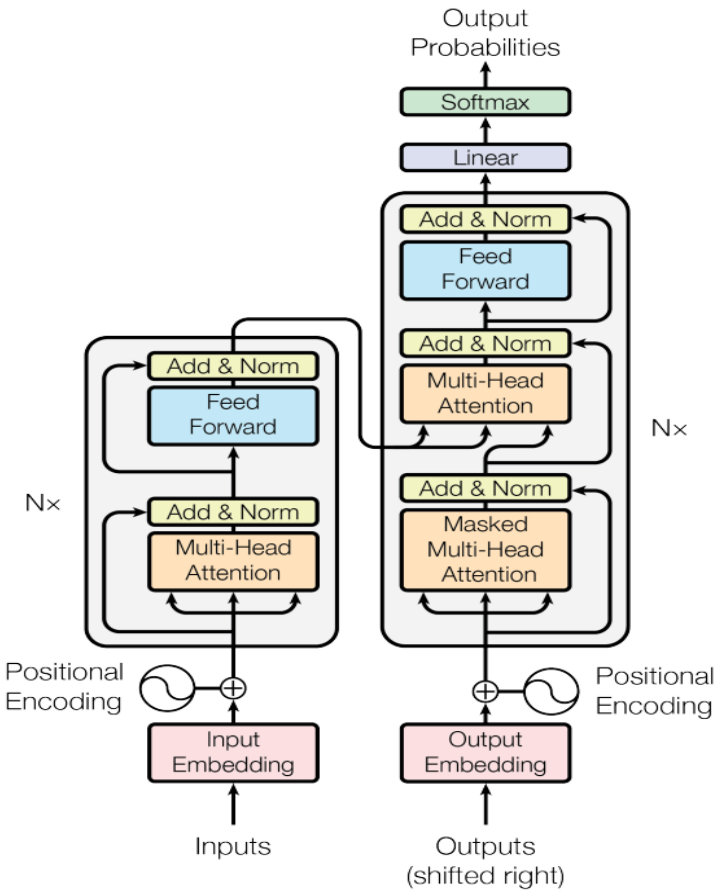


Figure 4: The Transformer - model architecture[26]

selecting the most relevant information [27]. Attention mechanisms are classified into self-attention, multi-head attention, and local attention, each offering different benefits and trade-offs.

- **Transformer**
  Introduced in the paper [26], Transformer architecture is the fundamental building block of Large Language Models (LLMs). The self-attention mechanism helps to capture long-distance context so that the sequence of the hidden states also attends to itself. A Transformer model can be classified into three categories: Encoders, Decoders, and Encoder-decoder models[28]. The attention mechanism involves three sets of matrices: query(Q), key(K), and value(V). These matrices are derived from the input sequence.

  - **Query (Q)**
    The query vector represents the element that is currently being focused on for the given context. In the context of self-attention mechanisms, it's the vector of target input for which the attention weights are obtained. By utilizing the query matrix to alter the word representation, the system produces a query vector. This vector is then employed to compare against other words within the sentence.
    $q_i = W_q x_i$, where $x_i$ represents the input query vector at position $i$, $q_i$ represents the projected query vector at position $i$,$W_q$ represents the weight matrix used for the query projection.
  - **Value (V)**
    The system utilizes the key matrix to generate key vectors for every word present in the sentence. These key vectors are used in evaluating the relevance or similarity between the target word (using the query vector) and other words within the sentence. A greater similarity score between the query vector and a key vector signifies a more robust relationship between the respective words.
    $v_i = W_v x_i$
  - **Key (K)**
    The value matrix produces value vectors for each word in the sentence. These vectors contain the contextual details of individual words. Following the computation of similarity scores using query and key vectors, the system proceeds to calculate a weighted sum

of the value vectors. The weights assigned to each value vector are based on the similarity scores, thereby ensuring that the resulting contextual representation is predominantly influenced by relevant words.
$k_i = W_k x_i$

- The encoding results in the word $C_i$, by applying attention mechanism to the projected vectors.
$r_{ij} = (q_i.k_j)/\sqrt{d}$

$a_{ij} = e^{r_{ij}}/(\sum_k e^{r_{ik}})$

$c_i = \sum_j a_{ij}.V_j$
where, $d$ represents the dimensionality of the key vectors.

The self-attention mechanism aims in capturing long-range context, enabling the sequence of hidden states to attend to itself. Self-attention represents just one aspect of the Transformer model, which comprises multiple sub-layers within each Transformer layer. Initially, self-attention is applied at every Transformer layer. Following this, the attention module's output undergoes processing through feedforward layers, where identical feedforward weight matrices are independently applied at each position. Subsequently, a nonlinear activation function, commonly ReLU, is typically applied after the initial feedforward layer.

- Inputs: Input sequential data tokens are converted to numerical representations called "input embeddings" using a dictionary-like mapping, where similar words have similar vector representations.
- Positional Encoding: Encodes the order of words in the input sequence as numbers, enabling the model to understand sentence structure.
$PE_{(pos,2i)} = \sin(\frac{pos}{10000^{2i/d_{model}}})$

$PE_{(pos,2i+1)} = \cos(\frac{pos}{10000^{2i/d_{model}}})$
where:
  * $pos$ is the position of the token in the sequence
  * $i$ is the dimension index within positional encoding.
  * $d_{model}$ is the dimension of the model

- Encoder: Processes the input text through multiple self-attention layers, generating hidden states that capture meaning and context at different abstraction levels.
- Outputs (shifted right): During training, the decoder predicts the next word based on previous words, achieved by shifting the output sequence.
- Output Embeddings: Similar to input embeddings, representing predicted words as numbers and applying positional encoding.
- Loss Function: Measures the difference between predictions and actual text, guiding parameter adjustments to improve accuracy.
- Decoder: Uses the encoded input and positional information to generate natural language text, similar to the encoder, using multiple layers.
- Linear Layer and Softmax: Transforms output embeddings back to the original input space and generates a probability distribution for each possible output token.

