AIMS Mathematics, 10(11): 27152–27170.
DOI: 10.3934/math.20251193
Received: 09 July 2025
Revised: 11 October 2025
Accepted: 13 October 2025
Published: 21 November 2025

https://www.aimspress.com/journal/Math

Research article

Transformers vs. LSTM-MLP for option pricing

Boye A. Høverstad[1], **Morten Risstad**[2,*] **and Lavrans K. Sagen**[1]

[1] Norwegian University of Science and Technology, Department of Computer Science, Trondheim, Norway

[2] Norwegian University of Science and Technology, Department of Industrial Economics and Technology Management, Trondheim, Norway

* **Correspondence:** Email: morten.risstad@ntnu.no; Tel: +47-97166263.

Abstract: In the realm of option pricing, parametric models originating from the Black-Scholes-Merton framework have proven extremely persistent. However, machine learning models have recently entered the field with success, arguably due to their flexible and non-parametric nature. A combined LSTM-MLP deep learning architecture that combines time series data with cross-sectional pricing information, avoiding explicit volatility estimates, has recently been proposed. This LSTM-MLP model outperforms relevant benchmarks in different dimensions. In this research, we investigated whether a transformer-based alternative is able to better capture the inter-temporal characteristics of the data than the LSTM-based LSTM-MLP model. We found that although the transformer performs better during the extreme market conditions of COVID-19, the LSTM-MLP architecture is overall superior.

Keywords: option pricing; deep learning; LSTM-MLP; transformers
Mathematics Subject Classification: 91G60, 68T07

1. Introduction

The subject of valuing optionality is important in numerous applications in economics and finance. Real options exist in forms of investment timing (option to wait), growth opportunities (option to expand), abandonment options, and flexibility (switching options). Financial options, in the form of financial contracts, constitute a major part of the derivatives markets. Financial options are written on a wide range of underlyings, including commodities, equities, foreign exchange, and bonds. Options are traded by both speculators and hedgers, either through regulated marketplaces such as exchanges or bilaterally. Since the mid-1970s, option pricing has been dominated by parametric models [1–4]. More recently, supervised machine learning models have been employed by [5–9], among others.

Arguably, the success of these models derives from the highly flexible and non-parametric nature of deep neural networks. The increasing focus on option pricing reflects investor needs for making informed investment decisions. More accurate forecasts of option prices are beneficial for buyers and sellers, and also for financial intermediaries, which provide liquidity to well-functioning markets.

The approach recently taken by [10] is particularly interesting. [10] combine two artificial neural networks: a long short-term memory (LSTM) and a multilayer perceptron (MLP). This novel LSTM-MLP architecture combines time series data with cross-sectional pricing information and is motivated by the hypothesis that the time series information extracted by the LSTM is more informative for option pricing than the explicit volatility input commonly used in the literature. Therefore, they replace the explicit volatility input to the MLP with extracted time series information conveyed via the LSTM outputs. First, time series data in the form of historical underlying returns is provided as input to the LSTM network. Subsequently, the output of the LSTM is used as input in the MLP, along with the option features for a given contract, to determine the option price. Through extensive benchmarking and performance evaluation out-of-sample using SPX European call options on the S&P 500 index from November 1, 2011, to December 31, 2022, [10] finds found that the merged LSTM-MLP achieves better pricing and trading performance than relevant parametric and machine learning benchmark models.

The performance of the LSTM-MLP model hinges on the LSTM component's ability to accurately capture the time-series dynamics of S&P 500 option prices. An alternative approach to sequential processing is the attention mechanism introduced by [11], further popularized through the transformer, proposed by [12]. According to a recent survey by [13], a number of well-performing extensions of the original transformer architecture have been developed in the time-series domain. Still, the relevance of transformers for time series modeling in finance is somewhat unclear. As far as we know, this paper is the first to directly contrast the LSTM-MLP model of [10] with a comparable temporal fusion transformer (TFT). Our overall finding is that, although the TFT performs better during the extreme market conditions of COVID-19, the LSTM-MLP architecture is still superior.

2. The temporal fusion transformer

The TFT model investigated in this paper is based on the original model proposed by [14]. We apply a few modifications to accommodate the problem at hand, as illustrated in Figure 1. In the following we outline the main elements of the architecture. Further details can be found in Appendix A.

All input variables are transformed linearly using learned embeddings to be of dimension d_{model} before they are processed by the model. The embeddings are learned in conjunction with all other model parameters during training. The first step in model inference is static variable selection using a variable selection network. The static variables are then encoded with a static covariate encoder to produce four context vectors: $c_s, c_c, c_h,$ and c_e. Next, the input sequence is run through gated residual networks (GRN) with the shared weights across the entire sequence. Each GRN takes c_s as the context vector. The sequence is then encoded using an LSTM with cell and hidden state initialized by c_c and c_h respectively. Each encoded timestep is then run through a gated linear unit (GLU), added with its corresponding residual LSTM input, and run through layer normalization [15]. We will refer to this part of the model as the encoder with its respective static and temporal submodules.

The sequence is then run through another layer of GRNs with c_e as the context vector and shared

weights across all GRNs in the layer. We will call this layer of GRNs the "static enrichment layer" as it corresponds to the layer with the same name in the original TFT architecture. After static enrichment, we add interpretable multi-head attention followed by a GLU, the addition of the residual attention input, and layer normalization is applied to the sequence.

Finally, the sequence is run through a layer of GRNs with shared weights, followed by a layer of instance-wise gating using GLUs, also with shared weights, the addition of the static enrichment input residual, and layer normalization [15]. The final token in the sequence thus far is then run through a liner layer with a ReLU activation to produce a single value output. The final ReLU activation function is there to avoid nonsense negative value predictions during evaluation but is not present during training to avoid zero-value gradients.

Dropout is applied immediately after the LSTM encoding and attention during training and is marked with a red dot in the proposed model illustration seen in Figure 1.

