Enhancing the Locality and Breaking the Memory Bottleneck of Transformer on Time Series Forecasting

Ragini Pant[304573]

Department of computer science, Universitätsplatz 1, Universität Hildesheim, 31141 Hildesheim, Germany
pant@uni-hildesheim.de

Abstract. Time Series Forecasting is an approach of using historical data to anticipate future observations. This is an imperative method utilized in various industry sectors that require assistance in forecasting the future. The authors aim to tackle Time Series Forecasting using Transformers. Transformer is stronger than its competitors like RNNs, LSTM and GRU, but it suffers from Memory Bottleneck issues when dealing with large amounts of data. The self-attention mechanism of transformer, uses point-wise linear transformation that diminishes the local context of data. This paper aims to eliminate both of these issues. The authors propose to enhance the locality of the transformers by using causal convolution in query and key transformation. They also propose a novel architecture called *LogSparse* Transformer which reduces memory constraint by choosing to focus on keys at certain indices and still maintaining the information flow. The proposed methods showcase better performance with less memory budget than canonical transformers as well as other baseline models.

1 Introduction

Time Series Forecasting is useful in almost every industrial sector. Each industry has different use-cases, different trends, and the traditional methods of Time Series Forecasting require expert knowledge to manually select trends, seasonality, and other components, thus making Time Series Forecasting very difficult and brittle. Since 2010, deep learning architectures have become a better alternative to solve these problems. Recurrent Neural Networks [**19**], Gated Recurrent Units [**18**], Long Short-Term Memory models [**16**] are some common ways to deal with forecasting problems. RNNs process data sequentially, and retain information through hidden states. They connect previously learned information to the present task. Although, when learning they suffer from vanishing gradient problem when the input is large. Advanced RNN techniques like GRUs [**18**] and LSTMs [**16**] have been introduced but even they suffer from the same issue, and the hidden states focus more on the current state than finding correlation with other hidden states.

The transformer [**1**] is a new deep learning architecture, proposed in 2017 that uses a self-attention mechanism. This mechanism allows Transformers to pay more focus on important information, and it can access any part of the history regardless of distance, making it potentially more suitable for grasping the recurring patterns with long-term dependencies. Transformer does not suffer from vanishing gradient problem, since it has access to all the input tokens at all layers, and also has residual connections, in contrast to RNNs where each token in processed sequentially [**19**].

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

Attention is calculated by computing Query(Q), Key(K), and Value(V) vectors from the input sequence. These vectors are transformed through a pointwise linear transformation. Further, a dot product between query and key is performed, which is scaled down by dividing with the square root of the dimension of keys (d_k). This allows for more stable gradients as multiplying values can have an exploding effect. To get the input probability, softmax is applied. The probability is multiplied with the value vector to get an attention score. The higher the value of the attention score, the more attention, the model will pay to that part of the input.

Apart from long-term dependencies, there are short-term dependencies in the data as well, that get diminished due to the linear transformation of queries and keys. This causes important local context like anomalies or local patterns in the data to be lost. Memory Bottleneck is another key issue in canonical Transformers. Lengthy input causes memory to grow quadratically, making it difficult to model long time-series data with fine granularity. This paper elaborates on the above mentioned issues of the transformer, and the solutions proposed by the authors. The paper was presented in NeurIPS 2019 and has 70 citations.[**15**] Some information about the authors:

1. **Xifeng Yan** is a Professor of Computer Science at the University of California. His research interests include Data Mining, Databases, NLP, Machine Learning and AI. He has received several awards and honors, like Best of ICDM 2019 Selection, ICDE 2013 Best Poster Award, IEEE ICDM 10-year Highest-Impact Paper Award, 2011, and many others. He was also the Program Co-chair at the 2019 SIAM International Conference on Data Mining.[**7**] His works have a total of 21970 citations.[**15**]
2. **Yu-Xiang Wang** is an Assistant Professor of the computer science department at the University of California. Before he worked at Amazon AI as a scientist, and with the Machine Learning Department at Carnegie Mellon University. His research interests include statistical machine learning, e.g. trend filtering, differential privacy, subspace clustering, large-scale learning or optimization, and reinforcement learning.[**8**] His works have a total of 2502 citations.[**15**]
3. **Shiyang Li** is a 3^rd year Ph.D. student at the Department of Computer Science at the University of California. Before that, he obtained his B.Eng.