## 4. Proposed Models

The daily price sequence of the Aluminium on the London Metal Exchange is the time series data. Researchers used this price data to generate derived spatial features to train machine learning models. Some of them used this price data to generate temporal features to train time series based models for price prediction. As the metal price data is time series data, the use of a time series or sequential model for the prediction of metal price looks promising.

The main problem with the sequential data is to identify which part of the sequence is relevant for current prediction. The deep learning model LSTM has the capability to capture the relevant information from the input sequence using selective read, write, and forget. The encoder-decoder model is suitable for sequential data but it is not capable of handling long-term dependency. The attention mechanism resolved this problem but as the attention-based encoder-decoder model is based on RNN, we have to provide input in a sequential manner. This is the hurdle for parallelism.

The Transformer is a recent deep learning algorithm that has shown outstanding performance on the sequential data in the NLP domain. The Transformer is based on the Attention mechanism to capture relevant information

of the given sequential data. The Transformer model is also based on the encoder-decoder model but it doesn't use RNN. Instead, it uses self-attention encoding and decoding. In Transformer parallel processing is possible, so no need to provide data sequentially.

In this research work our main objective is the identify the relevant parts of the input sequence for optimal prediction of the metal price using the state-of-the-art deep learning framework Transformer. We aim to explore the Transformer abilty of paying attention on relavant part of the data to predict the future price of the non-ferrous metals. Also, our another objective is to comapare the performance of the Transformer based model with other prominent deep learning approaches like LSTM, GRU for non-ferrous metals price prediction. We have proposed a Transformer-based deep learning model for metal price prediction as shown in Fig 6.

As in Fig 6, firstly we perform data preprocessing like handling NaN values. Then we transform the raw data into label data for the Supervise Learning problem. Input feature is thirty days close price $[d_{t-29}, d_{t-28}, ....., d_{t-5}, d_{t-4}, d_{t-3}, d_{t-2}, d_{t-1}, d_t]$ and expected output is close price for $t+1$ day $d_{t+1}$. We have to perform positional encoding to preserve the sequence information. This input is passed to the encoder of the Transformer, where self-attention is performed to obtain revised context vector embedding. Encoder operations are performed more than one time and then the final output of the encoder is passed to the fully connected layer. There is no need for the decoder as it is a time series prediction problem. The fully connected layer generates the output i.e., the closing price for the $t+1$ day based on the input previous thirty days' close price of the metal.

Figure 5 is a heatmap that illustrates the correlation between various features. Heatmaps are useful for visualizing the relationships between different variables. In this case, we aimed to understand the experimental data by investigating the potential correlation between non-ferrous metal prices and coal prices. Initially, we hypothesized that there might be a relationship between metal prices and coal prices. To test this, we conducted a correlation analysis between these variables. The results indicated that there is no significant correlation between coal prices and metal prices. The heatmap provides a matrix of features with correlation coefficients ranging from -1 to +1. As shown in Figure 5, we can conclude that the prices of the four experimental metals are correlated with each other.

In our proposed model, we used a batch size of 32 and ran the experiments on Google Colab. The sequence length was set to 30, meaning we used the

previous 30 days' closing prices of the metal to predict the next day's price. The training time, including validation, was approximately 10 minutes for 100 epochs. Also in all our baseline models sequence length was set to 30.

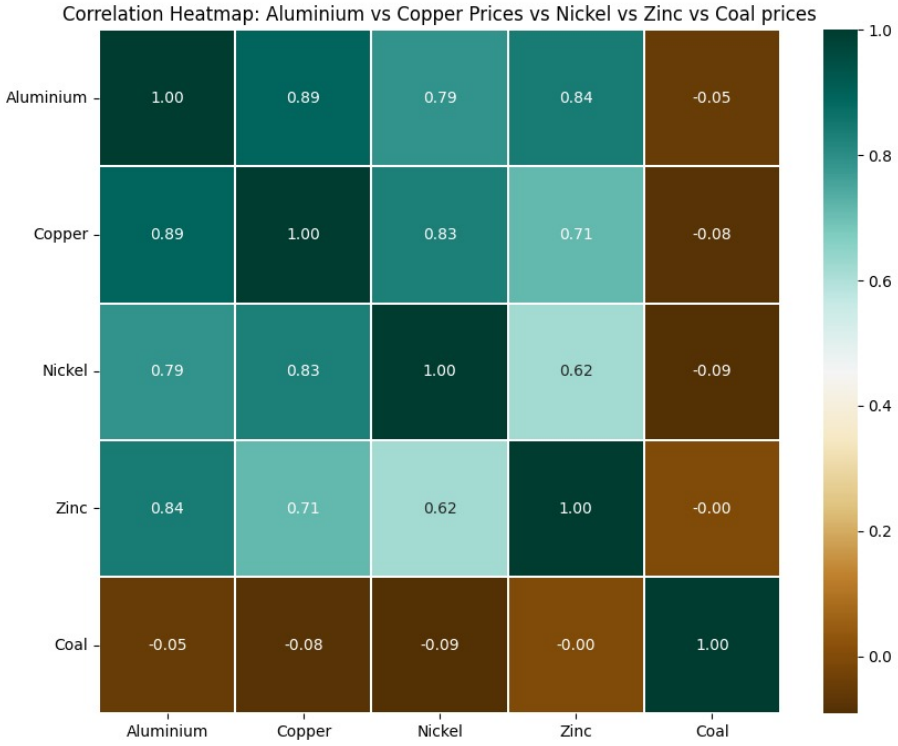


Figure 5: HeatMap

## 5. Experimentation

### *5.1. Experimental Data*

The global pricing of base metals such as Aluminium, Copper, Zinc, and others is determined by the London Metal Exchange, a renowned commodities exchange based in London, United Kingdom, established in 1877.