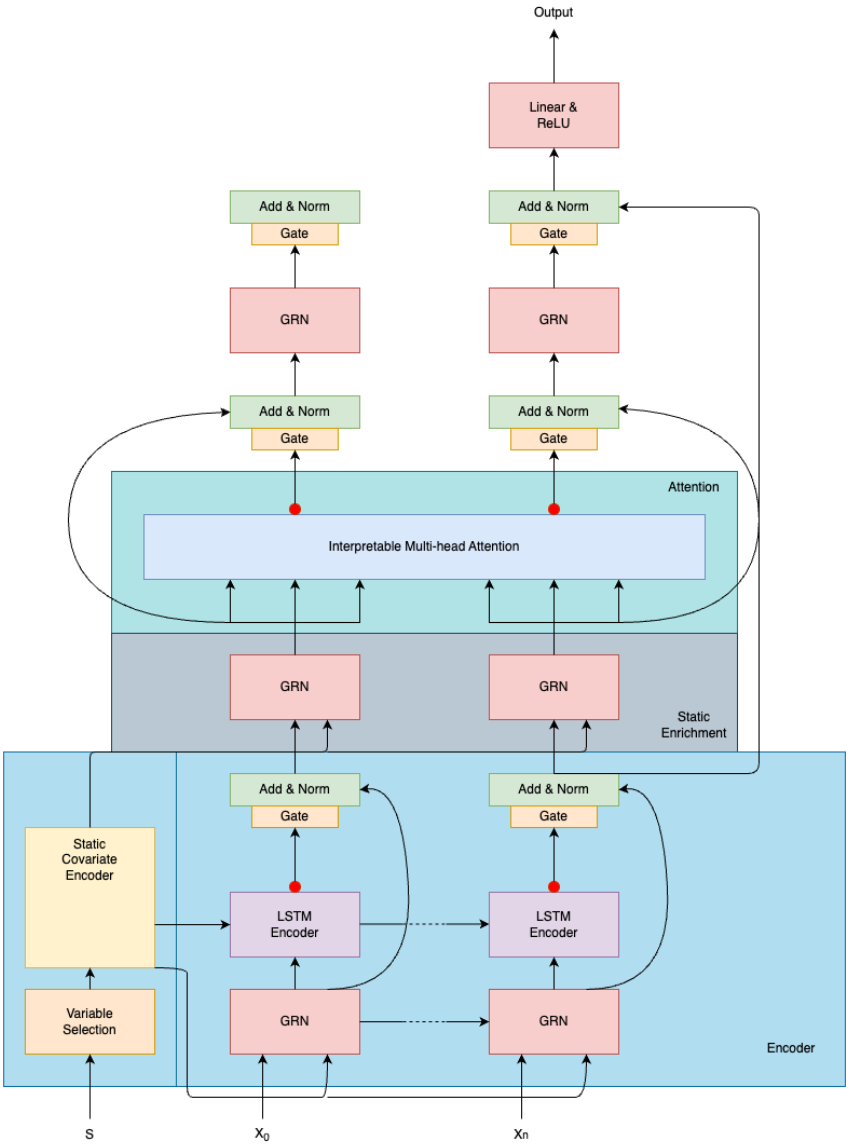


Figure 1. Illustration of the implemented temporal fusion transformer model.

3. Empirical setup

Our empirical setup mirrors that of [10] as closely as possible, except that we extend the sample period to cover January 2011 up to and including September 2023.* For details, we refer to [10], but repeat the most important points here for convenience.

The dataset contains daily end-of-day quotes for SPX index options from January 2011 up to and including September 2023. During preprocessing, any data points with missing values are removed. Additionally, moneyness and time to maturity (TTM) are calculated for each entry. Any data points with a moneyness value outside the range 0.8–2.0 are discarded as outliers tend to be less accurately priced and hence negatively impact performance for all models. Further, any data point with a TTM equal to zero or beyond two years is discarded as the option value is equal to its intrinsic value at expiry, and options with longer maturities also tend to be less accurately priced. This results in approximately 12% of the data being discarded. Descriptive statistics before and after preprocessing can be seen in Table 1 and histograms of the preprocessed data over moneyness and TTM can be seen in Figure 2. Table 2 contains descriptive statistics for explanatory variables.

We adopt the sliding window validation technique proposed by [10]. Each subset consists of 3 years of training data, 1 month of validation data, and 1 month of test data. The first subset of data begins in January 2011 and the out-of-sample test set is the month of February 2014. We then slide this window forward by 1 month to produce each subsequent subset, resulting in out-of-sample test sets for all data points ranging from February 2014 up to and including September 2023. For the benchmark MLP and LSTM-MLP models, we use the same hyperparameters as [10].

Table 1. Descriptive statistics of option premia.

	Moneyness	TTM
count	14,441,679	14,441,679
mean	1.296875	0.331734
std	1.223538	0.521102
min	0.315254	0.000000
25%	0.976729	0.051941
50%	1.079817	0.134247
75%	1.271240	0.369863
max	47.955700	5.493265

(a) Dataset before preprocessing.

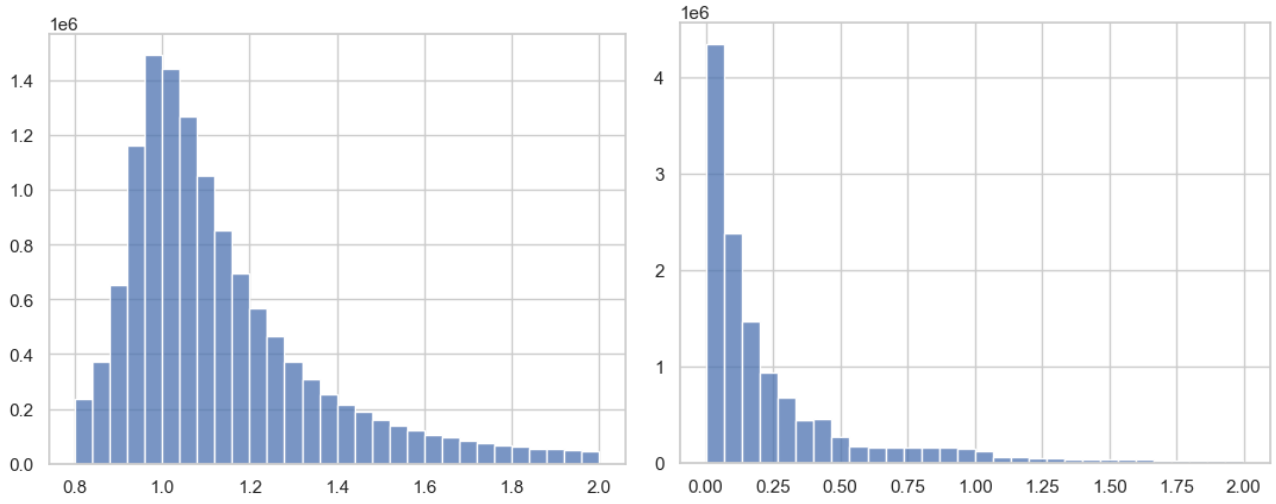
	Moneyness	TTM
count	12,689,290	12,689,290
mean	1.132876	0.254438
std	0.224140	0.337803
min	0.800003	0.002740
25%	0.980477	0.049315
50%	1.070773	0.120548
75%	1.220735	0.306849
max	1.999990	1.997260

(b) Dataset after preprocessing.

