Attention is Needed for Time Series

Jiakai Yu, Hongda Yuan, Yufan Zhang, Ziyi Wang

1 Introduction and Motivation

Time-series forecasting is a critical task across numerous domains, including finance, where the ability to predict future commodity stock prices can inform investment strategies and risk management. However, these forecasts are notoriously challenging due to the intricate temporal dependencies and volatility inherent in financial data. In this study, we focus on replicating and adapting the seminal GPT-2 model (Brown et al., 2020)—originally developed for language modeling—to the domain of financial time-series forecasting. In addition, we evaluate several other Transformer-based models, including Autoformer (Wu et al., 2021), FEDformer (Zhou et al., 2022), and Triformer (Cirstea et al., 2022), to compare their forecasting accuracy and robustness against GPT-2's performance. The Methods section details the encoder-decoder architectures of these models and their deviations from the vanilla Transformer. In the Experiments section, we thoroughly assess their predictive capabilities and explore the underlying factors driving their observed differences in performance.

2 Problem Statement

The data we are using is sourced from Kaggle [1] and consists of commodity prices, including features such as gold price, gold volume, natural gas price, natural gas volume, crude oil price, crude oil volume, copper price and copper volume. The dataset contains approximately 1,500 observations, where each observation includes multiple features relevant to forecasting gold prices.

Our objective is to reproduce GPT-2, using it to predict future gold prices using this data, and compare the performances between using different models. We will implement and compare the performance of four models: the decoder-only Transformer architecture (similar to GPT-2), Autoformer, FEDformer, and Triformer. The performance of these models will be evaluated by calculating the Mean Squared Error (MSE) for each, providing a quantitative assessment of their forecasting accuracy. And here are all the terms and (hyper) parameters we use, including the MSE formulas, as shown below.

For all the models mentioned above we conceptually define the questions to be: given the input sequences from a multivariate time series of of certain sequence length, we wish to predict the value of the response variable with horizon steps ahead. Equivalently, we wish to minimize the mean square error of the prediction outputs. Mathematically, it can be defined as the following optimization problem:

$$\min \textbf{MSE Loss} = \frac{1}{B \cdot P \cdot N \cdot F} \sum_{b=1}^{B} \sum_{p=1}^{P} \sum_{n=1}^{N} \sum_{f=1}^{F} \left(\hat{\boldsymbol{x}}_{b,t_{L+p},f_n} - \boldsymbol{x}_{b,t_{L+p},f_n} \right)^2$$

where B: Batch size (number of samples in a batch), P: Prediction horizon (number of future time steps to predict), N: Number of nodes, F: Number of features (dimensions of the multivariate data), $\hat{\boldsymbol{x}}_{b,t_{L+p},f_n}$: Predicted value for feature f in the n-th node at future time t_{L+p} for batch sample b, and $\boldsymbol{x}_{b,t_{L+p},f_n}$: Actual (ground truth) value for feature f in the n-th node at time t_{L+p} for batch sample b.

[1] https://www.kaggle.com/datasets/saketk511/2019-2024-us-stock-market-data/data

Preprint. Under review.

3 Related Work

Time-series prediction has benefited significantly from the development of transformer-based architectures. The foundational work by Vaswani et al. (Vaswani et al., 2017) introduced the transformer model with its attention mechanism, which enables effective modeling of long-range dependencies in sequential data. Brown et al. (Brown et al., 2020) extended this concept in GPT2, an autoregressive model designed for sequential prediction tasks, laying the groundwork for exploring its adaptation to time-series forecasting.

Building on this foundation, specialized models have been developed to address unique challenges in time-series data. Autoformer (Wu et al., 2021) incorporates decomposition and auto-correlation mechanisms to model temporal patterns efficiently, reducing computational complexity and overfitting. FEDformer (Zhou et al., 2022) extends Autoformer by incorporating frequency-domain decomposition to better handle long-term dependencies in periodic data. Triformer (Cirstea et al., 2022), based on the Informer model, introduces triangular attention mechanisms to manage variable-specific relationships in multivariate datasets. Together, these models demonstrate the versatility and continuous evolution of transformers to meet the specific demands of time-series forecasting.

4 Algorithm

4.1 Introduction of Overall Transformer Algorithm

4.1.1 Explanation of the Algorithm: Transformer for Time-Series Prediction

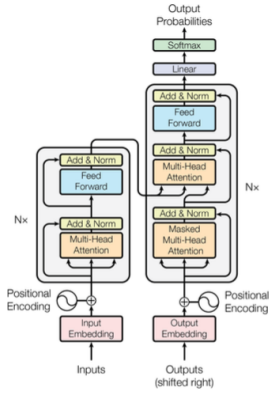

Figure 1: Attention visualization from Vaswani et al., 2017.

