Dear Reviewers:

Thank you for the reviewers’ comments. We have studied comments carefully and have corrected. The responses are as follows:

**Reviewer #1:**

1. This paper need a major revision. The current expression of the novel ideas are not clear**.** Quantization is a commonly used technique for FPGA based implementations and multi-PE is also not totally novel. I believe the systematic design of the paper should be a good one, but it is not convincible enough now.

**Response:**

Thank for your recognition to systematic design of our paper. Our main contributions are: a) we design a novel layer-by-layer pipeline which actually pipelining the layer in CNN model; b) we take hybrid bit-width to quantize our model. We mainly focus the difference between the data before normalization (intermediate data) and the data after normalization (feature data), based on discussion about such difference, we devise corresponding bitwise schemes; c) we implement sufficient performance comparison experiments between our accelerator and other hardware platforms. In Sec. 5.1, we also validate normalization parameter quantization’s influence on model’s prediction accuracy. We add an extra figure to show this experiment result.

All these contents are rearranged in our paper, together with necessary hardware design details. Thank you for your suggestions and we have made revision.

1. The 'level' in the paper is used wrongly, it should be 'layer' according to the architecture of the CNNs. Please double check all the scientific naming of the key words and revise them.

**Response:**

Thank you for reminding us and we are very for such mistakes. We have corrected all these errors.

1. Fig. 1 is not referred and also not helpful to the main context.

**Response:**

We have removed this figure and thanks for your reminding.

1. It is not clear through the entire paper why the CNNs show advantage to the temporal models on the temporal data, which is the speed data. Please further extend is and explain it clearly.

**Response:**

We have removed this figure and thanks for your remainding.

1. The idea that Fig.2 is presenting is too simple, should not use a figure but just explanation is enough to understand.

**Response:**

We have replaced this figure with description in text.

1. Does the 'level-by-level' pipeline actually pipelining the 'layer'? if so, it should be 'layer-by-layer' pipelining. Again, please verify the correctness of all the names of the proper nouns.

**Response:**

Our accelerator does work in “layer-by-layer” mode and we have modified all improper scientific names. Thanks for your suggestions.

1. Fig. 10 is hard to understand, how about using the bitwidth as the x-axis?

**Response:**

We have modified this figure’s x-axis into (IRDB, NRDB) in Fig.8 (corresponding to old version paper’s Fig.10) in our revised version paper. In this figure, the x-axis (IRDB, NRDB) just means the combination of intermediate result decimal bit-width and normalization result decimal bit-wise as you suggest. We think it is a better way to demonstrate our experiment. Thanks for your suggestion!

**Reviewer #2**

1. The two compared baselines are all published in 2015, which is already five years. I encourage the authors to compare with more recent and state-of-the-art FPGA implementations to clear show the advances of this paper.

**Response:**

Thanks for your suggestions! We have added comparison with recent and state-of-art FPGA implements into our paper. In Sec. 5.2 of our revised paper, we compare our design with recent fully-binarized CNN accelerators based on FPGA and sparse RNN’s FPGA accelerators. The detail is shown in Table.5b and Table.5c. As shown that, our accelerator has less accuracy loss and higher computing performance than full-binary CNN architectures. When compared with RNN-based accelerators, ours outperforms in computing latency.

1. Also, the quantization has been studied widely, what are the differences in this paper?

**Response:**

Quantization is widely used and studied, however in this paper, we focus on arranging different quantization schemes to different kinds of data: intermediate data and feature data. As we discussed in Sec 3.2, these two kinds of data need different numerical precision and we use this feature to find best bit-width allocation scheme, where both computation precision and computing efficiency can be ensured. We also explore the relationship between normalization parameter quantization and model’s prediction accuracy. This experiment result is illustrated in a new figure in Sec. 5.1.

1. How does the FPGA implementation compared against other ASIC solutions?

**Response:**

The comparison between ASIC solutions and our FPGA implement is illustrated in Table.5a. We choose serval typical ASIC designs which are similar with us in binary weight and quantization. Our solutions can have better computing performance.

1. The writing could be improved largely, including the figures and tables. For example, The Figure 2 is quite confusing. Where do the extra mantissa come from in the two fixed-pointed data? What does the accuracy mean? In the caption, what does the "bitwise" mean?

**Response:**

We are very sorry for our unmatured academic writing. We have replaced confused Fig.2 with more explicit description in Sec 2. The extra mantissa is given by us to show how mantissa bit-width influence data indication performance. As to the word “accuracy”, we have to admit that in original version of paper, we mix up the term “accuracy” and “precision”. In this revised version, we uniformly use word “accuracy” to reflect CNN model’s prediction accuracy and “precision” means quantization format’s numerical indication precision. The bitwise in caption means bit-width. Again, we are very sorry for our unmatured writing and all these confused descriptions in manuscript.

**Reviewer #3:**

1. The idea of this paper is not that novel although he improvement of performance and energy-efficiency of this accelerator is notable. Quantization has been widely used in conversion from floating point to fix-point number.Are there any new ideas in your quantization method?