degree in Computer Science and Technology with honors from Harbin Institute of Technology (HIT), China. His works have a total of 101 citations, in noteworthy conferences like NeurIPS, ICLR, IEEE Conference on Data Mining. [**9**] His works have a total of 118 citations.[**15**]

4. **Wenhu Chen** is a final-year Ph.D. student at the University of California. His research interests include natural language processing, deep learning, knowledge representation, and reasoning. He has also interned in Google Research, Microsoft AI Dynmics365, and Samsung Research America. He has served as Program Committee for ACL, NAACL, EMNLP, ICLR, NeurIPS, and CVPR. He was recognized as the top reviewer in NeurIPS 2019 and has also received the WACV best-student paper honorable mention in 2021.[**10**] His works have a total of 673 citations.[**15**]
5. **Kai-Xuan Yao** is a Graduate student at the University of Chicago, with interests in the field of Experimental Atomic Physics.[**11**] His works have a total of 39 citations.[**15**]
6. **Xiyou Zhou** is a software engineer at OctoML, a machine learning acceleration company in Seattle, US. He did his Bachelors from Fudan University in Computer Science in 2013 and later did his Masters in Computer Science from the University of California in 2017. He has also worked as a research intern at NVIDIA and Google and again as a software engineering intern at Google.[**12**] His works have a total of 121 citations.[**15**]
7. **Xiao-Yong Jin** is an Assistant Computational Scientist at Argonne National Laboratory, United States. He completed his Bachelors and Ph.D. in Physics from Fudan University, China, and Columbia University, US respectively. Later, he joined as a Postdoctoral researcher at RIKEN in 2011, and later at Argonne National Laboratory in 2014. His interests include Accelerator Physics, Applied Mathematics, Data Science, Data Analysis, Deep Learning, High Energy Physics.[**13**] [**14**] His works have a total of 806 citations.[**15**]

2 Related Work

In practical time-series forecasting problems, there are issues like missing data, irregularly sampled data, dealing with channels that are recorded in vastly different frequencies. In the past, some intermittent solutions like ARIMA, Matrix Factorization and Bayesian methods were used to deal with industry problems. ARIMA is an auto-regressive method that employs Box-Jenkins methodology [**2**]. Matrix Factorization approach proposed in [**20**] transforms time series data into a matrix and then use a novel TRMF framework which allows temporal learning using data. [**21**] proposes a hierarchical Bayesian formulation that accounts for explanatory variables and share statistical strength across groups of related time series.

Sharing information across time steps is key for improving the forecasting accuracy. However, this can be difficult to accomplish in practice, due to the often heterogeneous nature of the data. A prominent approach that utilizes sharing

information across time steps are deep neural networks like Recurrent Neural Networks. Some recent work in this field includes [**3**] that propose a novel architecture called DeepAR to produce accurate probabilistic forecasts, by training an auto-regressive RNN model on many related time series. [**4**] is another framework that offers an approach for general probabilistic multi-step time series regression. To incorporate the benefits of traditional models [**5**] uses a hybrid model that combining both traditional and RNN approaches.

In 2017, a new architecture called Transformer was proposed for sequence modeling. It uses a self-attention mechanism that employs correlation between data points [**1**]. However, calculating attention is memory-intensive as it increases with input length. This causes the space complexity of transformers to increase quadratically. This is a serious issue in forecasting time series with fine granularity and learning strong long-term dependencies.

3 Summary

This section summarizes the methodologies explained in the paper, the datasets used and the experiments performed along with their results.

3.1 Methodology

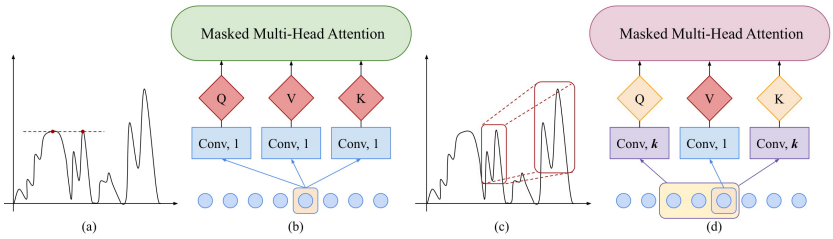

Fig. 1. The comparison between canonical and our convolutional self-attention layers.[**6**]

Enhancing the Locality of the Transformer - In canonical transformers, self-attention is calculated by a linear transformation of queries and keys (point-wise multiplication), as shown in Figure 1(a) and Figure 1(b). This leads to exclusion of the local context around the data points. To mitigate this issue, the authors propose to use causal convolution instead of point-wise convolution. Causal convolutions [**17**] are a type of convolutions that cannot violate the order in which they are modeled, preventing information leaks. The prediction $p(x_{t+1}|x_1....x_t)$ emitted by the model at timestep t cannot depend on any of the future timesteps $x_{t+1}, x_{t+2}, ..., x_T$.

