Machine Learning for Pilot Biometrics

DESIGN DOCUMENT

Team Number: sddec20-01
Client: Rockwell Collins
Advisers: Akhilesh Tyagi
Team Members/Roles:
Jianhang Liu -- Data Manipulation SME
Feng Lin -- Hardware SME
Xuewen Jiang -- Camera Interface SME
Xiuyuan Guo -- Algorithm SME
Sicheng Zeng -- Python SME
Junjie Chen --C code SME

Team Email: sddec20-01@iastate.edu
Team Website: https://sddec20-01.sd.ece.iastate.edu

Revised: Date/Version

Executive Summary

Development Standards & Practices Used

List all standard circuit, hardware, software practices used in this project. List all the engineering standards that apply to this project that were considered.

Development Standards:

- ISO/IEC 12207
- TCP/IP
- FTP Circuit Standards
- High and low level schematics
- Parts lists
- simulation results
- OpenCL
- XML
- Linux Standard Base
- Unified Modeling language
- Message Passing Interface
- PCB123

Summary of Requirements

List all requirements as bullet points in brief.

- Use live-feed camera input
- Extract Frame Using Open-CV libraries
- Use Tensorflow & Keras pre-trained model on large dataset
- 6 x faster processing than existing algorithm
- 99% above accuracy on open-closed eye detection
- Reduce the model to a acceptable rate

Applicable Courses from Iowa State University Curriculum

1. CPRE 281
2. CPRE 288
3. CPRE 482x
4. COMS 352
5. COMS 228
6. COMS 327
7. COMS 311
8. EE 224
9. EE 324

New Skills/Knowledge acquired that was not taught in courses

List all new skills/knowledge that your team acquired which was not part of your Iowa State curriculum in order to complete this project.

1. Machine Learning
2. Acceleration on Embedded Systems
3. Camera interface
4. OpenCV library
5. 2-D image processing
6. PCB design on schematic & Layout

Table of Contents

List of figures/tables/symbols/definitions (This should be the similar to the project plan)

Figures:

Tables:

1 Introduction

1.1 Acknowledgement

Special Thanks of gratitude to JR Spidell from Collins Aerospace who gave us the golden opportunity to do this wonderful project.

1.2 Problem and Project Statement

Problem:

Hypoxia, Fatigue, Strain Doubles would cause pilots to crush their airplane, huge losses economically and not to mention they put their life in danger.

Project Statement:

Machine Learning algorithm has been developed to detect the blink rate on pilots, based on research, several blink patterns are related to Hypoxia, Fatigue and strain Doubles. Machine learning is loaded onto a fast processing edge device(FPGA) to detect such fatigue patterns. Our goal is to improve the existing machine learning algorithm, in terms of accuracy and efficiency by accelerating the algorithm from both software and hardware perspective.

