Response Letter

Dear editors:

Thanks for providing us with this great opportunity to submit a revised version of our manuscript (IoT-32607-2023.R1, 'Take an Irregular Route: Enhance the Decoder of Time-Series Forecasting Transformer'). We appreciate the elaborate and constructive comments provided by the reviewers. We have carefully revised the manuscript by incorporating all the suggestions by the review panel. We also make an additional change to our work, which clarifies one statement in the previous manuscript that may lead to the misunderstanding of it. It is shown at the end of this response letter.

We first reply to a shared comment then the individual comments of two reviewers by order. The reviewer comments are summarized below in italicized font and specific concerns have been numbered. Our response is given in normal font and changes/additions to the manuscript are ***highlighted*** with red/blue. In addition, we point out the starting locations of changes/additions in the format of **Page XX, Line XX(Left/Right)/Fig. XX/Table XX/Footnote XX** with hyperlinks highlighted in yellow. We appoint the page containing the topic and the abstract of the revised manuscript as **Page 1**.

We hope this revised manuscript has addressed your concerns, and look forward to hearing from you.

Sincerely,

The Authors

Reply to the shared comment

Problem #1

Quantitative results are needed to verify that the computation and space complexity of FPPformer are literally linear with input sequence length. They are also needed to be compared with those of other baselines.

Response #1

We compare the training time per epoch/the GPU memory computation/the inference time per instance on GPU of FPPformer and Crossformer [1], and the identical four measuring criteria of FPPformer without decoder and PatchTST [2] (Section V-F)[Location1]. The input sequence lengths are chosen within {96, 192, 384, 576} and the prediction sequence length is 96. The other hyper-parameters and settings are identical with those used in the quantitative multivariate results, barring the size of the hidden(embedding) dimension. Notice that the size of the hidden dimension, which is the one that exceedingly affects the model complexity, is identical for all baselines in this experiment so that the model architecture design can determine the model complexity to the utmost extent. The first experiment is intended to illustrate that the full FPPformer owns linear computation and space complexity with input sequence length while the second is supposed to illustrate that the encoder of FPPformer also only owns linear complexity and the additional complexity, which is brought by element-wise attention, is subtle. More concrete analysis can be found in our revised manuscript.

Location #1

Page 9, Fig. 6; Page 10, Line 8-36 (Left)

-----End of Reply to the shared comment------

Reply to Reviewer #1

Dear Reviewer,

Thank you very much for your time involved in reviewing the manuscript, the appreciation on our work and the discussion/suggestions on the potential drawbacks of the first version of manuscript.

Problem #1

It seems that there have already existed several researches with well-designed decoder architectures and the rationality of using 'irregular' in the title of this work is doubtful.

Response #1

From the viewpoint of G. Woo et al [3], ETSformer is a well-designed time-series forecasting Transformer (TSFT) with level, trend and season decomposition and its encoder is supposed to extract the preceding three significant decomposed ingredients of input sequences while its decoder is supposed to utilize these extracted features to separately predict the corresponding ingredients of prediction sequences. However, in effect, *ETSformer does not own a genuine Transformer decoder*. The function of a decoder, generally speaking, is supposed to leverage the features outputted by encoders to attain the desired result. Nevertheless, in the definition of attention decoder or Transformer decoder [4], the (Transformer) decoder shall at least *own attention modules* (self-attention modules plus cross-attention modules, to be precise) and *receive the desired output*, with the initialization of certain numbers since they are unknown, as an additional input. We know that this can be a stereotype, however this is exactly something where some researchers lack sufficient consideration, which is one of the standpoints that we argue for. These researchers may have discovered that the current decoder design is somehow deficient, but they just replace the decoder with linear projections or other network architectures, in lieu of striving for solving this issue, albeit their other contributions cannot be suspected. PatchTST [2] and iTransformer [5] are recent typical examples with a linear projection to replace decoder. Albeit owning the function of 'decoder', the decoder of ETSformer *neither owns attention mechanisms*, *nor receives the inputs of the desired output with the some sort of initialization*. Its decoder is indeed composed of *several linear projections with different usages (Level and trend) and the direct segments of encoder feature maps (Season)*. ***It owns the function as a 'decoder', but not as a 'Transformer decoder'***. Another defective usage of decoder, i.e., simply mirroring the usage of the attention mechanisms in encoder, has already been elaborated and analyzed in the previous manuscript so that we keep the corresponding parts unchanged.