Figure 1(c) and Figure 1(d) demonstrates the author's proposal of causal convolution to calculate self-attention in transformers. Here, the kernel of size k

captures the locality around the data point, while self-attention on the input is calculated.

Breaking the memory bottleneck of Transformer - Self-Attention requires a dot product between query and key vectors. For an input of length N, there are N queries and N keys. This makes the space complexity be $O(N \text{ X } N)$ for just one layer of attention in a transformer.

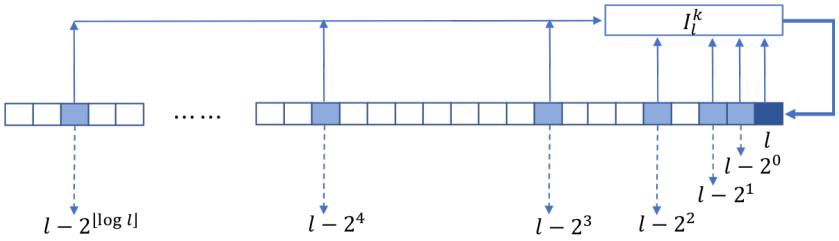

Fig. 2. Method of Indice selection for *LogSparse* Transformer [**6**]

To reduce the memory constraints, authors propose a novel method called *LogSparse* Transformer, where key indices are selected by a geometric sequence of factor 2, that shift by a multiple of 2 with every index. So, $log_2 N$ keys interact with N queries, as illustrated in Figure 2. This reduces the space complexity of the model to $O(N \text{ X } log_2 N)$, as it selects only a subset of the indices, and the memory does not grow quadratically. Selecting only certain indices in each layer will lead to less information retention. To mitigate this, the authors propose to expand the single layer to $log_2 N$ layers and the space complexity reduces to $O((N \text{ X } log_2 N) \text{ X } log_2 N)$ from $O(N \text{ X } N \text{ X } 1)$(1 layer).

Figure 3 illustrates some other proposed variants of the *LogSparse* Transformer:

1. **Local Attention :** To incorporate local information, each key(cell) densely attends to the queries(cell) in its left window of size $O(log_2 N)$.
2. **Restart Attention :** Here, the whole input of length N are divided into sub-sequences with each length set as $N_{sub} \propto N$. In each sub-sequence, *LogSparse* attention strategy is used.

3.2 Dataset

For the experiments, the authors created synthetic dataset consisting of 4 piece-wise sinusoidal signals, where A_1, A_2, A_3 are randomly generated between [0,60] and $A_4 = \max(A_1, A_2)$ and $N_x \sim N(0, 1)$. This dataset depicts various properties like different amplitude and frequency. An example data sequence is illustrated in Fig 4.

Table 1 explains the 7 real-world datasets that were chosen apart from the synthetic dataset. Electricity and Traffic datasets consist of both long-term and

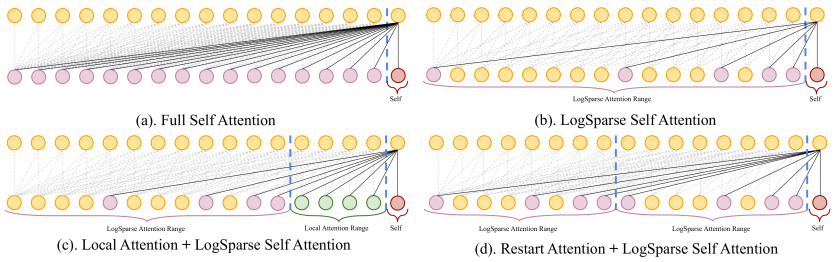

Fig. 3. Different attention mechanism between adjacent layers in Transformer.[**6**]

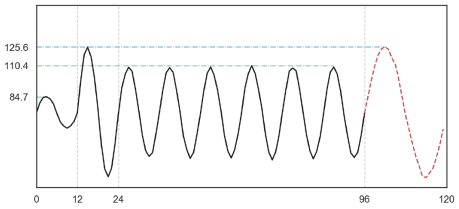

Fig. 4. Example of Synthetic Data Sequence. [**6**]

short-term dependencies in data, whereas solar, wind, and M4-Hourly are good real-world datasets short-term dependency modelling.