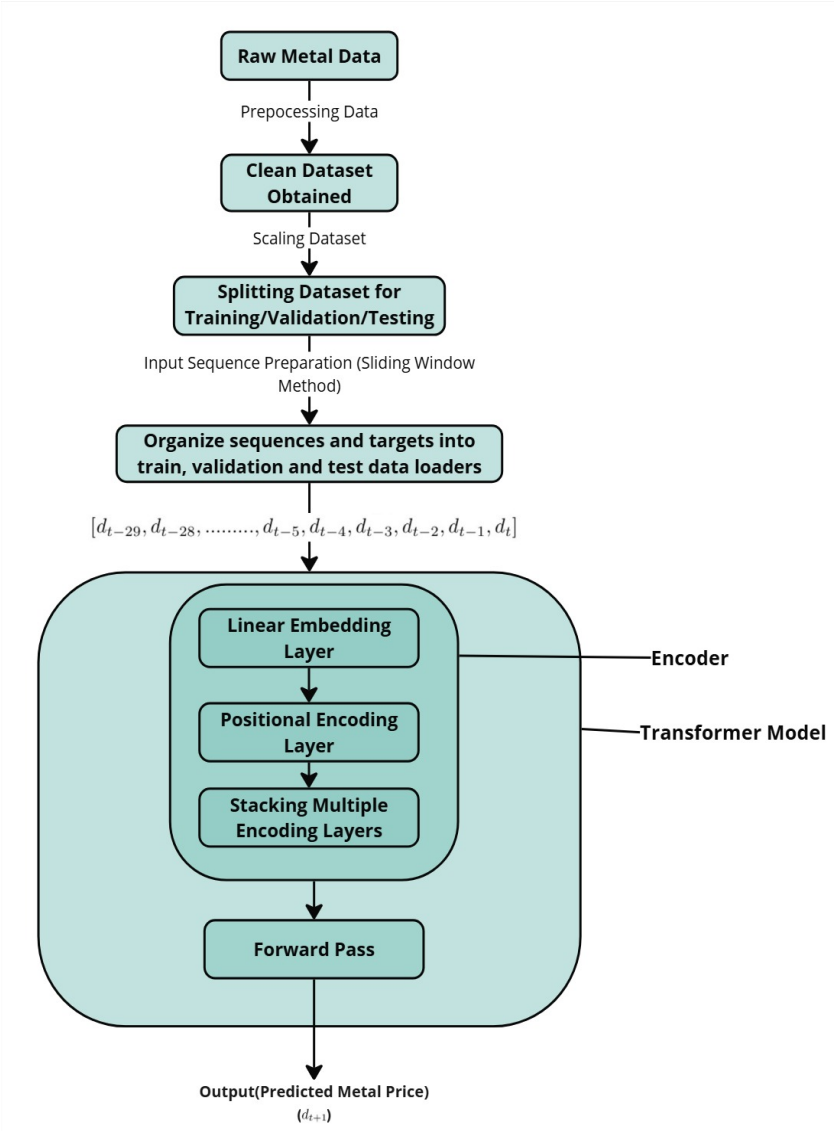


Figure 6: Proposed System

Experimental Data of non-ferrous Metal Prices from LME.

| **Stock Name** | **Time** | **Total Period** | **Training Period** | **Validation Period** | **Testing Period** |
|---|---|---|---|---|---|
| 2*Aluminium | Start | 2008-01-02 | 2008-01-02 | 2019-07-18 | 2022-11-03 |
| | End | 2024-07-01 | 2019-07-17 | 2022-11-02 | 2024-07-01 |
| 2*Copper | Start | 2008-01-02 | 2008-01-02 | 2019-07-18 | 2022-11-03 |
| | End | 2024-07-01 | 2019-07-17 | 2022-11-02 | 2024-07-01 |
| 2*Zinc | Start | 2008-01-02 | 2008-01-02 | 2019-07-18 | 2022-11-03 |
| | End | 2024-07-01 | 2019-07-17 | 2022-11-02 | 2024-07-01 |
| 2*Nickel | Start | 2008-01-02 | 2008-01-02 | 2019-07-18 | 2022-11-03 |
| | End | 2024-07-01 | 2019-07-17 | 2022-11-02 | 2024-07-01 |

The LME facilitates futures and options trading for various base and precious metals. Traders can engage in trading contracts for future delivery or options on a daily, weekly, and monthly basis. The prices of Aluminium, Copper, Zinc, and Nickel are denoted in US dollars per metric ton on the LME platform. In this work, we are interested in predicting the future price of the non-ferrous metal. Table 5.1 presents the experimental data used in this work. Since metal price data is a time series, traditional random shuffling and splitting for cross-validation would disrupt the temporal dependencies. Therefore, we used the first continuous 70% of the data for training, the next 20% for validation, and the remaining 10% for testing. Out of a total of 4171 days, 2920 days were allocated for training, 834 days for validation, and 417 days for testing. We acquired the experimental trading datasets for non-ferrous metals from the LME via Westmetall (https://www.westmetall.com/en/markdaten.php). Python was utilized for coding purposes. The code for both the proposed system and the baseline model is available in our GitHub repository at *https://github.com/Nachiket1234/LME-Metal-Price-Prediction*.