We prudently speculate that it is the fact that we did not clarify the decoder definition in Transformer that leads to your doubt. We remedy the definition of Transformer decoder in Section II-a[Location1] in our revised manuscript.

Location #1

Page 2, Line 42-44 (Left)

Problem #2

How do the authors use the other baselines for experiments? Why do the presented results quite different from the original results or the other results provided by other works?

Response #2

1. As claimed in Section V-B[Location1], we pursue the persuasive and fair comparison, rather than the deliberate shiny results in this work. Therefore, it can be observed that in our provided implementation details or GitHub repository (https://github.com/OrigamiSL/FPPformer) that we ***fix the hyper-parameters of FPPformer***, barring in the parameter sensitivity experiment, and so do the other baselines used for comparison. Moreover, these fixed hyper-parameters ***are very commonly seen in other works***, which indicates that we did not deliberately tune the hyper-parameters or settings. Many of these baselines own various settings to present seemingly outstanding performances in all chosen datasets. For instance, PatchTST [2] fails to achieve comparative performances

on ETT datasets if using its default setting, which is admitted by its authors in the Appendix A.1.4 (Page 14) of the original paper. Using a different smaller (1/8) latent space dimension may render PatchTST achieving better results in its main result table, but its capability of handling over-fitting is questioned. Their enthusiasms for challenging the limits of deep time-series forecasting are admired but this is not the reason to obscure their models' substantive performances. Nearly all employed baselines have such problems in their original papers, including your mentioned iTransformer [5]. Thereby, as for other baselines, we employ the default model hyper-parameters and model architecture for them. Then the answer to the question 'How do we use Scaleformer?' is immediately answered: We use its default version, i.e., the Scaleformer [6] combined with the Fourier version of FEDformer [7] (FEDformer-f). Indeed, we have elaborated the hyper-parameters of all other baselines, which are employed in this work, in the README.md of the provided GitHub repository so that anyone is able to check the fidelity of any experiment result in this work.

2. As mentioned in the foregoing point, we fix the settings of all other employed baselines and compare them respectively with the identical input sequence lengths ({96, 192, 384, 576}), so that our presented experiments results are different from the original results (Section V-C[Location2]). As for the results in iTransformer, we do not find any clue about how its authors chose the hyper-parameters of other baselines. We believe that our experiment results are more conceivable from the perspective of the way to choose the hyper-parameters and settings of other baselines.

3. To convince the future readers that our experiment results are persuasive, we additionally claim that the hyper-parameters of other baselines are also fixed and can be found in our provided GitHub repository, in the revised manuscript (Section V-B)[Location3].

Location #2

1. Page 7, Line 22, 23 (Left), 1-6 (Right)

2. Page 8, Table V

3. Page 7, Line 2-6 (Right)

Problem #3

Quantitative results are needed to verify that the computation and space complexity of FPPformer are literally linear with input sequence length. They are also needed to be compared with those of other baselines.

Response #3

It has already been answered in the above '**Reply to the shared comment**' part.

Problem #4

Case study shall encompass the visualization of the model feature extraction capability in the latent space.

Response #4

Following your reasonable request, we present three types of visualizations in the latent space to validate the functions of our proposed mechanisms or architectures in FPPformer, including:

1. The heat maps of the attention score distributions of the first encoder in FPPformer (Section V-G[Location1]). The showcases of FPPformer with our proposed DM patch-wise/element-wise attention, merely with DM patch-wise/element-wise attention in the training phase and with conventional patch-wise/element-wise attention are analyzed to verify that DM attention mechanism can literally assist in tackling the outliers in input sequences and extracting the universal feature maps of input sequences.