Table 1. Real-World datasets Information. [**6**]

	electricity-c	electricity-f	traffic-c	traffic-f	wind	solar	M4-Hourly
T	32304	129120	4049	12435	10957	5832	748/1008
M	370	370	963	963	28	137	414
S	1 hour	15 mins	1 hour	20 mins	1 day	1 hour	1 hour

3.3 Experimental Results

Long-term dependencies - For this experiment, the authors use synthetic dataset to demonstrate that transformers are good at learning long-term dependencies in data. Here, comparison is made between a 3-layer canonical Transformer with standard self-attention and DeepAR [**3**], a 3-layer LSTM with different hidden sizes 20,40,80,140,200 as the baseline.

Figure 5 shows a significant improvement over the state-of-the-art methods such as DeepAR, thus proving the assumption that transformers are good at modeling long-term dependencies. Also LSTM over length 72 cannot perform well, due to not learning long-term dependencies, and also vanishing gradient.
Long-term and short-term forecasting - In this experiment, the authors use electricity-c and traffic-c datasets to test the effectiveness of canonical trans-

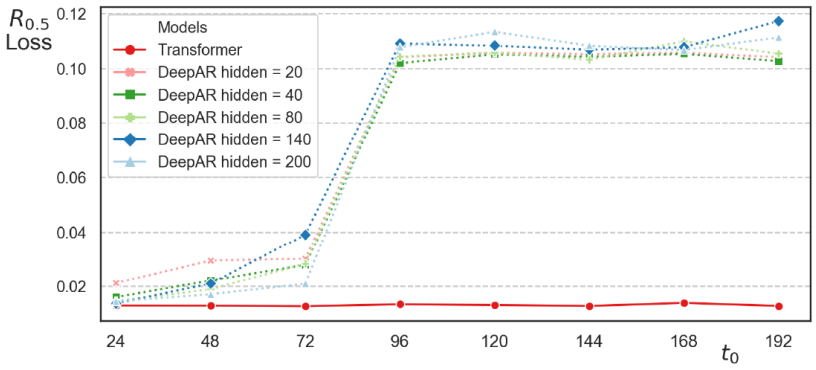

Fig. 5. Performance comparison between DeepAR and canonical Transformer along with the growth of t_0. [**6**]

former equipped with convolutional self-attention in both long-term and short-term forecasting.

Table 2. Results summary ($R_{0.5}/R_{0.9}$ loss) of baseline methods and the authors proposed approach. e-c and t-c represent electricity-c and traffic-c, respectively.[**6**]

	ARIMA	ETS	TRMF	DeepAR	DeepState	Ours
e-c_{1d}	0.154/0.102	0.101/0.077	0.084/-	0.075°/0.040°	0.083°/0.056°	**0.059/0.034**
e-c_{7d}	0.283°/0.109°	0.121°/0.101°	0.087/-	0.082/0.053	0.085°/0.052°	**0.070/0.044**
t-c_{1d}	0.223/0.137	0.236/0.148	0.186/-	0.161°/0.099°	0.167°/0.113°	**0.122/0.081**
t-c_{7d}	0.492°/0.280°	0.509°/0.529°	0.202/-	0.179/0.105	0.168°/0.114°	**0.139/0.094**

Table 2 depicts that models with convolutional self-attention get better results in both long-term and short-term forecasting than other baselines, concluding that adding local context around the data points produces better results. The authors argue that the results for causal convolution in the traffic-c dataset were better compared to the electricity-c dataset due to more long-term dependencies in the traffic-c dataset. Through the results shown in Figure 6, the authors also conclude that adding locality context reduces the training time through the plots of training loss.

Convolutional Self Attention - In this experiment, the authors aim to test the effect of locality, with different kernel sizes in convolutional self-attention on electricity-c and traffic-c dataset over a 1-day time-frame.