Figure 7, Figure 8, Figure 9, Figure 10 are the plots of the experimental datasets used in this work, it show the daily closing price of the stock.

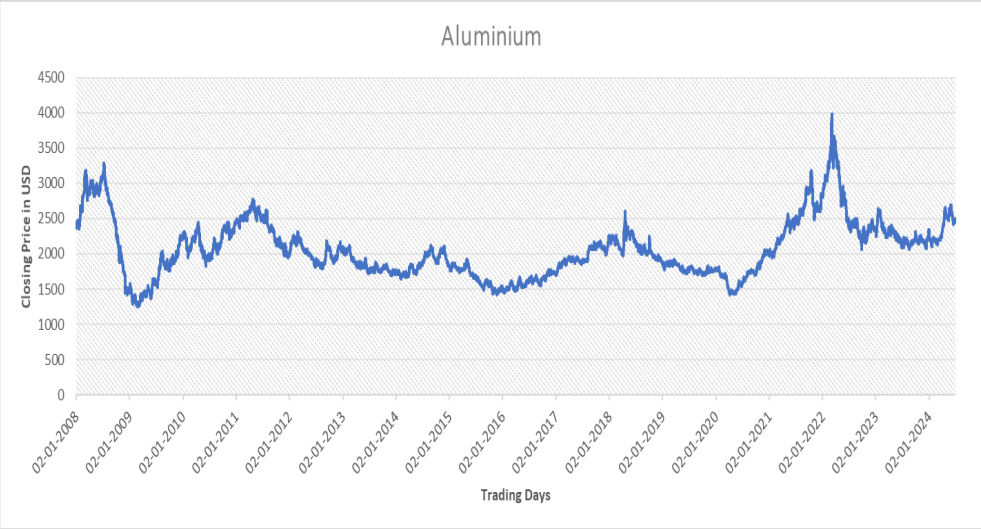


Figure 7: Daily close price series for Aluminium.

### *5.2. Performance Evalution Measures*

Performance evaluation metrics are essential for assessing the effectiveness of models in various tasks, including regression problems. Mean Squared Error (MSE), Mean Absolute Error (MAE), and Mean Absolute Percentage Error (MAPE) are commonly used metrics for evaluating the performance of regression models.

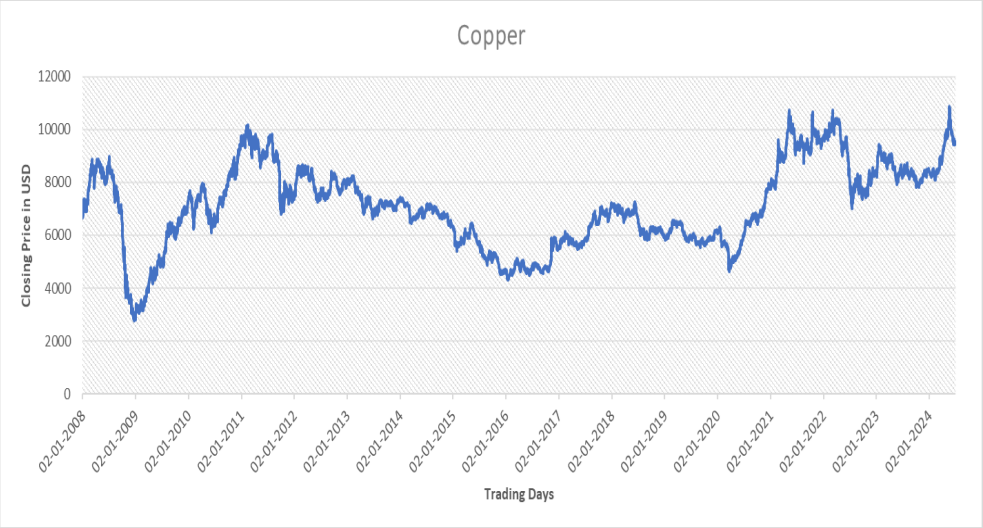


Figure 8: Daily close price series for Copper.

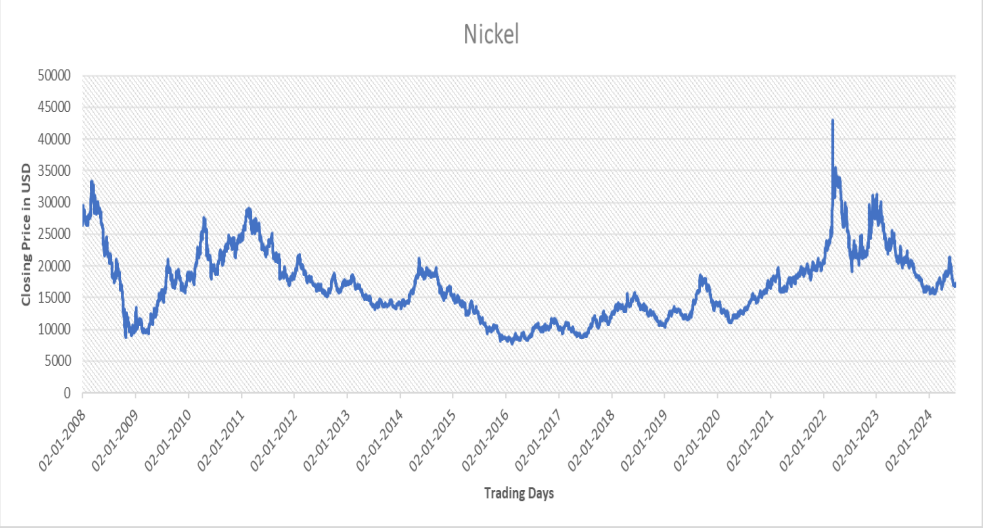


Figure 9: Daily close price series for Nickel.