2. To show that the additional element-wise attention employed in FPPformer is able to reinforce the inner-patch feature extraction and help the accompanied patch-wise attention better extract the inter-patch relationships, we visualize the feature maps outputted by each patch-wise attention in the encoders of FPPformer, PatchTST and Crossformer by T-SNE [8] (Section V-G[Location2]). To get rid of the influences from other different modules, this experiment is performed as a reconstruction experiment, which reconstructs the input sequences via the ultimate encoder features, and we remove the DM technique in patch-wise attentions of FPPformer but keep the DM element-wise attention. The input sequence length is set as 576 to provide enough data points in the

T-SNE figures.

3. To vividly illustrate that the top-down architecture in decoder literally can render the construction of prediction sequence feature maps more general in the latent space, we visualize the patch-wise cross-attention score distributions of different decoder layers in FPPformer and Crossformer via heat maps (Section V-G[Location3]). It can be observed that the cross-attention score distributions of FPPformer is much more uniform than that of Crossformer, manifesting the better universal prediction sequence construction capability of FPPformer.

Location #4

1. **Page 10, Fig. 8; Page 11, Line 8-25 (Left)**
2. **Page 11, Fig. 9, Line 26-33 (Left), 1-12 (Right)**
3. **Page 11, Fig. 10, 11, Line 13-24 (Right); Page 12, Line 1-16**

Problem #5

Why is FPPformer merely able to surpass PatchTST a little with long input sequence length?

Response #5

We believe that we have mediately answered this question in the very first of Section V-C[Location1]. Evidently, we present more detailed experiment results of short input sequence length than that of long input sequence length. The reason is simple: Providing a model with a longer input sequence length implies that the model can leverage more information so that the performance deviations of different models are curtailed. Recall that the purposes of our renovated patch attention and decoder architecture are to rationally extract input sequence features and hierarchically construct the prediction sequence. When the input sequence length is short, their functions are significant as the patch numbers are relatively small and the available input sequence information is limited. However, when the input sequence is long, they can still exert their enhancement, whereas they fail to play the dominant role since the patch numbers are large enough to provide sufficient useful information. Though there are some models employed in this work not qualified for tackling very long input sequence, those which are qualified can benefit a lot with longer input sequence and it is arduous for FPPformer to completely overwhelm them. If you have any interest in checking out the full results of all benchmarks, rather than the average ones in the manuscript, in our given GitHub repository, you will find out the performance deviations of FPPformer and other baselines, e.g., PatchTST, become smaller with the growth of input sequence length. Moreover, the results of M4 also prove that FPPformer is more competent to handle the forecasting occasions with limited data (Section V-C[Location2]).

Location #5

1. **Page 7, Table IV; Page 8, Table V**
2. **Page 8, Table VII**

We would like to take this opportunity to thank you for all your time and efforts involved. We hope you will find this revised version satisfactory.

Sincerely,

The Authors

-----End of Reply to Reviewer #1------

Reply to Reviewer #2

Dear Reviewer,

Thank you very much for your time involved in reviewing the manuscript and giving precious advises. We have painstakingly considered your suggestions on the corresponding improper statements and improved them. We also provide additional experiment to answer your questions with respect to the complexity analysis and Solar dataset.

Problem #1

The contribution summary in section Introduction is too broad to manifest the novelty of this work.

Response #1

We show our gratitude for your pertinent suggestion on re-organizing our writings. The contribution summary is now five folds in the revised manuscript, which additionally includes the DM attention mechanism and splits the previous second point into two different points for a more specific exposition. We put them here for a convenient scan and they are identical with the ones in our revised manuscript (Section I[Location1]):

1) We propose a novel time-series forecasting Transformer, i.e., FPPformer, which uncommonly and efficaciously improves the decoder architecture of TSFT to break its fetters and excavate its potential.

2) We renovate the decoder architecture of TSFT and change it into top-down architecture for the sake of rationally constructing the prediction sequence in a hierarchical manner.

3) Motivated by a pioneer anomaly detection method, we propose diagonal-masked self-attention to mitigate the negative impacts of the outliers in input sequences.

4) A new combination of element-wise attention and patch-wise attention is proposed by us to compensate the weakness of conventional patch-attention in extracting the inner-features of each patch, with only additional linear complexity.

5) Extensive experiments under diverse settings validate that FPPformer is capable of reaching state-of-the-art on twelve benchmarks with peerless accuracy and robustness.