Table 3 showcases the consistent gains due to more accurate query-key matching by increasing the kernel size. The traffic-c dataset comprises both long-term and short-term dependencies, and increasing the kernel size led to considerable

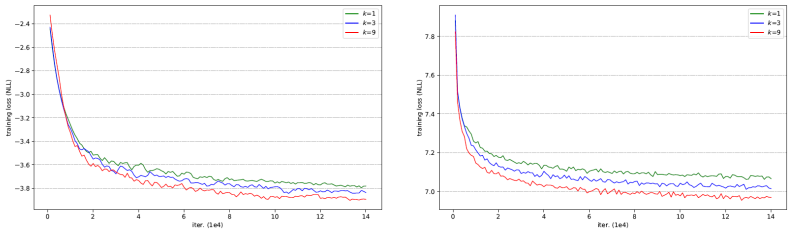

Fig. 6. Training curve comparison (with proper smoothing) among kernel size $k \in$ 1,3,9 in traffic-c (left) and electricity-c (right) dataset.[**6**]

9% improvement. In the electricity-c dataset, the authors argue that the results are less satisfactory as it is a less challenging dataset and the input features have already provided models with rich information for accurate forecasting. Hence, being aware of the larger local context may not help a lot in such cases.

Table 3. Average $R_{0.5}/R_{0.9}$ loss of different kernel sizes for rolling-day prediction of 7 days.[**6**]

	$k = 1$	$k = 2$	$k = 3$	$k = 6$	$k = 9$
`electricity-c`$_{1d}$	0.060/**0.030**	0.058/**0.030**	**0.057**/0.031	**0.057**/0.031	0.059/0.034
`traffic-c`$_{1d}$	0.134/0.089	0.124/0.085	0.123/0.083	0.123/0.083	**0.122/0.081**

Sparse Attention - Here, the authors aim to test the model with memory and length constraints. They create a memory constraint by using a fixed block size memory. This allows *LogSparse* Transformer model to use longer sequences compared to the full transformer. For length constraint, both canonical and *LogSparse* use a fixed-length input for prediction.

Table 4. Average $R_{0.5}/R_{0.9}$ loss comparisons between sparse attention and full attention models with/without convolutional self-attention by rolling-day prediction of 7 days.[**6**]

Constraint	Dataset	Full	Sparse	Full + Conv	Sparse + Conv
Memory	`electricity-f`$_{1d}$	0.083/0.051	0.084/**0.047**	**0.078**/0.048	0.079/0.049
	`traffic-f`$_{1d}$	0.161/0.109	0.150/0.098	0.149/0.102	**0.138/0.092**
Length	`electricity-f`$_{1d}$	0.082/0.047	0.084/0.047	**0.074/0.042**	0.079/0.049
	`traffic-f`$_{1d}$	0.147/0.096	0.150/0.098	0.139/**0.090**	**0.138**/0.092

From Table 4 we can see that under memory constraint sparse attention allows having a longer sequence, and hence leads to a better result. But under

fixed-length constraint, sparse attention performs badly. This is obvious since it uses lesser connections than a canonical transformer. The authors allude that convolution improves results, regardless of sparse or full attention transformer.

Further Exploration In a final experiment, the authors do a performance comparison between the proposed methods and other baselines on all the datasets. The results are shown in Table 5 prove that the authors achieved the best result with a considerable margin.

Table 5. $R_{0.5}/R_{0.9}$ loss of datasets with various granularities.[**6**]

	electricity-f$_{1d}$	traffic-f$_{1d}$	solar$_{1d}$	M4-Hourly$_{2d}$	wind$_{30d}$
TRMF	0.094/-	0.213/-	0.241/-	-/-	0.311/-
DeepAR	0.082/0.063	0.230/0.150	0.222/0.093	0.090°/0.030°	0.286/0.116
Ours	**0.074/0.042**	**0.139/0.090**	**0.210 /0.082**	**0.067 /0.025**	**0.284/0.108**

4 Discussion

This paper is a straightforward extension of the well-known transformer network for time series forecasting. However, it precisely targets two major limitations of the original algorithm [**1**] and proposes improvements to handle them effectively. There is a clear description of the research questions in the introduction of the paper.

This paper targets to the limitations of other RNN models in time-series forecasting, and proves that transformer is a better model architecture. The authors try to focus towards the limitations of transformers, but do not target on other issues of time-series forecasting like missing data, irregularly sampled data points, or forecasting under arbitrary time points.