*The dataset is publicly available at optionsDX `https://www.optionsdx.com/product/spx-option-chain/` and contains daily end-of-day quotes for the SPX index from January 2011 up to and including September 2023.

Table 2. Descriptive statistics for explanatory variables.

	mean	std
Strike ($)	2817.77	1039.35
Price ($)	362.08	399.60
TTM (years)	0.2552	0.3392
Risk Free Rate (%)	1.0617	1.4586
30–day RV	0.0095	0.0062
Moneyness	1.1356	0.2254
S&P 500 Returns (%)	0.0442	1.0745



(a) Moneyness **(b)** Time to maturity

Figure 2. Histograms over moneyness and time to maturity time to maturity (TTM) after preprocessing. The dataset is centered around at-the-money options with short TTMs.

For the MTFT we follow [16] and conduct a random hyperparameter search to obtain a sensible hyperparameter configuration. The search is conducted as a weights and biases sweep[†] This hyperparameter search space is defined in Table 3. The final hyperparameters used in our empirical analysis are listed in Table 4.

[†]`https://docs.wandb.ai/guides/sweeps` with 200 runs. Due to computational limitations, the search was run once on a subset of the data with training data from January 2012 up to and including December 2014 and a held-out validation set consisting of data from January 2015 and the same hyperparameter configuration is then applied across all subsets of the data.

Table 3. TFT hyperparameter search space.

Parameter	Values
Learning Rate	0.0005–0.5
Gamma	0.975–1.0
Dropout	0.05–0.3
Attention Heads	1, 2, 3, 4
LSTM Layers	1, 2, 3, 4
Embedding Dimensions	4, 8, 12, 16
Timesteps	100, 140, 200, 230

Table 4. TFT hyperparameters.

	Parameter	Value
Optimization algorithms	Optimizer	Adam
	LR Scheduler	Exponential
Optimization parameters	Learning Rate	0.0028
	Gamma	0.965
Model parameters	Dropout	0.13
	Embedding Dimensions	64
	Timesteps	140
	Attention Heads	4
	LSTM Layers	2

4. Results and discussion

The LSTM-MLP has a slight edge over the TFT, and the TFT clearly achieves the second-best performance.[‡] The TFT seems to outperform the LSTM-MLP under specific conditions and somewhat overfit under others. The performance of each model, aggregated over all windows, in terms of RMSE, MAE, and ME, is listed in Table 5.

Table 5. RMSE, MAE, and ME, over the entire test set from February 2014 to September 2023. The best result for each metric is highlighted in bold.

	RMSE	MAE	ME
BS	53.36	27.13	-23.66
MLP	14.94	8.27	-1.21
LSTM-MLP	**11.89**	**6.61**	0.72
TFT	12.77	7.16	**-0.38**

[‡]Unreported Diebold-Mariano tests confirm that these results are statistically significant.

4.1. Results by year

We aggregate the predictions by year and measure overall performance in terms of RMSE, MSE, and ME per year for each model, listed in Tables 6–8, respectively. The non-parametric approaches dominate over all years with the LSTM-MLP achieving the best performance most years. A notable exception is 2020, a year characterized by a market shock with large negative returns due to the COVID-19 pandemic, where the TFT achieves the best RMSE and MSE, but with a tendency to undervalue options as indicated by the ME.

Table 6. RMSE aggregated over years for each model. The best result for each year is highlighted in bold.

Year	BS	MLP	LSTM-MLP	TFT
2014	19.77	6.64	4.92	**3.82**
2015	21.47	8.70	4.42	**4.30**
2016	20.78	6.68	**3.60**	6.54
2017	17.57	4.01	**3.94**	5.89
2018	26.33	11.18	**5.60**	10.95
2019	28.04	7.89	**6.51**	8.60
2020	64.75	30.32	23.84	**22.51**
2021	68.90	14.77	**12.15**	13.27
2022	85.78	16.00	**13.96**	15.72
2023	54.19	9.48	**9.39**	10.18

Table 7. MAE aggregated over years for each model. The best result for each year is highlighted in bold.

Year	BS	MLP	LSTM	TFT
2014	10.35	4.18	3.37	**2.45**
2015	11.74	5.00	**2.96**	3.09
2016	10.72	4.36	**2.57**	3.75
2017	8.23	2.93	**2.90**	3.76
2018	14.12	7.60	**3.88**	6.29
2019	15.42	5.49	**4.58**	4.66
2020	38.67	17.23	14.62	**13.26**
2021	38.20	9.74	**7.52**	8.49
2022	52.73	10.85	**9.61**	10.50
2023	30.98	6.32	**6.09**	6.71

Table 8. ME aggregated over years for each model. The best result for each year is highlighted in bold.

Year	BS	MLP	LSTM	TFT
2014	-4.68	-1.53	-1.34	**-0.90**
2015	-6.85	-1.59	**0.03**	0.23
2016	-5.54	1.64	**-0.11**	-1.10
2017	-3.35	-0.37	**0.01**	0.54
2018	-9.66	1.60	**0.94**	2.27
2019	-9.09	-0.32	**-0.23**	-2.26
2020	-36.19	-5.15	**0.16**	-2.22
2021	-36.61	-4.87	**0.63**	-1.15
2022	-51.61	1.49	3.21	**-0.70**
2023	-28.48	**-0.91**	1.36	2.18

The prediction errors over years per model are visualized in Figure 3. It interesting that the TFT and LSTM-MLP are seemingly affected similarly by different regimes except in 2020, when the market crashed. In 2020, the LSTM-MLP overvalues options, which makes sense given that the values are likely lower compared to what was seen in the training data. However, the TFT actually undervalues the options for this year, possibly indicating an enhanced ability to model regime changes.