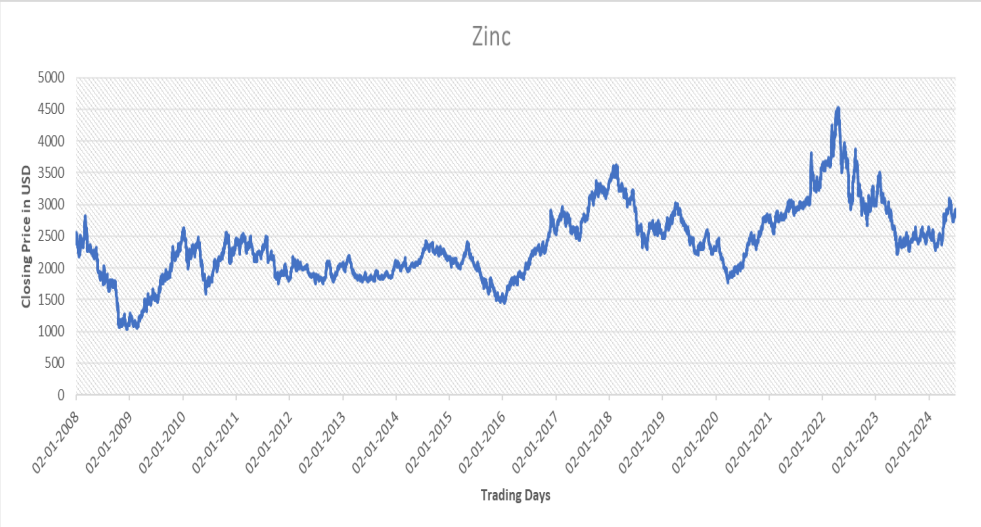


Figure 10: Daily close price series for Zinc.

*5.2.1. Mean Squared Error(MSE)*

A lower MSE indicates better model performance, and the metric is sensitive to outliers due to the squaring of errors.

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \tag{13}$$

where:

$n$ : Number of data points

$y_i$ : Actual value of the i-th data point

$\hat{y}_i$ : Predicted value of the i-th data point

*5.2.2. Mean Absolute Error(MAE)*

Like MSE, a lower MAE indicates better model performance, and the metric is less sensitive to outliers than MSE.

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i| \tag{14}$$

*5.2.3. Mean Absolute Percentage Error (MAPE)*

MAPE provides a percentage measure of the average relative error. A lower MAPE indicates better model performance, but it is sensitive to zero

values in the actual data.

$$MAPE = \frac{1}{n} \sum_{i=1}^{n} \left| \frac{y_i - \hat{y}_i}{y_i} \right| \times 100\% \tag{15}$$

where:

$n$ : Number of data points
$y_i$ : Actual value of the i-th data point
$\hat{y}_i$ : Predicted value of the i-th data point

## 6. Experimental Results

We experimented with the proposed model on four non-ferrous metals viz Aluminium, Copper, Zink, and Nickel. We also compared the performance of the proposed model with the state-of-the-art sequential deep learning methods LSTM and GRU. Experimented with univariate and multivariate LSTM and GRU. All models are trained and tested on the same dataset for fair comparison. The performance of the models is compared using the performance evaluation measures discussed in Section 5.2 on testing data only.

The experimental results of all models including the Transformer-based proposed model for experimental datasets of Aluminium, Copper, Zink, and Nickel are shown in the Table 6, 6, and 6 in terms of MSE, MAE, and MAPE respectively. The proposed Transformer-based model outperformed all four baseline models on all four metals datasets with regards to MSE, MAE, and MAPE.

The performance of all experimental models in terms of MSE on all experimental datasets is compared in Table 6, while Figure 11 graphically depicts the same results. It is evident from both Table 6 and Figure 11 that the proposed model outperforms the baseline models in terms of MSE. Additionally, Table 6 and Figure 12 demonstrate the superiority of the proposed model in terms of MAE, while Table 6 and Figure 13 illustrate its superiority in terms of MAPE.