In the Related Work section as well, the authors explain the different model architectures that can be used for time series forecasting, and which can have better performance, but they do not provide a description to the issues of time-series forecasting that other related works focus on. Different approaches to deal with time-series forecasting were discussed like ARIMA [**20**], DEEPAR [**3**] and others, although their limitations were not included. In the end, the authors also mention about [**1**], upon which the current work is extended.

There is clear information on how the methodologies work for both research questions, with proper diagrams, proofs, and ablation studies. The proposed approach have several advantages, however it misses out on certain details such as -

1. Adding a local context to the data point increases the model performance. Important information like anomalies, local patterns of the data can be learned by the transformer through causal convolutions. However, they failed

to mention the type of anomalies that can be learnt using this architectures, for instance point, contextual or collective anomaly.
2. Using causal convolution prevents any information leak, unlike a normal convolution. Although, the authors do not mention about how the parameters for these convolution kernels have been initialized.
3. Causal convolution reduces the training time and training loss, since it won't have to make extra efforts to remember local information around the data point. Although the authors have not provided any explicit reasoning behind such claims, and therefore, it remains upon the reader to hypothesize such conclusions.
4. The idea behind *LogSparse* is very intuitive. Figure. 3 illustrates how it utilizes the recent cells in the sequence more than the distant cells, thus becoming sparser as the distance from the current key to other unit grows.
5. In transformer architecture, concept of positional embedding was introduced to create transformation of the input that embeds the certain position aspects of the input sequence. In the current works, authors have completely ignored that aspect of transformer, and have promoted causal convolution to solve the embedding of local contextual information.

The authors use 7 different real-world datasets and a synthetic dataset. Although, in my opinion the synthetic dataset was created to yield an impression that transformer performs better when there are long-term dependencies in the data. Instead of synthetic data, the authors could have performed the experiment on a dataset that is more closely aligned to real world dataset and showcase those results.

For instance, it is a widely known fact that real world time series dataset contains various types of noises and anomalies arising from numerous factors such as measurement errors, inconsistent sampling intervals, and missing values. Therefore, in order to make a more affirmative claim on the efficiency of the proposed models the authors should have included such factors in their synthetic dataset.

The authors were able to explain the benefit of each proposed component through carefully designed experiments neatly presented with tables and graphs. Some things that the authors should have considered while performing the experiments, and presenting the results are :

1. The 4-piece sinusoidal waves were constructed with different frequencies and amplitude, the phase value was kept constant. Experiments with different phase values could have been conducted as well, since they are a representation of contextual anomalies that are prominent in time series signals.
2. The authors have not provided any explanation about the metric, or any reasoning about why it has been chosen. Understanding the constraint of the limited publishing space offered for the publication of result, I believe, authors should atleast present a reference to a source or include in appendix, the effectiveness and rational behind the choice of a non common metric for evaluations.

3. There are scant or no details given on how alternative methods (Arima [**20**], TRMF [**21**], DeepAR [**3**]) have been set up. Added to that, there are no official codebase available from the authors, such that the benchmarked model's performance can be recreated. This creates a serious gap in the authenticity of the improvements in the results.
4. In the Long-term and short-term forecasting and Convolutional Self Attention experiments, the authors argue that the performance in the traffic-c dataset is better than electricity-c data due to long-term dependencies in the data. Here, some plots on the real data sequences could have been provided for a more clear understanding. Such plots can clearly validate the author's argument over the increased capability of transformers.
5. The advantages and disadvantages of other variations of *LogSparse* Transformer i.e, local and restart attention could have been explained, and experiments to showcase which performs better should be mentioned. An ablation study of such nature could have created value for the proposed variations.
6. The documented improvement in the performance of both dataset in Table 5 showcases a gain of in the range of 0.1 to 0.002. In such cases, the improvement can have significant dependency over the parameter initialisation. Considering the number of parameter required in transformer, this can become a critical issue while replicating the results mentioned in the paper. To overcome this, author's should have ran multiple runs of the same setup and presented results in the form of mean $\pm$ standard deviations. Presented in this format, it allows the author to estimate the range of performance variance that can be expected during the replication of the proposed algorithm.
7. In the experimentation part for causal convolution, author's have presented training time results and one has to infer seeing the training loss, that the training time might be reduced due to early convergence. Therefore, there are not enough conclusive evidence pointing towards the claim of reduced training time through the use of causal convolution in the current literature.
8. An experiment to compare memory and training time could have been included.
9. In the Table 5, the authors do a comparison with other baselines in all the datasets, but they provide no information on which proposed author's approach; *LogSparse*, or Sparse+Conv is used.
10. Though in Table 5 authors have presented their overall best results across all the datasets, they failed to mention the variant of the proposed approach that yielded such results for solar, winds and M-4 datasets.