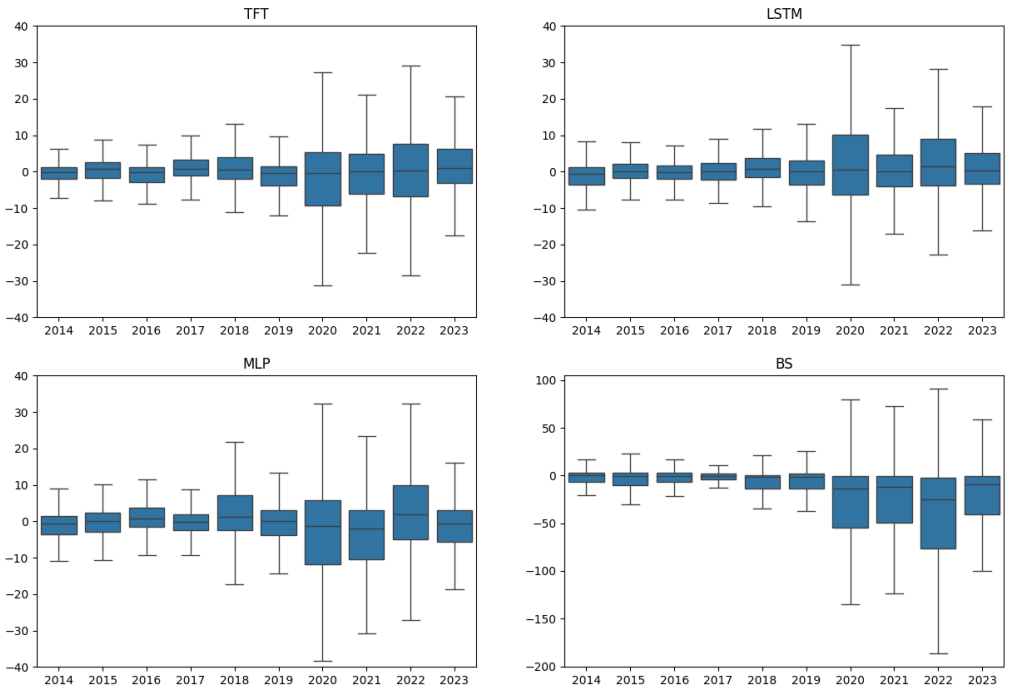


Figure 3. Boxplots of error terms per model aggregated over years. The line in the middle of the box represents the median, the box represents the quartiles, and the whiskers mark 1.5 times the interquartile range beyond the quartiles. Note that the BS boxplot has a different y-axis scale as its errors are larger.

4.2. *Results by moneyness and time to expiry*

The LSTM-MLP achieves the best performance over most of the moneyness values with the MLP achieving comparable results for options with a moneyness value above 1.4 and the TFT achieving comparable results for at-the-money options. Interestingly, the TFT's performance deteriorates rapidly as moneyness increases. Given the large number of at-the-money options in the dataset compared to in-the-money options this might indicate that the TFT is overfitting to the at-the-money data points and is unable to extrapolate to data points with higher moneyness values. A plot of the RMSE over different levels of moneyness is available in Figure 4.

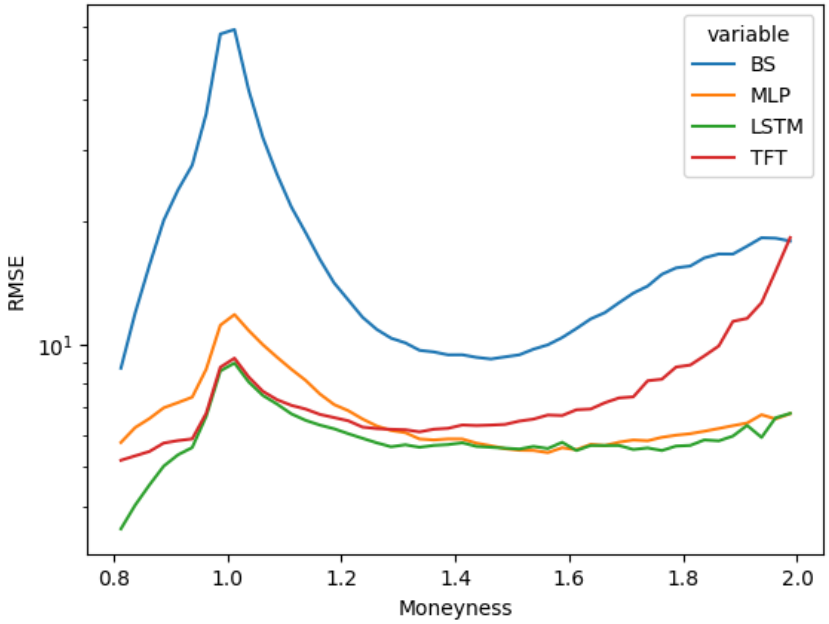


Figure 4. Prediction error in terms of RMSE over different levels of moneyness. The LSTM-MLP and the TFT achieve comparable results for at-the-money options. However, the TFT's performance rapidly deteriorates as moneyness increases. Note that the y-axis is log-scaled.

All models perform better for shorter TTMs, in general. This is natural as the valuation of an option is tightly coupled with the uncertainty of the underlying asset's value at expiration. The RMSE over different TTMs are plotted in Figure 5.

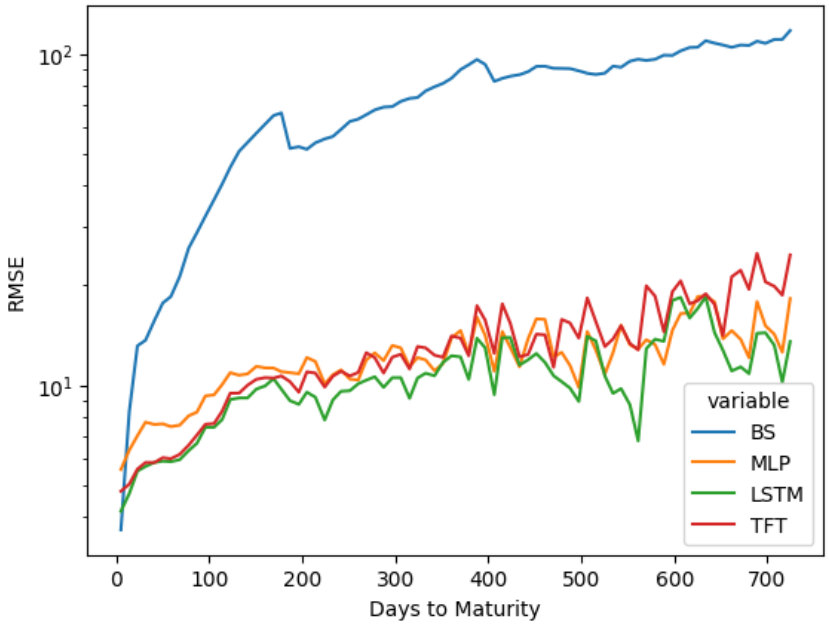


Figure 5. Prediction error in terms of RMSE over TTM. The LSTM-MLP and the TFT achieve comparable results for options close to maturity with the TFT's performance deteriorating more rapidly as TTM increases. Note that the y-axis is log-scaled.

