Answer to Reviewers

We sincerely thank all the reviewers and editors for providing us so many valuable suggestions to improve the quality of this paper. All the review comments are addressed and answered. Please see the answers below and the corresponding contents in the paper.

The major improvements are listed as follows:

- In section 3, we add a table of all the symbols used so that the concepts are more clear. The original description is also modified correspondingly.
- We add section 6 as a brief summary of all the techniques introduced in section 4 and 5 to give a clear overview.

Reviewer 1

(There are no comments.)

Reviewer 2

The paper is much improved. My main concern is that there are still numerous grammatical issues and still some typos in the paper. A significant amount of copyediting is required.

I have a number of small issues remaining that should be addressed:

1. p 11.4: SRAM is not defined

> **Answer:** Thanks for your comment. We have addded the full name of SRAM in the paper in bracket.
>
> Typical FPGA chips consist large on-chip storage units like registers and SRAM(Static Random-Access Memory), ...

2. Figure 2b needs more explanation. Be more explicit on what the x and y axes represent and the dotted lines.

> **Answer:** Thanks for your comment. We use Figure 2b in this paper to show that the size of state-of-the-art NN models usually far exceeds the size of the storage units (register and SRAM) of FPGA chips. The bar chart denotes the available storage resource on FPGA chips. The dotted lines denote the size of NN models. We have added corresponding explantation in the caption of Figure 2b.

3. p 11.5: I don't think batch, batch size, layer pipeline have been defined. Remember that the reader is not necessarily fluent in all the terminology.

Answer: Thanks for your comment. We agree that the readers may not be familiar to the concepts of batch, batch size, and layer pipeline. What we want to claim here is that processing different inputs in parallel can only improve parallelism, not latency. We think that it is not necessary to use these concepts here. So we simplified this paragraph with none of the above concepts. These concepts are introduced in section 5 when corresponding techniques are introduced.

Most of the FPGA based NN accelerators compute different inputs one by one. Some designs process different inputs in parallel. We refer to the number of concurrently processed inputs as concurrency. So the latency of the accelerator is expressed as equation 2.

4. p 11.10: sect 5.1.2 we estimate the theoretical benefit. sect 5.1.3 we estimate the theoretical hardware performance gain as 2x. How are you making these estimations? You have to say enough for the reader to make a judgement as to whether your estimations are valid.

Answer: Thanks for your comment. For the estimations, we have added more explanations. The 4× performance gain is from the analysis of previous work [4, 6]. Larger transformation size may leads to higher performance gain but also needs more resources and reduce the DSP utilization ratio because of the large unrolling parameters. So the theoretical performance gain from fast convolution should be at least 4×. The 2× performance gain estimation of the optimization on DSP working frequency also comes from the result of existing designs [5, 1].

Multiplication with transformation matrices A, B and G induce only a small number of shift and addition because of the special matrix entries. In this case, the number of multiplication is reduced from 36 to 16.The most commonly used Winograd transformation is for 3×3 convolutions in [4, 6].
The theoretical performance gain from fast convolution depends on the convolution size. Limited by the on-chip resource and the consideration of flexibility, current designs are not choosing large convolution sizes. Existing work point out that up to 4× theoretical performance gain can be achieved by fast convolution with FFT [7] or Winograd [4] with reasonable kernel sizes.
...
Xilinx has also proposed the CHaiDNN-v2 [1] with this technique and achieves up to 700MHz DSP working frequency. Compared with existing designs for which the frequency is within 300MHz, this technique brings at least 2× peak performance gain.

5. p. 11.16: Fig. 6 - I suggest putting FP in squares and GPU in triangles to make them stand out even better and make it visually easier to see where the different approaches lie. It is especially important for those that are

color challenged!

> **Answer:** Thanks for your suggestion. We have changed the mark to make GPU results and 32-bit floating-point results more distinct to other results.

6. "Overall, the improvement does not match the estimation ..." What estimation?

> **Answer:** Thanks for your question. Here, we would like to claim that existing work does not reach the theoretical performance gain. The word 'estimation' is not precise and we have changed the description here.
>
> Compare with the theoretical 4× performance gain introduced in section 5.1.2, there is still 1.3 1.5× gap. Not all the layers can use the most optimized fast convolution method because of kernel size limitation.

7. p. 11.17: Fig. 7. You do not comment on the Logic/BRAM chart. If you do not say anything about it in the text, then it should be removed.