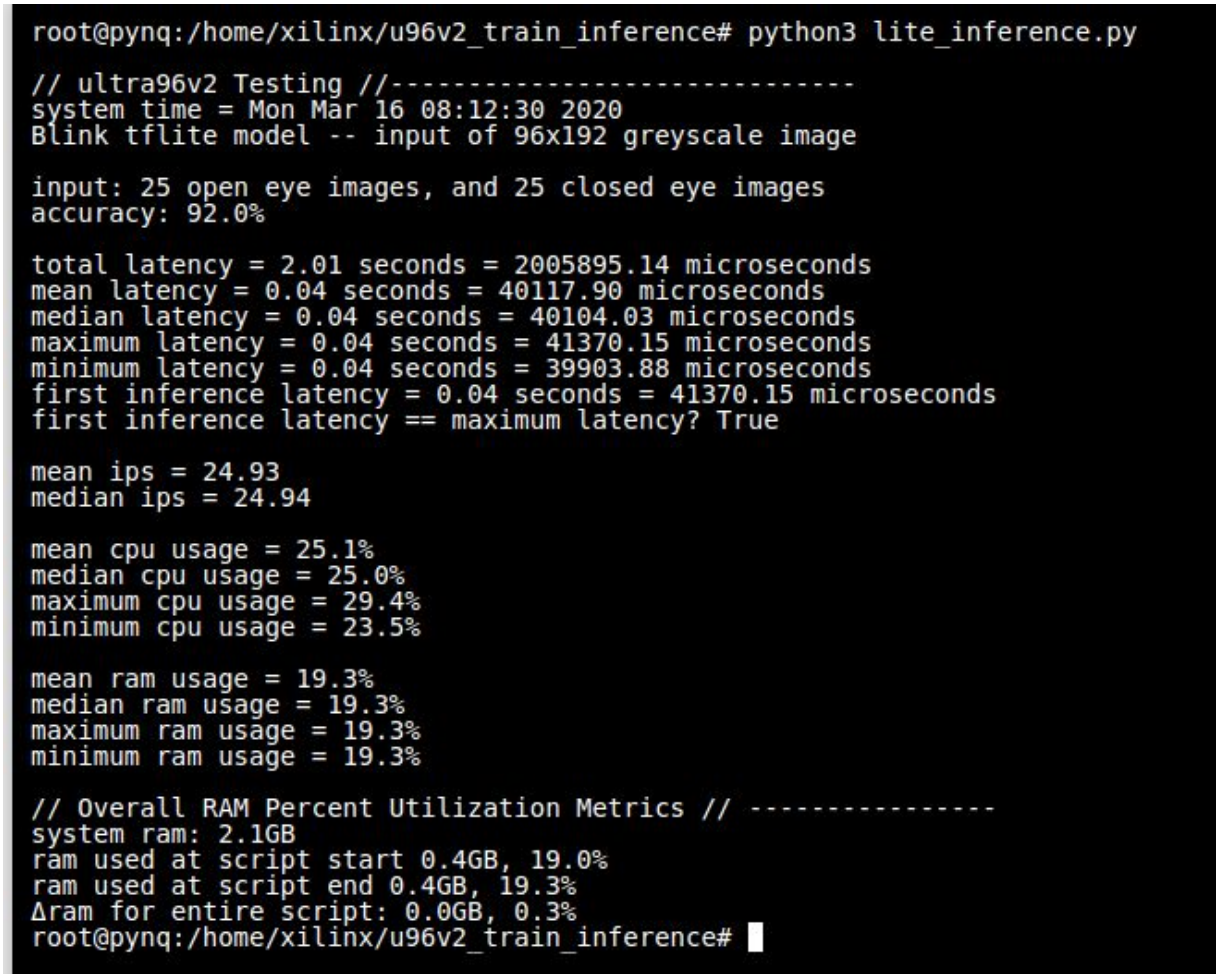


figure 1

[this is the screenshot of the baseline metrics using tensorflow lite, we can observe the memory usage and latency]

1.3 Operational Environment

The end product will go into an aircraft cockpit environment. There are a lot of environmental constraints associated with that. However, for our project, we will not consider an aircraft cockpit environment involved in our design.

However, we are not designing the mechanical or electrical systems here. So this specific project doesn/t have environmental constraints. There are few software environments and software platforms.
1.we are using Anaconda framework and a list of packages need to be installed.
2.the ML inference will run on Ultra-96 board which is a linux os.

Instead of these software environments, the constraints we have to work with are related to latency, accuracy, and memory consumption.

1.4 Requirements

our goal is to improve performance of ML algorithms. However, if we want to reduce latency, we have to do some changes on our original algorithms, like pruning , changing hyper-parameters, data pre-manipulation all these changes will cause a drop of accuracy. so we want to minimize the accuracy drop or keep the accuracy no difference.

Functional Requirements:

Stage 1: Improve accuracy of the algorithm -- not less than 97%

Stage 2: Improve accuracy of the algorithm after pruning -- higher than 84%

Reduce the latency of the algorithm -- reduce latency by 75%

Reduce the memory requirements -- reduce memory requirement by 75%

Reduce the memory requirements -- reduce memory requirement by 80%

Non-Functional Requirements:

Usability: The system shall be used by pilots to achieve quantified objectives with effectiveness, efficiency, and satisfaction in a quantified context of use.

Constraints:

There are multiple architectures that we could choose for the DPU(FPGA part), however the edge board memory limits our choices, if we got a larger memory and frequency development board, we could take more advantages of choosing larger architecture. To solve this problem, we have other approaches such as hyper-parameter tuning, network quantization.
Software environment is also one of the constraints. For example , at the beginning of the project we were try to re-do some existing project in order to get familiar with tools that we might use in our project, however each testing project we were re-doing require different environment software version, so it's hard to decide which version of software and environment that we were using for our project.

Economic/Market requirement:

1) material costs, 2) development costs

Material costs can be 1) development materials 2) the material cost to make the product.
From a development material perspective, we have to live within the constraint of $1K for materials that have already been purchased.
The material cost to make the project is not a constraint for this project, since we are focused on the algorithms.
From a development cost perspective, the primary driver is the cost of people's time. Our budget for this is the allowable time we have between now and your final report.