Comparison of the performance of the five models, including the proposed Transformer-based model, across all four datasets in terms of Mean Squared Error.

| **Model** | **Aluminum** | **Copper** | **Nickel** | **Zinc** |
|---|---|---|---|---|
| Transformer | 0.0001 | 0.0003 | 0.0002 | 0.0003 |
| LSTM Multivariate | 0.0002 | 0.0004 | 0.0003 | 0.0004 |
| GRU Multivariate | 0.0002 | 0.0005 | 0.0003 | 0.0004 |
| LSTM | 0.0003 | 0.0004 | 0.0004 | 0.0005 |
| GRU | 0.0003 | 0.0004 | 0.00035 | 0.0005 |

Comparison of the performance of the five models, including the proposed Transformer-based model, across all four datasets, evaluated in terms of Mean Absolute Error.

| **Model** | **Aluminum** | **Copper** | **Nickel** | **Zinc** |
|---|---|---|---|---|
| Transformer | 0.01026 | 0.0132 | 0.01146 | 0.0143 |
| LSTM Multivariate | 0.0114 | 0.0161 | 0.0121 | 0.0163 |
| GRU Multivariate | 0.0122 | 0.0163 | 0.0125 | 0.0161 |
| LSTM | 0.0138 | 0.0174 | 0.0143 | 0.0181 |
| GRU | 0.0128 | 0.0161 | 0.0136 | 0.0178 |

Comparison of the performance of the five models, including the proposed Transformer-based model, across all four datasets in terms of Mean Absolute Percentage Error.

| **Model** | **Aluminum** | **Copper** | **Nickel** | **Zinc** |
|---|---|---|---|---|
| Transformer | 2.65 | 1.80 | 2.88 | 2.99 |
| LSTM Multivariate | 3.01 | 2.15 | 3.13 | 3.40 |
| GRU Multivariate | 3.15 | 2.20 | 3.08 | 3.47 |
| LSTM | 3.64 | 2.41 | 3.83 | 3.87 |
| GRU | 3.29 | 2.18 | 3.58 | 3.72 |

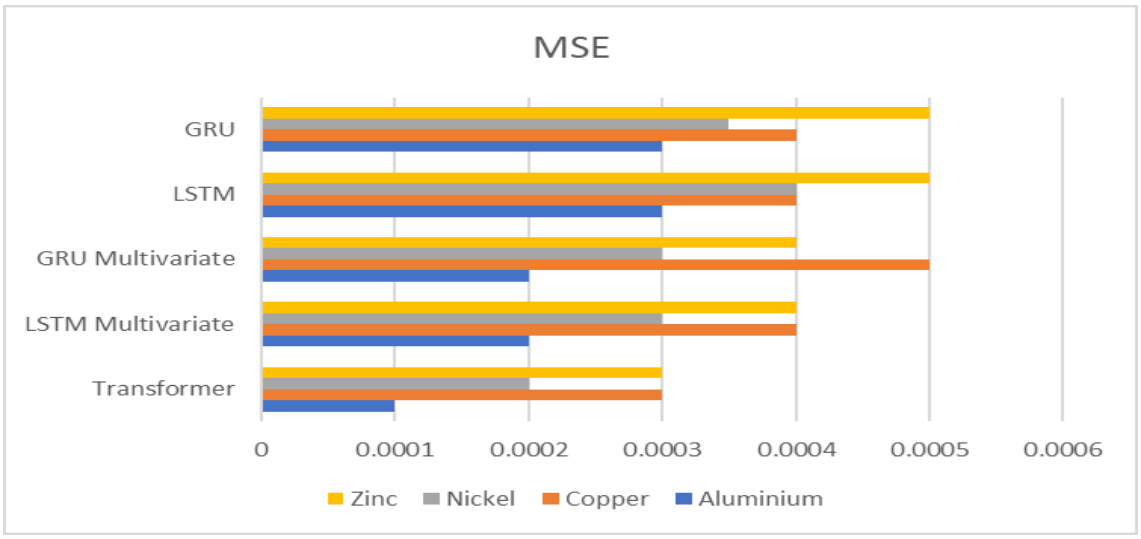


Figure 11: MSE of the five models including the proposed Transformer based model on testing datasets of four experimental metals.

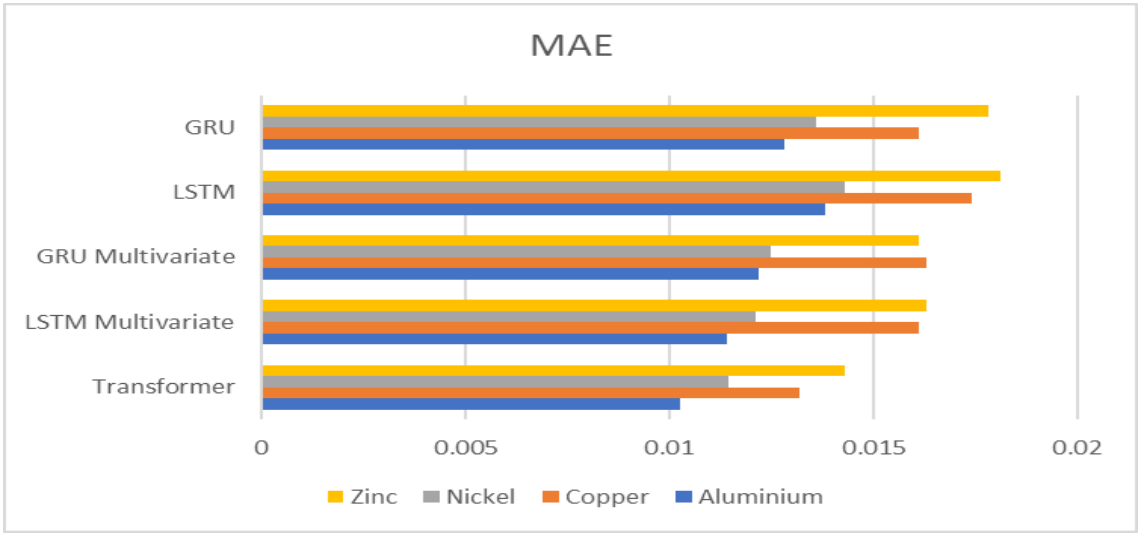


Figure 12: MAE of the five models including the proposed Transformer based model on testing datasets of four experimental metals.