**Response:**

Thank for your recognition to performance improvement of our paper. Quantization is widely applied and studied in the territory, however in this paper, we focus on arranging different quantization schemes to different kinds of data: intermediate data and feature data. As we discussed in Sec. 3.2, these two kinds of data need different numerical precision and we use this feature to find best bit-width allocation scheme, where both computation precision and computing efficiency can be ensured. We also explore the relationship between normalization parameter quantization and model’s prediction accuracy. This experiment result is illustrated in a new figure in Sec. 5.1.

1. In Section 4.3, the authors said FC-1 is layer is the bottleneck of this accelerator, is this because of the massive computation on parameters? If so, why does the bottleneck become Conv-2 layer in Section 5.1?

**Response:**

FC-1 layer is the bottleneck of this accelerator mainly for two reasons: massive computation on parameters and FC-1 layer having to wait for convolutional layers. Large size of parameters demands massive memory resource on FPGA. On the other hand, if FC-1 layer waits until Conv-2’s computation is done, it brings unbearable cost, for it demanding lots of on-chip storage resource to keep intermediate result and increase pipeline’s waiting time.

Conv-2 is bottleneck on CPU platforms for it’s huge computation size, however, in our layer-by-layer pipeline architecture, Conv-2’s computation is accelerated by pipeline and this layer is no more the bottleneck on accelerators.

1. The title of Section 5.1 is "Quantized Model's Performance". However, the authors firstly show the accuracy result with various bitwise. Then the result of non-quantization version on CPUs is listed in Table 2. Why is the result for quantization not shown, or what is the relationship of Fig.10 and Table 2?

**Response:**

The original Fig.10 is rearranged as Fig.8 with same content. We do not show the quantization code’s running results for quantization version running slower than non-quantization code on CPU platforms. The quantization functions library on CPU platforms is not the “real” quantization functions, it converts fixed into floating data, operates floating data on CPU and finally turns results into fixed again. We think this complex is responsible for more time-cost of quantization code. Based on this reason, different quantization schemes (showed in Fig.10) do not influence code’s performance on CPU platforms, but in revised paper, we still add the quantization code’s running time in Table. 2b and analyse it in Sec. 5.1.

1. In addition, in Fig. 10, the accuracy is best when the decimal bitwise is 8 and 9 for middle results and normalized results, respectively.The question is why the accuracy with more bits is worse, e.g., the accuracies of the third, fourth and fifth groups in Fig.10.

**Response:**

The original Fig.10 is rearranged as Fig.8 in latest version of our manuscript. It is noticeable that when mantissa for middle results (or intermediate results in revised paper) and normalized results is over (8, 9), model’s final accuracy does not rise with better indication precision. Some works have proven that neural networks have numerical robustness in low-precision data format (Zeng X, et al; Cheng G, et al), this feature is widely used by deep neural networks’ quantization and can keep model’s original prediction performance well. On the other hand, final accuracy can do be improved by better indication precision at initial stage of experiment. We consider that the point (8, 9) in Fig.8 lies in a balance condition where both robustness and numerical precision can function well. When decimal bits are increased and model is over this balancing point, model’s robustness is weakened by better data indication performance and this boost on numerical precision cannot cover the loss of robustness, which we consider is the reason why model’s prediction effect cannot be enhanced in last couples of experiments.

**Reference:**

Zeng X, Zhi T, Zhou X, Du Z, Guo Q, Liu S, et al. Addressing Irregularity in Sparse Neural Networks Through a Cooperative Software/Hardware Approach. IEEE Transactions on Computers[J], 69(7):968–85, 2020.

Cheng G, Yao C Ye L, Tao L, Cong H, et al. Vecq: Minimal loss DNN model compression with vectorized weight quantization. IEEE Transactions on Computers[J], 2020.

1. In Section 5.2, the authors compares the results with previous accelerators based on RNN. Firstly, both of these two RNN works are not the most recent. Secondly, I think it is better to compare the result with CNN based accelerators.

**Response:**

Thanks for your suggestions! We have added comparison with recent and state-of-art FPGA implements into our paper. In Sec. 5.2 of our revised paper, we compare our design with recent fully-binarized CNN accelerators based on FPGA and sparse RNN’s FPGA accelerators. The detail is shown in Table.5b and Table.5c. As shown that, our accelerator has less accuracy loss and higher computing performance than full-binary CNN architectures. When compared with RNN-based accelerators, ours outperforms in computing latency.

1. These is no section about related work.

**Response:**

We have removed this figure and thanks for your remainding.

1. Many figures are not cited in the paper, e.g. Fig.1, Fig.5, Fig. 6, Fig. 8, Fig.9, and there are no specified descriptions about these figures, either. Some figures are not clear, which needs to be enhanced. There is no Table 3 in this paper.

**Response:**

We are very sorry for our mistakes and unmatured academic writing. We have rearranged figures and tables in our paper. Also, all the figures and tables are cited and illustrated in text.