Location #1

Page 2, Line 8-27

Problem #2

Why does FPPformer seem to perform not that promising under relatively easy forecasting conditions, such as Solar dataset whose data exhibits apparent periodicity?

Response #2

As you point out, in Table IV[Location1] where the quantitative multivariate forecasting results with short input sequence length (96) locate, Triformer and Crossformer completely outperform FPPformer while Scaleformer (combined with FEDformer-f) can partially outperform FPPformer, under Solar dataset. You may have discovered that these three models are the only ones that *are not channel-independent* among all the baselines. Meanwhile, the Solar dataset appears stable and apparent periodicity, which indicates that this periodicity is shared by most of the variables, so that those models who leverage the correlations of different variables can benefit a lot under Solar dataset. Indeed, that's why we would like to additionally present univariate forecasting results with M4 datasets in the previous manuscript (Section V-C[Location2]) since the multivariate forecasting strategy does not count towards the forecasting performances such conditions. Moreover, other baselines, which are also channel-independent, own even worse performances than FPPformer under Solar dataset. We employ the channel-independent multivariate forecasting strategy merely because this work does not propose any novel ideas on handling the correlations of different variables and this strategy is the relatively simple and prevailing one. Additionally, we present the full results of quantitative multivariate forecasting with long input sequence

lengths in our GitHub repository (https://github.com/OrigamiSL/FPPformer) to show that FPPformer is able to take the leading position in whichever dataset.

However, as our response to *Problem #5* of *Reviewer # 1* shows, long input sequence length may result in the marginal performance deviations of different baselines. Thereby, we ***provide the forecasting results of FPPformer-Cross***, which owns an additional cross-variable[1] attention after two cross-time attention, i.e., the original two attentions in FPPformer, in each stage like Crossformer, under Solar dataset in our revised manuscript (Section V-C$^{\text{Location3}}$) and ***supplement the corresponding implementations in our GitHub repository***. An additional baseline, iTransformer [5] accepted recently by ICLR2024, is also used for comparison in this special experiment as it also utilizes both cross-time and cross-variable information. The results show that FPPformer-Cross can easily outperform the other two baselines now with short input sequence length, which demonstrates that the original architecture of FPPformer can be easily expanded or modified to other forecasting formula to achieve better results in specific forecasting occasions.

Location #2

1. Page 7, Table IV

2. Page 8, Table VII

3. Page 8, Table VI, Line 18-25

Problem #3

Quantitative results are needed to verify that the computation and space complexity of FPPformer are literally linear with input sequence length. They are also needed to be compared with those of other baselines.

Response #3

It has already been answered in the above '**Reply to the shared comment**' part.

Problem #4

The manuscript needs a discussion section to point out the limits of this work and the potential future research directions to get around them.

Response #4

We totally agree with your recommendation on adding a discussion section in the manuscript to clarify the limits of this work from the perspective of the authors and we apologize for the absence of it.

We believe that there are at least two limits of this work. Here are their statements and discussions, where are identical to the ones in the revised manuscript (Section VI$^{\text{Location1}}$):

1. The hierarchy in FPPformer can be more exquisitely devised. The 'merging' operation in the encoder of FPPformer is too simple to well represent the feature map of the bigger patch via the two smaller patch ingredients. So does the 'splitting' operation in the decoder. The cutting-edge methods to handle the combination or the split of patches, e.g., SwinTransformer [9], in CV field, where patch-wise attention is also prevailing, can be learned, imitated and modified in time-series forecasting Transformer.

2. Currently, the outlier is tackled via DM self-attention, which roughly mask the entire diagonal of the self-attention score matrix, in FPPformer. Notice that the outliers shall be fewer than the normal segments of time-series sequences, which implies that the majority of masked patches are indeed normal and the masking behavior can negatively influence the feature extraction of input sequences. We believe that applying a prior anomaly detection method to each input sequence before forecasting and then only masking the detected anomalous patches can be a better format of utilizing the DM self-attention.

Both of the foregoing two limits and potential solutions will be our future research directions.

[1] We use the statement of 'cross-variable', rather than 'cross-dimension' in Crossformer, to maintain the identical description of the variable dimension in our manuscript.