An interesting observation is the fact that the TFT and LSTM-MLP achieve nearly identical results for options with a TTM below (approximately) 150 days, with the TFT's performance decreasing more rapidly than the LSTM-MLP's beyond this point. As with moneyness, the TFT achieves comparable performance for commonly seen data points and underperforms when exposed to outliers. We interpret this as further evidence of a tendency to overfit.

4.3. Prediction error

When forecasting financial instruments, the sign of the prediction error is important. Thus, in this section we investigate the error terms $e_i = \hat{y}_i - y_i$, where $\hat{y}_i$ is the predicted value for option i and y_i is the actual market value for option i. In other words, a negative error term means that the model undervalues the option, while a positive error term indicates that the model overvalues the option.

The results indicate that all non-parametric approaches tend to average out close to zero. Further, all models, including BS, have a median value tendency to zero. The non-parametric approaches, however, are less volatile in their prediction errors. Interestingly, BS performance seems to deteriorate rapidly from 2020 onward with a tendency to undervalue options. This is indicative of a regime change, which in turn appears to be better captured by the non-parametric models. However, their performance is also negatively impacted, implying that the market conditions are more challenging to model post-COVID-19. This is consistent with [10].

We also find it interesting that the TFT and LSTM-MLP are seemingly affected similarly by different regimes except in 2020, when the market crashed. In 2020, the LSTM-MLP overvalues options, which makes sense given that the values are likely lower compared to what was seen in the training data. However, the TFT actually undervalues the options for this year, possibly indicating an enhanced

ability to model regime changes. Descriptive statistics for the prediction errors of the LSTM-MLP and TFT in 2020 are listed in Table 10. Descriptive statistics over all test sets are listed in Table 9.

Table 9. Prediction error statistics aggregated over all test periods from February 2014 to September 2023 for all models.

	BS	MLP	LSTM	TFT
Mean	-23.66	-1.21	0.72	-0.38
Standard Deviation	47.83	14.89	11.87	12.76
Quartile 1	-29.69	-4.91	-3.04	-3.82
Median	-3.90	-0.06	0.13	0.23
Quartile 3	0.31	4.03	4.03	3.54

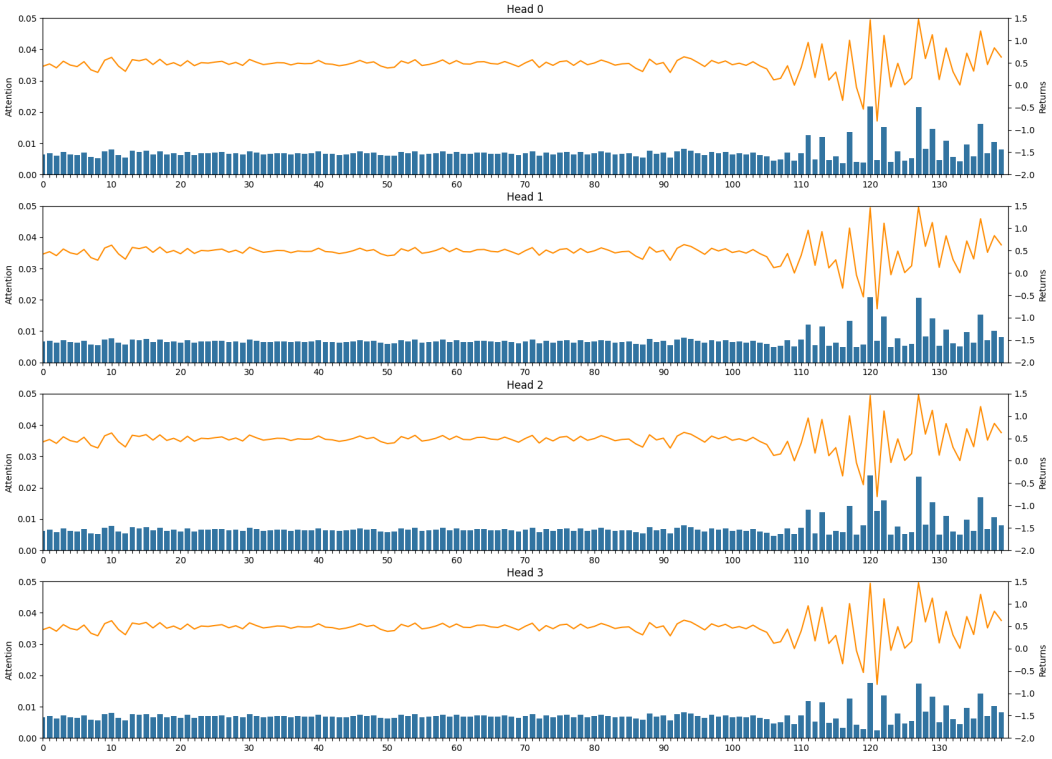
Table 10. Prediction error statistics over the year 2020 for the LSTM-MLP and the TFT.