The **Transformer** architecture, introduced in *"Attention Is All You Need"*, relies entirely on attention mechanisms to process sequential data. Its design eliminates recurrence and convolution layers, making it computationally efficient and highly parallelizable. For time-series prediction, the Transformer architecture can be adapted to capture temporal dependencies and model long-range interactions effectively.

The Transformer consists of an encoder-decoder structure, where the encoder maps an input sequence of time-series data into a sequence of continuous latent representations, and the decoder processes these latent representations to generate predictions for the target sequence. A key component of the Transformer is the self-attention mechanism, specifically the scaled dot-product attention, which models dependencies between different positions in a sequence using query (Q), key (K), and value (V) matrices. This is calculated as:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V.$$

In time-series forecasting, the Query represents the specific feature or timestep the model focuses on for prediction, while the Keys represent all timesteps in the sequence used as a reference for

comparison. The Values contain the actual data associated with each Key. To enhance its ability to capture complex patterns, the Transformer uses multi-head attention, allowing it to focus on different subspaces of the input features simultaneously. Additionally, position-wise feed-forward networks improve the model's representation capacity, while positional encoding compensates for the lack of inherent temporal order in the architecture by embedding sequence information directly into the input.

To adapt the Transformer for time-series forecasting, the input data is structured as $\mathbf{X} \in \mathbb{R}^{T \times F}$, where T is the number of timesteps and F is the number of features. The target sequence for prediction is represented as $\hat{\mathbf{Y}} \in \mathbb{R}^{P \times F}$, where P is the prediction horizon. The self-attention mechanism is particularly effective for capturing long-term temporal dependencies, while the multi-head attention ensures that cross-variable interactions in multivariate time-series are adequately modeled.

For preprocessing, the data must be cleaned by addressing missing values through imputation or interpolation and normalized using techniques like Standard scaling to ensure numerical stability during training. Sliding windows are generated to create input-output pairs from historical data. Post-processing involves applying inverse transformations, such as de-normalization, to make the predicted values interpretable. Additionally, evaluation metrics like Mean Squared Error (MSE) or Mean Absolute Error (MAE) are employed to assess the model's performance.

Transformers offer several advantages for time-series forecasting. Unlike RNNs, they allow for parallel processing, which speeds up training and inference. Their self-attention mechanism enables the model to capture dependencies over long time horizons, and their scalability makes them suitable for large datasets and long sequences. This work is particularly important because it adapts the powerful GPT-2 model, a general-purpose sequence model, to the unique requirements of time-series forecasting. By comparing it with state-of-the-art models like Autoformer, FEDTransformer, and Triformer, this project provides valuable insights into the strengths and limitations of general-purpose and specialized architectures, addressing challenges such as modeling temporal dynamics, learning cross-variable relationships, and efficiently handling long sequences.

4.2 Decoder-only Transformer

The GPT architecture is a decoder-only transformer, which means only the causally-masked attention layers and MLP layers are used for directly transforming the input vector to the output. We modified the GPT2 model by replacing the regular token embedding part with a linear transformation from the feature space to embedding space, and adding a linear transformation as the last layer to output the predicted price. The decoder-only Transformer is configured with a maximum sequence length of 1024, 12 transformer blocks, 12 attention heads, an embedding size of 768, and 2 input features. For data preprocessing, we employed cross-sectional standardization to normalize the input data. Predicting stock prices using time series data is challenging due to complex temporal dependencies, non-stationarity, and inherent noise in financial markets.

4.3 Autoformer: Decomposition and Auto-Correlation Mechanisms

The Autoformer (Wu et al., 2021) is designed to address the challenges of long-term time-series forecasting by introducing two key innovations: a progressive decomposition architecture and an Auto-Correlation mechanism. The decomposition architecture separates input time-series data into trend and seasonal components as $X_t = X_{\text{trend},t} + X_{\text{seasonal},t}$, progressively refining these patterns at each layer during training and inference. The seasonal component is further segmented into sub-sequences, enabling the model to focus on localized patterns and compute attention across these sub-sequences, better capturing long-term trends and periodic variations. The Auto-Correlation mechanism, inspired by stochastic process theory, replaces traditional self-attention by identifying and leveraging periodic dependencies in time-series data. It calculates the auto-correlation function $R_X(\tau) = \frac{\gamma_X(\tau)}{\gamma_X(0)}$, where $\gamma_X(\tau)$ is the auto-covariance and $\gamma_X(0)$ is the variance of the series, to quantify the similarity between sub-sequences at different time lags τ, to quantify the similarity between sub-sequences at different time lags, focusing on the top-k lags with the highest correlation. These significant lags are aggregated through time-delay operations such that Aggregated Output $= \sum_{i=1}^{k} \text{Roll}(V, \tau_i) \cdot \text{Auto-Correlation}(Q, K, \tau_i)$, aligning and combining the most relevant sub-sequences. This mechanism achieves efficient computation with $O(L \log L)$ com-