In the Conclusion section of the paper, the authors indicate that the proposed approach works and improves the performance of a simple canonical transformer. They establish that the proposed model alleviates all the disadvantages incurred by the original Transformer model architecture [**1**]. The authors were able to prove the proposed methodology through sufficient experiments. With the results obtained, it can be concluded that both causal convolution and novel *LogSparse* Transformer can be a good choice towards improving the performance of the time series prediction.

The paper is presented in a clear format, with necessary information on problem formulation and the background of self-attention. It is formally correct, with correct spelling and no grammatical error. The graphs, figures are well-illustrated. The paper has been cited 68 times since 2019 and has not been revised since.

5 Conclusion

The paper proposed to handle the limitations of the original transformer, through the Causal Convolution approach of adding locality around the data points, and novel *LogSparse* architecture to remove the memory bottleneck. The authors performed various experiments to demonstrate the effectiveness of their approach, and positive results were demonstrated compared to baselines and the original transformer model[**1**] on a time-series dataset.

References

1. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L. and Polosukhin, I., 2017. Attention is all you need. arXiv preprint arXiv:1706.03762.
2. Box, G.E., Jenkins, G.M. and MacGregor, J.F., 1974. Some recent advances in forecasting and control. Journal of the Royal Statistical Society: Series C (Applied Statistics), 23(2), pp.158-179.
3. Salinas, D., Flunkert, V., Gasthaus, J. and Januschowski, T., 2020. DeepAR: Probabilistic forecasting with autoregressive recurrent networks. International Journal of Forecasting, 36(3), pp.1181-1191.
4. Wen, R., Torkkola, K., Narayanaswamy, B. and Madeka, D., 2017. A multi-horizon quantile recurrent forecaster. arXiv preprint arXiv:1711.11053.
5. Maddix, D.C., Wang, Y. and Smola, A., 2018. Deep factors with gaussian processes for forecasting. arXiv preprint arXiv:1812.00098.
6. Li, S., Jin, X., Xuan, Y., Zhou, X., Chen, W., Wang, Y.X. and Yan, X., 2019. Enhancing the locality and breaking the memory bottleneck of transformer on time series forecasting. arXiv preprint arXiv:1907.00235.
7. Information taken from https://sites.cs.ucsb.edu/ xyan/. Accessed on 29-March-2021
8. Information taken from https://sites.cs.ucsb.edu/ yuxiangw/. Accessed on 29-March-2021
9. Information taken from http://shiyangli.me/. Accessed on 29-March-2021
10. Information taken from https://wenhuchen.github.io. Accessed on 29-March-2021
11. Information taken from https://physics.uchicago.edu/people/profile/kaixuan-yao/. Accessed on 29-March-2021
12. Information taken from https://www.linkedin.com/in/zxybazh/. Accessed on 29-March-2021
13. Information taken from https://www.anl.gov/profile/xiaoyong-jin. Accessed on 29-March-2021
14. Information taken from https://www.linkedin.com/in/xiao-yong-jin-58075815/. Accessed on 29-March-2021

15. Citation counts taken from https://scholar.google.com/. Accessed on 26-March-2021.
16. Hochreiter, S. and Schmidhuber, J., 1997. Long short-term memory. Neural computation, 9(8), pp.1735-1780.
17. Oord, A.V.D., Dieleman, S., Zen, H., Simonyan, K., Vinyals, O., Graves, A., Kalchbrenner, N., Senior, A. and Kavukcuoglu, K., 2016. Wavenet: A generative model for raw audio. arXiv preprint arXiv:1609.03499.
18. Cho, K., Van Merriënboer, B., Bahdanau, D. and Bengio, Y., 2014. On the properties of neural machine translation: Encoder-decoder approaches. arXiv preprint arXiv:1409.1259.
19. Connor, J.T., Martin, R.D. and Atlas, L.E., 1994. Recurrent neural networks and robust time series prediction. IEEE transactions on neural networks, 5(2), pp.240-254.
20. Yu, H.F., Rao, N. and Dhillon, I.S., 2016, December. Temporal Regularized Matrix Factorization for High-dimensional Time Series Prediction. In NIPS (pp. 847-855).