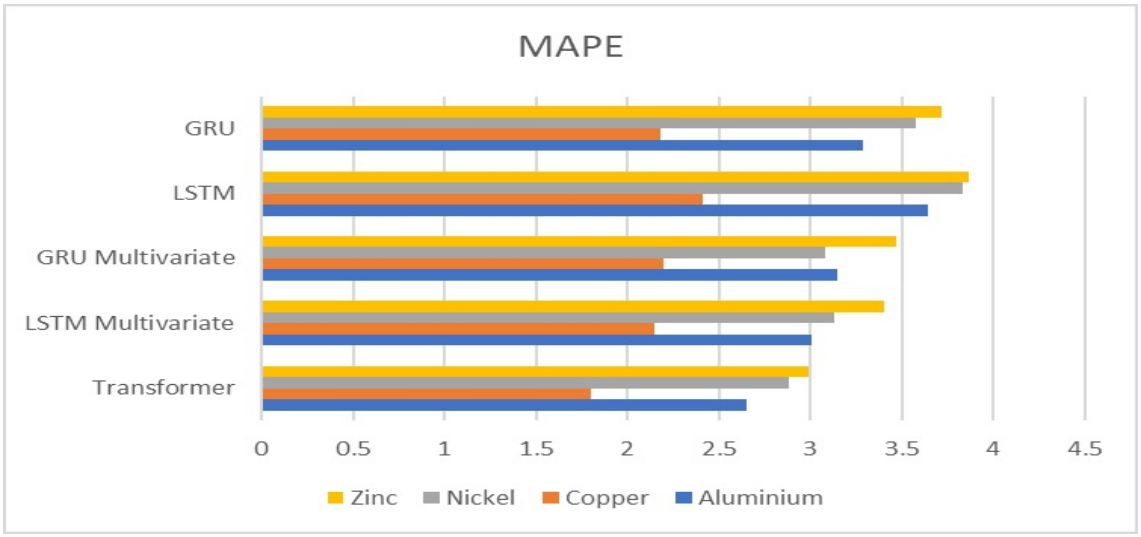


Figure 13: MAPE of the five models including the proposed Transformer based model on testing datasets of four experimental metals.

To determine whether the experimental results are statistically significantly different, we performed a T-test. The T-test assesses whether there is a statistically significant difference between two groups. For a fair comparison, we evaluated the proposed model against univariate LSTM and GRU models, as the proposed Transformer-based model is also univariate in our case.

Table 6 presents the Mean Squared Error (MSE) of the Proposed Model, the LSTM model, and the GRU model across four datasets. Table 6 displays the results of the T-test comparisons between the Proposed Model and the LSTM model, as well as between the Proposed Model and the GRU model. In Table 6, the p-values are less than the significance level ($\alpha$) of 0.05, indicating that we reject the null hypothesis, which states that there is no significant difference between the group means. Therefore, the results of the Proposed Model compared to both the LSTM and GRU models are statistically significantly different. Similarly, Table 6 shows comparable results for the same models on Mean Absolute Error (MAE) in Table 6.

MSE on four metals by Proposed Model, LSTM and GRU

| **Metal Name** | **Proposed Model** | **LSTM** | **GRU** |
|---|---|---|---|
| Aluminium | 0.0001 | 0.0003 | 0.0003 |
| Copper | 0.0003 | 0.0004 | 0.0004 |
| Nickel | 0.0002 | 0.0004 | 0.00035 |
| Zinc | 0.0003 | 0.0005 | 0.0005 |

Result of the T-test between different model's MSE.

| **T-test matrix** | **Proposed Model and LSTM** | **LSTM and GRU** |
|---|---|---|
| t-Statistic | -2.7815 | -2.5333 |
| p-Value | 0.0319 | 0.0444 |
| Significance Level($\alpha$) | 0.005 | 0.005 |

MAE on four metals by Proposed Model, LSTM and GRU

| **Metal Name** | **Proposed Model** | **LSTM** | **GRU** |
|---|---|---|---|
| Aluminium | 0.01026 | 0.0138 | 0.0128 |
| Copper | 0.0132 | 0.0174 | 0.0161 |
| Nickel | 0.01146 | 0.0143 | 0.0136 |
| Zinc | 0.0143 | 0.0181 | 0.0178 |

Result of the T-test between different model's MAE.

| **T-test matrix** | **Proposed Model and LSTM** | **LSTM and GRU** |
|---|---|---|
| t-Statistic | -2.5560 | -2.2135 |
| p-Value | 0.0431 | 0.0487 |
| Significance Level($\alpha$) | 0.005 | 0.005 |

## 7. Conclusion

Handling the long-term dependency on sequential data like metal price time series data is a problem of interest for industry and academia. LSTM results are better compared to RNN up to a certain extent. Recently deep learning model Transformer has shown significant performance improvement on sequential data. We proposed a model based on a Transformer to predict the future price of the non-ferrous metal using historical and current price time series data of the non-ferrous metal. Our experimental results on datasets of four non-ferrous metals viz Aluminium, Copper, Zink, and Nickel from LME conclude that the proposed model outperformed the baseline deep learning methods viz LSTM and GRU for sequential data. Till now Transformer for metal price prediction has not been studied. The main contribution of this research work is twofold. First, the use of a Transformer for non-ferrous metal price prediction. Second, is a comparison with the current state of art deep learning base model for sequential data with the proposed model. In the future, we would like to explore the Large Language Model for metal price prediction.