	LSTM	TFT
Mean	0.16	-2.22
Standard Deviation	23.84	22.40
Quartile 1	-6.33	-9.37
Median	0.59	-0.29
Quartile 3	10.08	5.25

4.4. Attention activations

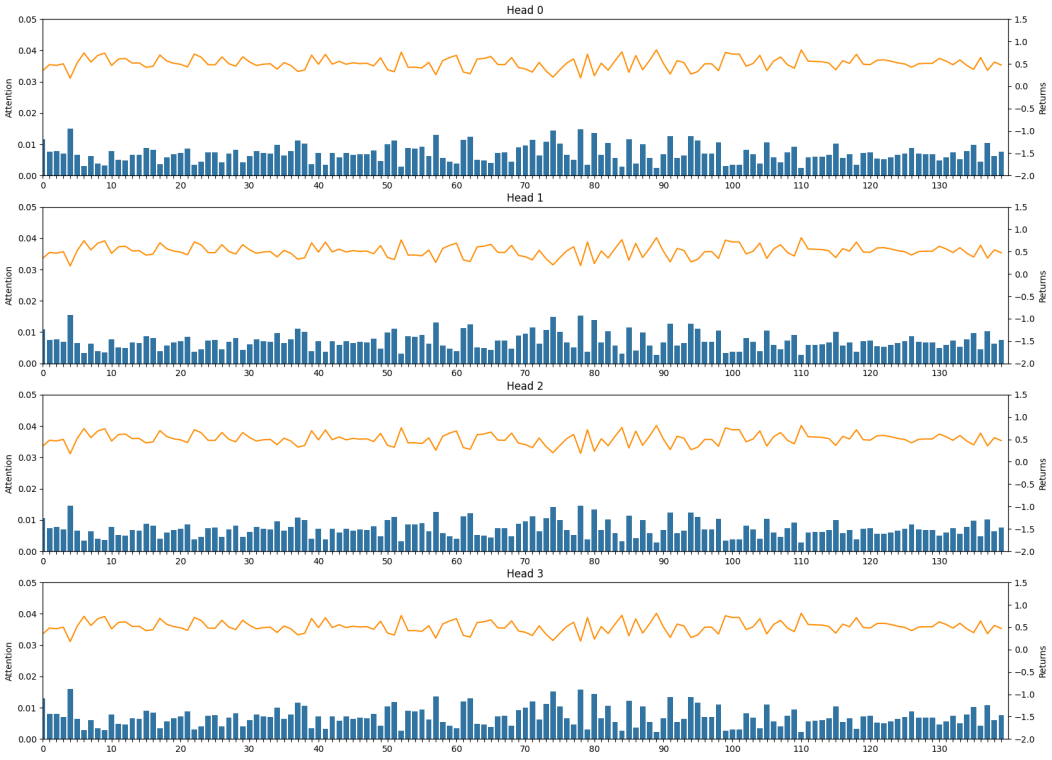
To investigate the effects of the attention mechanism, we analyze the attention weights of high and low-performing models over different regimes. We find that the best-performing models typically have very uniform activations over tranquil periods and high attention to peaks and valleys over volatile periods. While the poorly performing models exhibit the same tendencies, the activation patterns are much more exaggerated and the activations are less uniform during tranquil periods. Examples of typical attention activations over different market regimes and models are available in Figures 6 and 7.

(a) Attention activations of a poorly performing model.

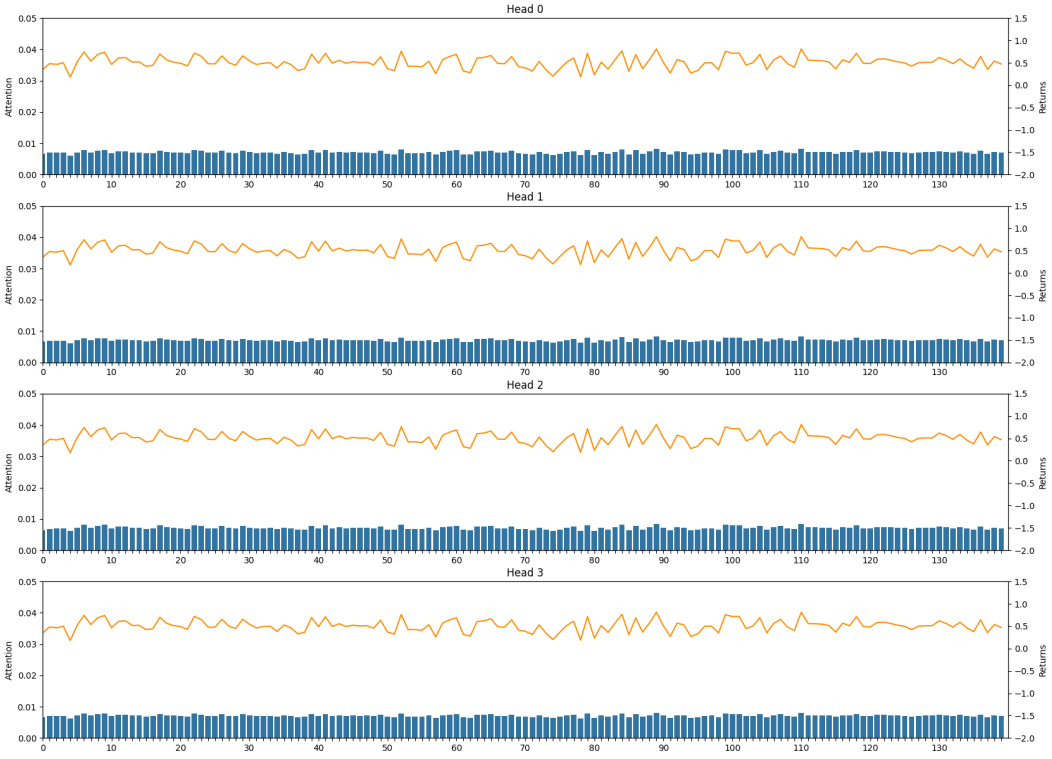


(b) Attention activations of a well-performing model.

Figure 6. A typical example of the attention activations of different TFT models during a market shock. The attention activation at each timestep is represented by the blue bars with the S&P 500 returns for the period represented by the orange lineplot.



(a) Attention activations of a poorly performing model.



(b) Attention activations of a well-performing model.

Figure 7. A typical example of the attention activations of different TFT models during normal market conditions. The attention activation at each timestep is represented by the blue bars with the S&P 500 returns for the period represented by the orange lineplot.

The uniformity of attention activation for well-performing models indicates that the introduction of attention is of little impact. In fact, the presence of notable attention activations seems to negatively impact model performance under normal market conditions. However, the model trained on data ranging from March 2013 to March 2016, a well-performing model never exposed to a market crash, showed increased activations when exposed to data from the market crash in 2020. This might explain why the TFT achieved the best performance of any model when tested on this period out-of-sample if we assume that attention is favorable during regime changes. It is hard to assert the validity of this assumption based on the limited data, but the attention mechanism clearly captures the fact that something unusual is happening and the TFT achieves the lowest RMSE and MAE over the period.