> **Answer:** Thanks for your comment. We are trying to get some insight of the resource requirements of current accelerator designs which may help both accelerator designers and FPGA manufacturers. Figure 8 is referred to in the following paragraph:
>
> *We also investigate the resource utilization of the designs in Table 3. Three kinds of resources (DSP, BRAM, and logic) are considered. We plot the designs in Figure 8 using two of the utilization ratio as x and y coordinate. We draw the diagonal line of each figure to show the designs' preference on hardware resource. The BRAM-DSP figure shows an obvious preference on DSP over BRAM. A similar preference appears on DSP over logic. This indicates that current FPGA designs are more likely computation bounded. FPGA manufacturers targeting neural network applications can adjust the resource allocation accordingly.*

Reviewer 3

A survey paper needs to add value to its readers. The readership of a survey paper are researchers who want a snapshot of the state-of-the-art, understand the trends, as well as challenges and opportunities. The original manuscript failed to provide this value. The revised manuscript adds some superficial changes and a nice Section 2.1. But beyond that the authors simply ignored most of my concerns. As such I cannot recommend this paper for publication.

In particular, the survey only focuses on a narrow topic, namely CNN inference with primarily quantization and unrolling as the only optimizations. These optimizations are only at the kernel level; but the survey does not consider higher design architectural design issues such as how these kernels are put together or even CNN consisting of multiple layers and communications among layers are put together. The survey is completely missing the big picture, the system level design issues, memory bandwidth

and storage issues, major architectural trends etc.

Answer: Thanks for your comment. We agree that a survey paper should give insights of the whole area from a higher level. Following your suggestion, we added section 8 as a summary of all the techniques mentioned in section 4 and 5 and discuss the future of NN accelerator design.
But we also need to point out the contents in section 5.3 aready covers some of the points you mentioned.
The roofline model shows how the bandwidth limits the system design. The techniques in section 5.3.2-5.3.4 all address this problem.
The main concern of connecting multiple layers is the data transfer from one layer to the next. Most of the existing designs process the network layer by layer so the connection is still memory and bandwidth issue. We also introduce the cross layer scheduling strategy in section 5.3.3, which involves a network level concern to the design.
To make it clear, we added sub titles to different parts of section 5.3.

1. As your survey focuses on CNN and inference, please state that directly in the tile and abstract.

Answer: Thanks for your comment. We agree that we are focusing inference but not training. But we also cover accelerator designs for other networks like [3, 2]. Networks like LSTM with only fully connected layers can be treated as a special case for CNN besides the difference in network topology, where there is a loop for RNN. Most of the techniques introduced are suitable to both CNN and RNN or other state-of-the-art networks. We have added the 'inference' attribute to neural network accelerator in both the title and the abstract.

2. Section 3 remains as confusing as before. In Equation (1) what is actual_performance? What's the metric? What is workload? What's the metric again. How is utilization measured? What is the unit of throughput?

Answer: Thanks for your comments. To make it clear, we added the new table 1 to give the definition and unit of all the symbols used in section 3 and rewrote the rest part.

3. As there has been some time between the submission and current time in a fast moving field, the authors need to include newer systems, for example, CHaiDNN etc.

Answer: Thanks to your suggestion. We have included the CHaiDNN you mentioned and reviewed the latest conferences in the past few months including FCCM, DAC, ISLPED, etc.

4. There are numerous typos and grammatical mistakes even in this version.

Answer: Thanks to your comments. We have also gone through this paper again to check spelling and grammatical errors.

References

[1] `https://github.com/Xilinx/chaidnn`. Accessed August 23, 2018.

[2] Yijin Guan, Hao Liang, Ningyi Xu, Wenqiang Wang, Shaoshuai Shi, Xi Chen, Guangyu Sun, Wei Zhang, and Jason Cong. Fp-dnn: An automated framework for mapping deep neural networks onto fpgas with rtl-hls hybrid templates. pages 152–159, 2017.

[3] Song Han, Junlong Kang, Huizi Mao, Yiming Hu, Xin Li, Yubin Li, Dongliang Xie, Hong Luo, Song Yao, Yu Wang, et al. Ese: Efficient speech recognition engine with sparse lstm on fpga. In *FPGA*, pages 75–84, 2017.

[4] Liqiang Lu, Yun Liang, Qingcheng Xiao, and Shengen Yan. Evaluating fast algorithms for convolutional neural networks on fpgas. In *Field-Programmable Custom Computing Machines (FCCM), 2017 IEEE 25th Annual International Symposium on*, pages 101–108. IEEE, 2017.

[5] Ephrem Wu, Xiaoqian Zhang, David Berman, and Inkeun Cho. A high-throughput reconfigurable processing array for neural networks. In *Field Programmable Logic and Applications (FPL), 2017 27th International Conference on*, pages 1–4. IEEE, 2017.

[6] Qingcheng Xiao, Yun Liang, Liqiang Lu, Shengen Yan, and Yu-Wing Tai. Exploring heterogeneous algorithms for accelerating deep convolutional neural networks on fpgas. In *Proceedings of the 54th Annual Design Automation Conference 2017*, page 62. ACM, 2017.

[7] Chi Zhang and Viktor Prasanna. Frequency domain acceleration of convolutional neural networks on cpu-fpga shared memory system. In *Proceedings of the 2017 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays*, pages 35–44. ACM, 2017.