plexity by leveraging Fast Fourier Transform (FFT), enabling the model to handle long sequences effectively.

The Autoformer employs an encoder-decoder structure, where the encoder models seasonal components, and the decoder progressively refines trend information to generate accurate predictions. Preprocessing steps include normalization of input data, handling missing values, and creating sliding windows for training, while post-processing involves de-normalizing predictions and evaluating performance using metrics like Mean Squared Error (MSE) and Mean Absolute Error (MAE). By capturing intricate temporal patterns and long-term dependencies with computational efficiency, the Autoformer is well-suited for real-world applications such as energy, traffic, and weather forecasting. Its ability to outperform traditional and Transformer-based models highlights its significance in advancing time-series forecasting methodologies, making it a robust solution for long-term sequence modeling challenges.

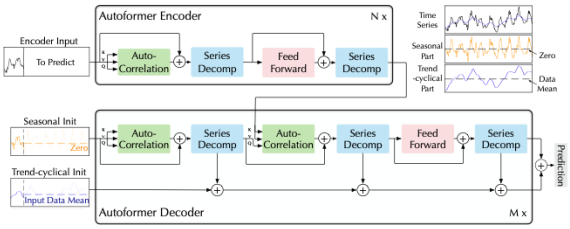

Figure 2: Autoformer visualization from Wu et al., 2021.

4.4 FEDformer: Seasonal Trend Decomposition and Fourier-Enhanced Attention

The limitations of the vanilla Transformer in time series forecasting stem from its point-wise attention mechanism and independent timestep predictions, which often fail to capture the global properties and overall trends of the time series as a whole. To address this, FEDformer (Zhou et al., 2022) introduces two key ideas. The first idea is to incorporate a seasonal-trend decomposition approach, using a Frequency Enhanced Decomposed Transformer architecture with a mixture of experts. This explicitly separates global properties such as trends and seasonality from local variations, enabling the model to better align the predicted and ground truth distributions. The second idea is to combine Fourier analysis with the Transformer-based method. Exploiting the fact that time series tend to have (unknown) sparse representations on a basis like the Fourier basis, FEDformer applies the Transformer to the frequency domain instead of the time domain. By selecting a random subset of frequency components—spanning both low and high frequencies—this model helps the Transformer better capture the global (both seasonal and trend) properties of time series.This design not only enhances the ability of FEDformer to capture global characteristics but also reduces the computational complexity of the Transformer from quadratic to linear, making it more efficient for long-term forecasting tasks.

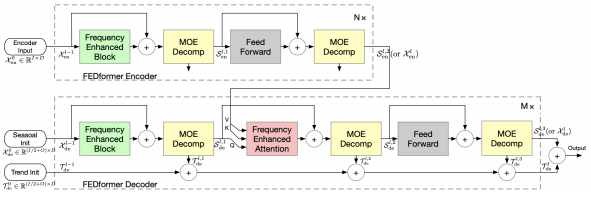

Figure 3: FEDformer visualization from Zhou et al., 2022.

By incorporating the two ideas discussed earlier, we redesign the encoder and decoder structure of the model. Due to space limitations, we focus on introducing the Frequency Enhanced Block (FEB), the most critical component of both the encoder and decoder. The input $x \in \mathbb{R}^{N \times D}$ is first linearly projected using a weight matrix $W \in \mathbb{R}^{D \times D}$, resulting in $q = x \cdot W$. This projected input q is then transformed from the time domain to the frequency domain using the Fourier transform, denoted as $\mathcal{F}(q) = Q \in \mathbb{C}^{N \times D}$. In the frequency domain, a select operator retains only M

modes, with $M \ll N$, producing $\widetilde{Q} \in \mathbb{C}^{M \times D}$. This selection significantly reduces computational complexity while preserving the most critical components of the signal. Next, a parameterized kernel $R \in \mathbb{C}^{D \times D \times M}$ is applied to the selected modes, performing element-wise multiplication $\widetilde{Q} \circ R$. The kernel R serves as a flexible and learnable transformation tool that (1) models interactions between input and output channels ($D \times D$), (2) adapts transformations to each selected frequency mode (M), and (3) enhances the ability to capture both global and local patterns in the frequency domain. The result of this operation is padded back to $\mathbb{C}^{N \times D}$ and transformed back to the time domain using the inverse Fourier transform. By focusing on meaningful frequency components and leveraging efficient computation, this process enhances the model's ability to capture the global properties of time series data while maintaining computational efficiency.