Environment Requirements:

There are no specific environmental requirements for this project.

Operating environment: The system must work under the Xilinx FPGA board.

1.5 Intended Users and Uses

Our intended users for the end product, the whole large project, will be the military pilot and help them protect their life from unexpected health problems which may make the warcraft out of control.

However, we are the middle part of the whole large project. In our perspective, we are working for our client's request, and the client will use our design to improve the end product. So, we don't need to think about how to serve the military pilot into our project. We also don't need to think about the airspace usage thing like that.

So, the constraints we have to work with are related to latency, accuracy, and memory consumption.

1.6 Assumptions and Limitations

Assumptions	For our previous assumption on the project, the accuracy of the algorithm will not be less than 97%; the latency of the algorithm will reduce latency by 75%; memory requirement for reducing memory by 75%.
Limitations	We don't have actual users for our project at this time. The limitation is the optimization of accuracy, latency and the memory should get the client requirements. Limited numbers of eye datasets. Existing helmet mounted open/closed eye datasets lacking.

table[1]

1.7 Expected End Product and Deliverables

Expected End product and Deliverables:

1. Optimized Machine Learning Algorithm

We will improve the existing machine learning algorithm using xilinx IP (optimization framework), tensor board, camera interface, to find the optimized efficiency while not losing significant accuracy. The technique we use such as finding optimized hyperparameters, pruning, parallel computing in programmable logic, in hope to achieve a 6 times faster detection capability.

2. Use the MIPI camera interface to catch the eye image.

The MIPI camera will improve the latency and memory compared with the USB camera we use at the beginning. The MIPI camera will work well and catch the image we want in the testing.

3. Integrated eye-detection system composed of FPGA, MIPI, and needed software

2. Specifications and Analysis

2.1 Proposed Approach

Functional requirements:

Our large functional requirements are to reduce the Latency, accuracy and memory consumption for the current design.

In order to get our deliverables, we should work on pruning, hyper-parameters, camera, FPGA, data pre-manipulations, and a few ways to improve our ML algorithm, like quantization, tensorflow lite.

hyper-parameters :result we have is that latency with 1598 and the accuracy would be 93% which for both latency improvement with 12.73 percent and the accuracy decreased would be 4.25%(for more information please go page 12 Hyperparameter Tuning)

camera string: We can get the image but we are not satisfied with it, we want to use some camera features to do data pre-manipulations in order to reduce workloads for our main CPU.

pruning: we have tested one pruning way to reduce our ML algorithm layers, and it hurt our accuracy a little, but it reduces almost half memory usage which is a significant improvement.

FPGA: we implement a sample project for learning how to use several tools to design and modify FPGA on ultr96-board, and we will continuously work on our final goal which is using a DPU to do math calculation for our main CPU.(page 12 FPGA)

data pre-manipulations: we have tested multiple manipulated images processed from Matlab to see how they affect our ML algorithm.(page 13 matlab)

Non-functional requirements:

Usability: The system shall be used by pilot to achieve quantified objectives with effectiveness, efficiency, and satisfaction in a quantified context of use

Scalability: The system shall be robust enough to work with different pilot's eyes

Maintainability: The system shall be easily maintainable

For non-functional requirements, we don't have environmental requirements.

Standards:

1. Circuits standards:
    - Schematics
    - Parts lists
    - Simulation results
2. Digital Design standards:
    - High and low level schematics
    - Simulation results
3. Software standards:
    - Source code
    - In-line comments
    - External documentation

2.2 Design Analysis

1. Set up environment:

discuss what we did so far:

V1: We setted up the anaconda environment, Xilinx tool suite, Ultra96-v2 board, experimented with Tensor board, and made our first progress on making RGB color space pictures into grayscale, which gives us significant improvement in time efficiency.

Did it work? Why or why not:

V1: It worked, because for pictures that are 3 color space requires a lot more computation, when we can reduce it to grayscale without losing accuracy, grayscale gives 42% faster performance,

Our observations, thoughts, and ideas to modify or continue:

---Machine learning algorithm acceleration can be done from both software and hardware, the real significant acceleration only comes from writing VHDL and programming at register level, however

software acceleration is just as important. The best acceleration comes from a deep understanding of the algorithm from a software perspective, then designing a custom chip, ASIC for that specific machine learning algorithm. .