Our experiment on the four datasets indicates that the Transformer-based model outperformed the LSTM and GRU-based models to predict future metal prices in terms of MSE, MAE, and MAPE.

## References


[1] J. Chakole, M. Kurhekar, Tutorial on automated trading using api, in: Proceedings of the 6th Joint International Conference on Data Science & Management of Data (10th ACM IKDD CODS and 28th COMAD), 2023, pp. 305–307.

[2] J. Chakole, M. P. Kurhekar, Convolutional neural network-based a novel deep trend following strategy for stock market trading., in: CIKM Workshops, 2021.

[3] C. Watkins, M. McAleer, Econometric modelling of non-ferrous metal prices, Journal of Economic Surveys 18 (5) (2004) 651–701.

[4] T. Kriechbaumer, A. Angus, D. Parsons, M. R. Casado, An improved wavelet–arima approach for forecasting metal prices, Resources Policy 39 (2014) 32–41.

[5] W. Kristjanpoller, E. Hernández, Volatility of main metals forecasted by a hybrid ann-garch model with regressors, Expert Systems with Applications 84 (2017) 290–300.

[6] P. Hajek, J. Novotny, Fuzzy rule-based prediction of gold prices using news affect, Expert Systems with Applications 193 (2022) 116487.

[7] B. Guha, G. Bandyopadhyay, Gold price forecasting using arima model, Journal of Advanced Management Science 4 (2) (2016).

[8] Z. He, J. Huang, A novel non-ferrous metal price hybrid forecasting model based on data preprocessing and error correction, Resources Policy 86 (2023) 104189.

[9] J. Zhou, Z. He, Y. N. Song, H. Wang, X. Yang, W. Lian, H.-N. Dai, Precious metal price prediction based on deep regularization self-attention regression, IEEE Access 8 (2019) 2178–2187.

[10] Y. Liu, C. Yang, K. Huang, W. Gui, Non-ferrous metals price forecasting based on variational mode decomposition and lstm network, Knowledge-Based Systems 188 (2020) 105006.

[11] G. Astudillo, R. Carrasco, C. Fernández-Campusano, M. Chacón, Copper price prediction using support vector regression technique, applied sciences 10 (19) (2020) 6648.

[12] Z. Li, Y. Yang, Y. Chen, J. Huang, A novel non-ferrous metals price forecast model based on lstm and multivariate mode decomposition, Axioms 12 (7) (2023) 670.

[13] M. Méndez-Suárez, F. García-Fernández, F. Gallardo, Artificial intelligence modelling framework for financial automated advising in the copper market, Journal of Open Innovation: Technology, Market, and Complexity 5 (4) (2019) 81.

[14] B. Shao, M. Li, Y. Zhao, G. Bian, Nickel price forecast based on the lstm neural network optimized by the improved pso algorithm, Mathematical Problems in Engineering 2019 (2019).

[15] Q. Gu, Y. Chang, N. Xiong, L. Chen, Forecasting nickel futures price based on the empirical wavelet transform and gradient boosting decision trees, Applied Soft Computing 109 (2021) 107472.

[16] C. Zhao, Y. Wang, Y. Cen, L. Wu, J. Zhou, Risk control of metal raw materials based on deep learning, Measurement and Control 55 (9-10) (2022) 1016–1030.

[17] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, et al., An image is worth 16x16 words: Transformers for image recognition at scale, arXiv preprint arXiv:2010.11929 (2020).

[18] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, S. Zagoruyko, End-to-end object detection with transformers, in: European conference on computer vision, Springer, 2020, pp. 213–229.

[19] L. Dong, S. Xu, B. Xu, Speech-transformer: a no-recurrence sequence-to-sequence model for speech recognition, in: 2018 IEEE international conference on acoustics, speech and signal processing (ICASSP), IEEE, 2018, pp. 5884–5888.

[20] D. C. Payne, J. W. Kim, Y. C. Chen, Musenet: A deep neural network for music generation, arXiv preprint arXiv:1909.09575 (2019).

[21] J. M. J. Valanarasu, P. Oza, I. Hacihaliloglu, V. M. Patel, Medical transformer: Gated axial-attention for medical image segmentation, in: Medical Image Computing and Computer Assisted Intervention–MICCAI 2021: 24th International Conference, Strasbourg, France, September 27–October 1, 2021, Proceedings, Part I 24, Springer, 2021, pp. 36–46.

[22] D. E. Rumelhart, G. E. Hinton, R. J. Williams, Learning representations by back-propagating errors, nature 323 (6088) (1986) 533–536.

[23] Y. Bengio, P. Simard, P. Frasconi, Learning long-term dependencies with gradient descent is difficult, IEEE transactions on neural networks 5 (2) (1994) 157–166.

[24] S. Hochreiter, J. Schmidhuber, Long short-term memory, Neural computation 9 (8) (1997) 1735–1780.

[25] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, Y. Bengio, Learning phrase representations using rnn encoder-decoder for statistical machine translation, arXiv preprint arXiv:1406.1078 (2014).

[26] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, I. Polosukhin, Attention is all you need, Advances in neural information processing systems 30 (2017).

[27] D. Bahdanau, K. Cho, Y. Bengio, Neural machine translation by jointly learning to align and translate, arXiv preprint arXiv:1409.0473 (2014).

[28] I. Sutskever, O. Vinyals, Q. V. Le, Sequence to sequence learning with neural networks, Advances in neural information processing systems 27 (2014).