4.5 Triformer: Variable-Specific Patch Attention for Multivariate Time Series

To enhance the ability to model multivariate time series data, the Triformer architecture proposed by Cirstea et al. (Cirstea et al., 2022) introduces a novel mechanism called triangular, variable-specific patch attention. This architecture is designed with a hierarchical structure where the input size progressively decreases at deeper layers in attention module, achieving a computational complexity that scales linearly with respect to the sequence length.

Each variable i (e.g., stock indicators such as price, volume, etc.) is assigned specific parameters (weight matrices, specifically, memory matrices mentioned below), enabling the model to capture variable-specific temporal dependencies effectively. The triangular attention mechanism refers to the arrangement of attention layers in a hierarchical manner. Lower layers operate on smaller patches of the time series, denoted as $x_p^{(i)}$ for the p^{th} patch of variable i, capturing finer-grained information, while upper layers process larger patches, summarizing broader temporal patterns. To ensure compatibility, the number of patches P in consecutive layers must successively divide the input sequence length L.

The variable-specific attention mechanism is implemented by factorizing the weight matrices in fully connected layers into three distinct components: L: A shared projection matrix (on left) that maps the input dimension to a memory dimension, R: Another shared matrix (on right) for projecting the memory back to the output dimension, and $M^{(i)}$: A variable-specific memory matrix unique to each variable i.

These matrices dynamically generate the attention weights $\boldsymbol{W}^{(i)} = \boldsymbol{L}\boldsymbol{M}^{(i)}\boldsymbol{R}$. This design reduces the parameter count while tailoring memory representation for each variable. Moreover, the shared $\boldsymbol{L}$ and $\boldsymbol{R}$ matrices allow information to flow across variables, enhancing representation capacity.

The patch attention mechanism divides the time series into smaller patches, with attention applied only within patches. While this localized attention improves efficiency and fine-grained pattern detection, it sacrifices direct connections between patches. To mitigate this, the Triformer introduces a recurrent relation and gating procedure, which restore the links between patches within each triangular attention layer.

Lastly, the decoder structure is not explicitly defined in the original paper. However, based on our analysis of the architecture, we identify the final forward propagation through fully connected layers, including the skip connection that aggregates outputs from triangular attention layers, as the decoder.

For reasons of experimental rigour as well as space considerations, Triformer's experimental data were treated in the same way as the previous model, as detailed above.

5 Experiments

We successfully reproduced the published results for the Autoformer and FEDformer models using the ETTm1 dataset, a widely recognized benchmark for time-series forecasting. By carefully implementing these models and aligning the hyperparameter configurations with those specified in their respective papers, we obtained results consistent with the original publications. This confirmed the validity of Autoformer and FEDformer on the ETTm1 dataset and demonstrated their effectiveness in handling long-term temporal dependencies and complex patterns. For the GPT-based model, since we modified the original version to handle time-series sequences, we only trained and evaluated this model on the stock dataset we selected. The Triformer model was also trained and tested on stock

dataset, ETTm1, and ETTh1, however the results didn't appear to be ideal (will explain in Conclusion section). These adaptations allowed us to assess the performance of the GPT-based and Triformer models in the context of our selected dataset, despite the constraints on reproducibility.

For the purpose of our project, we tested all models on a stock market dataset, where the objective was to predict gold prices using eight extracted features as mentioned in the problem statement section. The application of these models to this new dataset provided valuable insights into their robustness in handling diverse time-series data.

To evaluate the performance of the models, we used Mean Squared Error (MSE) and Mean Absolute Error (MAE) as the primary metrics. These standard measures of forecasting accuracy enabled clear and objective comparisons between the models. For hyperparameter tuning, we focused on sequence length, label length, and prediction length. After testing various combinations, we set the parameters to 30, 15, and 7, respectively. This configuration represented one month of historical data for learning trends, half a month for capturing recent information, and one week for predictions. This balance ensured that the models captured both long-term and short-term dependencies efficiently.