Location #4

Page 12, Line 17-41 (Left)

We would like to take this opportunity to thank you for all your time and efforts involved. We hope you will find this revised version satisfactory.

Sincerely,

The Authors

-----End of Reply to Reviewer #2------

An additional change:

During the process of answering the comments of two reviewers and revising the previous manuscript, we find out that there is one statement in the previous manuscript that may result in the misunderstanding of it. In several places, we stated that *Crossformer merges the patches in both encoder and decoder*. In effect, the decoder of Crossformer [1] even does not own a pyramid architecture. We use this statement since the hierarchical process of constructing the unknown prediction sequence is determined by how the model hierarchically uses the encoder features. As the encoder of Crossformer keeps merging patches and its decoder uses the encoder features from the start to the end, i.e., it employs the bottom-up architecture, it is equivalent to say that its decoder also 'merges' patches as the constructed prediction sequence features are the reflections of the feature maps belonging to larger and larger input sequence patches , i.e., from fine-grained to coarse-grained, if neglecting the self-attention module in decoder, which is temporally not involved in the discussion of the decoder architecture design (Section IV-B, the self-attention design is discussed in the later Section IV-C and IV-D). Then the comparison of our proposed FPPformer and Crossformer in Fig.3. can be more vivid. We add the corresponding explanation in the revised manuscript to refrain from any controversy about it (Section IV-B[Location1]).

Location

Page 4, Footnote 1

-----End of the additional change------

Reference

[1] Y. Zhang and J. Yan, “Crossformer: Transformer Utilizing Cross-Dimension Dependency for Multivariate,” in The Eleventh International Conference on Learning Representations, 2023. [Online]. Available: https://openreview.net/forum?id=vSVLM2j9eie.

[2] Y. Nie, N. H. Nguyen, P. Sinthong, and J. Kalagnanam, “A Time Series is Worth 64 Words: Long-term Forecasting with Transformers,” in The Eleventh International Conference on Learning Representations, 2023. [Online]. Available: https://openreview.net/forum?id=Jbdc0vTOcol.

[3] G. Woo, C. Liu, D. Sahoo, A. Kumar, and S. C. H. Hoi, “ETSformer: Exponential Smoothing Transformers for Time-series,” ArXiv, vol. abs/2202.01381, 2022.

[4] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. u. Kaiser, and I. Polosukhin, “Attention is All You Need,” in Advances in Neural Information Processing Systems, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds., vol. 30. Curran Associates, Inc., 2017. [Online]. Available: https://proceedings.neurips.cc/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf.

[5] Y. Liu, T. Hu, H. Zhang, H. Wu, S. Wang, L. Ma, and M. Long, “iTransformer: Inverted Transformers Are Effective for Time Series,” ArXiv, vol. abs/2310.06625, 2023. [Online]. Available: https://api.semanticscholar.org/CorpusID:263830644.

[6] M. A. Shabani, A. H. Abdi, L. Meng, and T. Sylvain, “Scaleformer: Iterative Multi-scale Refining Transformers for Time Series,” in The Eleventh International Conference on Learning Representations, 2023. [Online]. Available: https://openreview.net/forum?id=sCrnllCtjoE.

[7] T. Zhou, Z. Ma, Q. Wen, X. Wang, L. Sun, and R. Jin, “FEDformer: Frequency Enhanced Decomposed Transformer for Long-term Series,” in Proceedings of the 39th International Conference on Machine Learning, ser. Proceedings of Machine Learning Research, K. Chaudhuri, S. Jegelka, L. Song, C. Szepesvari, G. Niu, and S. Sabato, Eds., vol. 162. PMLR, 17–23 Jul 2022, pp. 27 268–27 286. [Online]. Available: https://proceedings.mlr.press/v162/zhou22g.html.

[8] L. van der Maaten and G. Hinton, “Visualizing Data using t-SNE,” Journal of Machine Learning Research, vol. 9, no. 86, pp. 2579–2605, 2008. [Online]. Available: http://jmlr.org/papers/v9/vandermaaten08a.html.

[9] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, and B. Guo, “Swin Transformer: Hierarchical Vision Transformer using Shifted Windows,” in 2021 IEEE/CVF International Conference on Computer Vision (ICCV), 2021, pp. 9992–10002.