4.5. Static variable importance

Most modern deep learning models act as black boxes and provide little insight into their inner workings. To better understand how these models infer their predictions, there has been a growing interest in interpretability analysis. A common goal when doing an interpretability analysis is to quantify the contribution of each input variable to the inferred output. Popular methods of doing this are accumulated local effects (ALE) and shapley additive explanations (SHAP). These methods examine the contribution of input parameters both globally (across multiple sets of input variables) and locally (for a single distinct set of inputs). The TFT instead seeks to tackle interpretability directly through its variable selection networks. The variable selection network allows for quick and computationally inexpensive analysis of static variable importance with a high degree of accuracy, providing insights comparable to ALE or SHAP.

To evaluate relative variable importance in the different trained models, we randomly select 1,000,000 samples from the dataset, run them through each model, and average their selection weights. The resulting averages are reported in Table 11.

Table 11. The average static variable importance over 1,000,000 randomly selected samples for the best and worst performing models before and after the COVID-19-induced market crash. The most important variable is highlighted in bold for each model and the second most important variable is marked with a star. S denotes the latest S&P 500 price, K denotes the strike price, T denotes the time to maturity, r denotes the risk-free rate, and RV denotes the 30-day realized volatility.

		S	K	T	r	RV
	Performance					
Pre-Covid	Worst	0.0723	**0.7626**	0.0625	0.0168	*0.0859
	Best	0.1218	**0.4484**	*0.3287	0.0323	0.0689
Post-Covid	Worst	*0.2672	0.0413	**0.4706**	0.0231	0.1977
	Best	0.0395	**0.5455**	0.0618	*0.2278	0.1253

As expected, the strike price is a determining factor. Pre-Covid, we find that TTM and the underlying price are important factors. This is in line with [10]. However, post-Covid, we find that the risk-free rate and RV play a more prominent role. This is most likely driven by rising interest rates

toward the end of our sample, following a period of expansive U.S. monetary policy.[§]

5. Conclusions

The LSTM-MLP achieves better performance aggregated over all windows' test set predictions and over most of the windows individually, compared to the TFT. Notably, the TFT seems to handle the market crash in 2020 better than the other models. Further, while the LSTM-MLP achieves lower magnitude in its overall error metrics, the TFT achieves the lowest ME. In other words, the LSTM-MLP is more accurate in its predictions, but the TFT is less biased.

Buyers and sellers of SPX options, along with financial intermediaries such as market markers or clearing members, will benefit from the findings in this paper. The capabilities of the proposed model for non-parametric hedging are of particular interest. Notably, the strong performance of the TFT during the highly volatile COVID-19 period makes it particularly relevant under challenging market conditions.

Within the domain of option pricing, convolutional downsampling in combination with attention is an exciting direction for researchers looking to explore transformers' capabilities within the field. Although attention did not prove useful in this paper, that is not to say that any form of attention is unfavorable. Additionally, interesting novel approaches to sequential processing are present in the transformer variations discussed in this paper, besides variations on the attention mechanism. For instance, including series decomposition blocks [18] in an LSTM-based model could be an interesting avenue for future researchers who aim to design improved option pricing models.

Finally, an important direction for future research concerns hedging performance. Although parametric models may be misspecified for price prediction, they provide explicit functional forms for the greeks (such as delta, gamma, and vega), which enable stable, interpretable, and readily implementable hedging strategies. By contrast, while machine-learning approaches can often achieve lower pricing errors, they tend to produce greeks that are noisier, less smooth, and more difficult to extrapolate. Combined with the limited interpretability inherent in many black-box models, this poses significant challenges for practical hedging applications. In this context, probabilistic artificial intelligence models, as outlined in [19], are particularly promising, since they can quantify both epistemic (model) and aleatoric (market) uncertainty, thereby offering a more transparent framework for risk management and hedging.

[§]Note that our sample ends in September 2023, whereas [10] used data up until December 2022.

Appendix A

A. TFT details

A.1. Modifications

First, we are not concerned with the prediction of values beyond the present-day option value, as such there are no future-value predictions and we may therefore remove the decoder part entirely. Second, there is only one temporal input variable (underlying returns), which makes the variable selection network equivalent to using a standard GRN with no context vector. This would, however, mean that we would lose the static enrichment in this part of the processing. Therefore, we opt for GRNs with a context vector input in place of the temporal variable selection networks. Lastly, we remove the quantile forecasts as the P50 loss would be the only one of interest either way. Thus, the final piece of processing is a simple linear layer and a ReLU activation. The ReLU activation is omitted during training, but present when running validation or testing as it does not make sense to predict a negative value for an option. Note that ReLU activation is omitted during training as the gradient of any negative predictions would be 0 for the entire network in case of a negative prediction. As these modifications only aim to make the TFT fit the problem description, they are not expected to be of consequence to model performance.

A.2. Gated linear unit

[17] proposes a gating mechanism that reduces the vanishing gradient problem in deep networks by maintaining a linear path through the gradients while still maintaining non-linear capabilities. They achieve this by running the input through two separate linear layers, applying a sigmoid activation function to one of the output vectors, and multiplying the resulting outputs. This is only of importance as it is used in conjunction with any layer normalization [15] in the TFT and in the GRN. Mathematically, the GLU performs the following operations:

$$\mathrm{GLU}(x) = \sigma(W_1 x + b_1) \cdot (W_2 x + b_2) \tag{A.1}$$

Where,

σ : the sigmoid activation function

x : the input vector

W_i, b_i : learnable parameters

A.3. Gated residual network

The GRN consists of two linear layers with an exponenial linear unit activation, a GLU, layer normalization [15], and a residual connection. It may optionally take an external context vector that is linearly transformed and added to the input vector.

The order of operations in a GLU are as follows: the input vector is first processed by a linear layer and added to a linearly transformed context vector if present. Then an ELU activation function is

applied before a final linear layer. Finally, the produced output is run through a GLU before the residual is added in and the resulting vector is normalized with layer normalization [15]. Mathematically, the GRN performs the following operation:

$$\text{GRN}(a, c) = \text{LayerNorm}(a + \text{GLU}(\eta_1)) \tag{A.2}$$

Where,

$$\eta_1 = W_1\eta_2 + b_1$$
$$\eta_2 = \text{ELU}(W_2a + b_2 + W_3c)$$
$$a: \text{ the input vector}$$
$$c: \text{ the optional context vector}$$
$$W_i, b_i: \text{ learnable parameters}$$

Note that during training, dropout is applied to η_1 before the GLU is applied.