The results of the experiments highlighted the strengths and limitations of the models. On the ETTm1 dataset, FEDformer achieved the best performance, with slightly lower MSE and MAE compared to the others, likely due to its frequency decomposition mechanism designed for periodic time-series. On the stock dataset, Autoformer and FEDformer delivered comparable results, with FEDformer showing a slight advantage in MSE. This indicates that FEDformer is robust and adaptable to new datasets, while Autoformer also demonstrated strong performance. Triformer maintained competitive results on both datasets but lagged slightly behind, possibly due to its triangular attention mechanism being less suited for the stock dataset's noisier structure.

	GPT2-based	Autoformer	FEDformer	Triformer
MSE	0.9547	0.0624	0.1328	1.0390

Table 1: Results of Models

Despite achieving comparable results, the process of reproducing the results highlighted the sensitivity of these models to hyperparameters and preprocessing techniques. For instance, inconsistencies in data scaling or batch size configuration could noticeably affect performance. The stock dataset also presented challenges due to its noisier nature compared to ETTm1, requiring additional tuning to achieve optimal results. Nonetheless, these experiments validate the capability of the Transformer models to handle both benchmark datasets and novel, domain-specific data effectively.

6 Conclusions

The FEDformer methods we applied significantly improved accuracy, with the Mean Squared Error (MSE) reducing by 86.18% compared to the baseline decoder-only model. However, contrary to expectations, FEDformer underperformed Autoformer in our experiments, with Autoformer achieving an MSE of 0.0624—nearly half that of FEDformer. This discrepancy may be due to differences in the datasets: unlike the ETTm1 dataset with 15-minute intervals, our dataset consists of daily data, which may favor Autoformer's focus on top-k sub-sequence correlations. In contrast, FEDformer's global frequency decomposition might be less effective for datasets dominated by local patterns or short-term dependencies. Additionally, FEDformer's computation across all frequency components may have caused overfitting to noise, particularly in a dataset with limited frequency-domain information. This highlights the importance of aligning models with dataset characteristics and tuning parameters for optimal performance.

For the Triformer experiments, despite scaling the dataset size and improving the preprocessing steps, the test loss remained significantly higher than the training loss in all experiments. This suggests that the model may be overfitting the training data. Unlike models with full training configurations, our training phase from scratch lacks overfitting detection, so early stopping cannot be implemented. Even when we reduced the model parameters, increased the dropout rate and replaced the dataset with a larger one, we were still unable to avoid overfitting. We believe that Triformer, which has more than 100,000 parameters (using the same hyperparameters) according to the ablation study in the paper, may explain why our model underperforms even though the ETTm1 dataset has around 60,000 data.

7 Contributions

Yufan Zhang is responsible for the algorithm section focusing on the Decoder-only Transformer, including reproducing the decoder code, which serves as the model for our reproduction. Ziyi Wang is responsible for the mathematical aspects of the problem statement, the Autoformer section in the algorithm part, the experiments section, and the evaluation of Autoformer using our dataset. Jiakai Yu is responsible for the introduction and motivation, the FEDformer section in the conclusion, and the evaluation of FEDformer using our dataset. Hongda Yuan is responsible for the conceptual aspects of the problem statement, the Triformer section in the conclusion, and the evaluation of Triformer using our dataset.

References

[1] Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D., Wu, J., Winter, C., ... Amodei, D. (2020). *Language models are unsupervised multitask learners.* OpenAI.

[2] Cirstea, R.-G., Guo, C., Yang, B., Kieu, T., Dong, X., & Pan, S. (2022). Triformer: Triangular, variable-specific attentions for long sequence multivariate time series forecasting. *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*.

[3] Hyndman, R. J., & Khandakar, Y. (2008). Automatic time series forecasting: The forecast package for R. *Journal of Statistical Software*, 27(3), 1–22.

[4] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., & Polosukhin, I. (2017). Attention is all you need. *Advances in Neural Information Processing Systems*, 30.

[5] Wu, H., Xu, J., Wang, J., & Long, M. (2021). Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting. *Advances in Neural Information Processing Systems*, 34.

[6] Zhou, T., Ma, Z., Wen, Q., Wang, X., Sun, L., & Jin, R. (2022). *FEDformer: Frequency enhanced decomposed transformer for long-term series forecasting.* Proceedings of the 39th International Conference on Machine Learning (ICML 2022), 27268–27286. https://proceedings.mlr.press/v162/zhou22g.html