A.4. Variable selection network

The variable selection network [14] first processes each input variable independently using GRNs. The resulting outputs are then multiplied by a vector consisting of variable selection weights. The variable selection weights are a learned weighing of variable importance obtained by concatenating and flatting all input vectors, then running them through a GRN that has an output dimension corresponding to the number of input variables. Before the matrix multiplication, softmax is applied to the selection weights. This forces the model to prioritize input variables, possibly reducing the interference of irrelevant input variables during inference. Additionally, it grants the ability to inspect the produced variable selection weights over a number of samples to determine the relative importance of each input variable.

A.5. Interpretable multi-head attention

Interpretable multi-head attention [14] is a minor modification of multi-head attention described in [12]. The difference is that each head shares the same value vector. Using the same value vector across all heads safeguards against the possibility of interpreting different activation patterns in each head incorrectly as different patterns learned as the shared value vector guarantees such an interpretation to be correct. In other words, when the value vector differs, differing activations may actually model the same learned relationships. This also means that we can simplify the multi-head attention equation proposed by [12] to be $\overline{A(Q, K)}VW^O$ where $\overline{A(Q, K)}$ is the average scaled dot-product attention over all heads, V is the transformed value vector, and W^O are learnable parameters.

A.6. Static covariate encoder

The Static Covariate Encoder employs four unique GRNs to produce four learned encodings of the static covariates that are used to enrich the context at various places throughout the model; Two of them are used as context vectors in GRNs before and after the LSTM encoder, and two of them are used to initialize the hidden state and the cell state of the LSTM encoder.

Author contributions

BAH: Conceptualization, methodology, investigation, writing – reviewing and editing, supervision; MR: Conceptualization, methodology, investigation, writing – original draft, writing – reviewing and editing, supervision; KAS: Conceptualization, methodology, software, formal analysis, data curation, writing – original draft, writing – reviewing and editing. All authors have read and approved the final version of the manuscript for publication.

Data availability statement

The dataset is publicly available for from optionsDX `https://www.optionsdx.com/product/spx-option-chain/`. All code used in this paper is available at `https://github.com/Lavrans/Thesis/`.

Use of Generative-AI tools declaration

The authors declare they have not used Artificial Intelligence (AI) tools in the creation of this article.

Conflict of interest

The authors report no conflicts of interest.

References

1. F. Black, M. Scholes, The valuation of option contracts and a test of market efficiency, *J. Financ.,* **27** (1972), 399. https://doi.org/10.2307/2978484
2. R. C. Merton, Option pricing when underlying stock returns are discontinuous, *J. Financ. Econ.,* **3** (1976), 125–144. https://doi.org/10.1016/0304-405X(76)90022-2
3. J. Hull, A. White, The pricing of options on assets with stochastic volatilities, *J. Financ.,* **42** (1987), 281–300. https://doi.org/10.1111/j.1540-6261.1987.tb02568.x
4. S. L. Heston, A closed-form solution for options with stochastic volatility with applications to bond and currency options, *Rev. Financ. Stud.,* **6** (1993), 327–343. https://doi.org/10.1093/rfs/6.2.327
5. P. C. Andreou, C. Charalambous, S. H. Martzoukos, Pricing and trading European options by combining artificial neural networks and parametric models with implied parameters, *Eur. J. Oper. Res.,* **185** (2008), 1415–1433. https://doi.org/10.1016/j.ejor.2005.03.081
6. R. Culkin, R. D. Das, Machine learning in finance: The case of deep learning for option pricing, *J. Invest. Manag.,* **15** (2017), 1–9.
7. Y. Cao, X. Liu, J. Zhai, Option valuation under no-arbitrage constraints with neural networks, *Eur. J. Oper. Res.,* **293** (2021), 361–374. https://doi.org/10.1016/j.ejor.2020.12.003
8. L. Liang, X. Cai, Time-sequencing European options and pricing with deep learning—Analyzing based on interpretable ALE method, *Expert Syst. Appl.,* **187** (2022). https://doi.org/10.1016/j.eswa.2021.115951

9. R. Pimentel, M. Risstad, S. Rogde, E. S. Rygg, J. Vinje, S. Westgaard, et al., Option pricing with deep learning: A long short-term memory approach, *Decis. Econ. Financ.,* **1** (2025), 1–32. https://doi.org/10.1007/s10203-025-00518-9
10. J. Vinje, E. S. Rygg, C. Wu, M. Risstad, R. Pimentel, S. Westgaard, et al., Merged LSTM-MLP for option valuation, *Quant. Financ.,* **1** (2025), 1–16. https://doi.org/10.1080/14697688.2025.2493965
11. D. Bahdanau, K. Cho, Y. Bengio, Neural machine translation by jointly learning to align and translate, *arxiv preprint,* 2014.
12. A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, et al., Attention is all you need, *Adv. Neur. Inform. Process. Syst.,* 2017.
13. Q. Wen, T. Zhou, C. Zhang, W. Chen, Z. Ma, J. Yan, et al., Transformers in time series: A survey, *arxiv preprint,* 2023. https://doi.org/10.48550/arXiv.2202.07125
14. B. Lim, S. Ö. Arık, N. Loeff, T. Pfister, Temporal fusion transformers for interpretable multi-horizon time series forecasting, *Int. J. Forecasting,* **37** (2021), 1748–1764. https://doi.org/10.1016/j.ijforecast.2021.03.012
15. J. L. Ba, J. R. Kiros, G. E. Hinton, Layer normalization, *arxiv preprint,* 2016. https://doi.org/10.48550/arXiv.1607.06450
16. J. Bergstra, Y. Bengio, Random search for hyper-parameter optimization, *J. Mach. Learn. Res.,* **13** (2012), 281–305.
17. Y. N. Dauphin, A. Fan, M. Auli, D. Grangier, *Language modeling with gated convolutional networks*, In: Proceedings of the 34 th International Conference on Machine Learning, Sydney, Australia, 2017.
18. H. Wu, J. Xu, J. Wang, M. Long, *Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting*, In: Advances in Neural Information Processing Systems 34 (NeurIPS 2021), 2021.
19. S. Eggen, T. J. Espe, K. Grude, M. Risstad, R. Sandberg, Financial time series uncertainty: A review of probabilistic AI applications, *J. Econ. Surv.,* 2025. https://doi.org/10.1111/joes.70018

AIMS Press