Results:

Improved existing machine learning algorithm, by reducing color space from RGB to grayscale, we see an improvement in time efficiency by 42%

Strength:

---Huge improvement on existing machine learning algorithm in terms of speed

---Huge improvement on model size after prune, the model size decrease from 10MB to 1MB

weakness:

A lot more acceleration is left undone, pruning, optimized hyperparameters, hardware acceleration etc.

---Model after pruning data have some lack for sparsity and we need to collect more images to test it.

2. Camera Setting:

discuss what we did so far:

We set up the USB camera on Ultra 96.

Did it work? Why or why not:

It works and we get the image from the Ultra96.

Our observations, thoughts, and ideas to modify or continue:

We can get the image but we are not satisfied with it. Can we modify the camera interface so we can do some analysis or change the pixel, color to serve our algorithm?

Weakness:

It is hard to modify the USB camera we use because it is a finished product, not used for the experiment testing. We need to find another camera interface product that we can modify with it.

3. Matlab

Matlab is a data processing software that can be used to apply filters on input images and modify the image with a variety of methods with the help of built-in functions. In this project, Matlab is used to pre-manipulate the input data so that the machine learning algorithm can achieve higher accuracy and less time cost. One idea is to modify the image so that the algorithm will have less calculation (e.g. noise reduction), another idea is process the image to replace layers in the algorithm, which may have the same effect as the first method (e.g. gabor filter). I have built a variety of image filters and tested them in the algorithm, comparing the results with original image input: Noise reduction, gabor filter, overexposure, resizing, and more.

4. Hyperparameter Tuning

discuss what we did so far:

I am working on the hyperparameter of our training model to find the best hyperparameter to reduce the latency of the model while the accuracy is still in a high standard. So far the hyperparameter I have chosen includes: padding, number of the hidden layer, kernel size and max pooling for both filter and the hidden layer, number of the filter, epoch, batch_size, optimizer, activations function. stride of both max pool and the stride of the pool. Upon that, I have been working with the data premapulating side to reduce the filter layer by applying the existing layer.

Our observations, thoughts, and ideas to modify or continue:

so far the best hyperparameter we have find is to use the number of filter to 16 , number of the filter layer to 1 , number of the hidden layer to 1 , epoch to 25 , batch size to 64, external filter i have used is the over exposed, the optimizer is the nadam, padding would be the same as what we used before, filter size I have pick the 5 for the kernel size, Activation function would relu, the pool size would be 2 and the stride of each pooling would be 2 stride of filter 5, activation function would be relu. the result we have is that latency with 1598 and the accuracy would be 93% which for both latency improvement with 12.73 percent and the accuracy decreased would be 4.25%

weakness:

The weakness of this approach would be it took a lot of time to find the best hyperparameter that could fit into your model. There are so many choices that you can pick from and it could find so many hyperparameters that do not match. Therefore, I think in the future I will find the best hyperparameter by using the auto tuning tool which is called SegaMaker.

5. FPGA

The FPGA in our project is using a DPU(Deep learning process unit) to accelerate algorithms. DPU is a co-processor designed by Xillinx, and it can do complex matrix math for CPU(arm cortex-a53 on ultra-96). To implement FPGA and DPU, we have to learn it step by step, so we did a similar project in order to learn how those tools work and get familiar with FPGA. First we have to build an environment and get two software tools ready. It requires at least 160 GB free space and vitas + vivado. After that we would rebuild PYNQ. The rebuild resource could be found on github which was published by Xillinx. Through this 'study' project, we have an idea about how to modify or upload FPGA design on board. First we have to build or get a .bsp file, which is a FPAG design, then we .xpr file of the FPGA design and load it into Vivado and export hardware, next we open Vitas to create a application project in order to run hardware file which is exported from Vivado, final step is connect Vitas to our ultra-96 board and run the project.

For the future work, we will use the same steps to implement a DPU on ultra-96 board and see how fast will a DPU improve our ML algorithm.

6.Daughter card design

We use PCB123 to build our own daughter card for our MIPI camera and the Ultra 96. The adapter is important, because there is no port for MIPI camera directly on Ultra 96. We find a similar module, AISTARVISION MIPI Adapter as our prototype to design the daughter card.

Weakness: Many parts and functions are not what we want on the prototype of the MIPI adapter. We need to find our requirements, and think of the cost performance.

7. prune

We use the TensorFlow prune tool and ANN visualization to reduce the size and increase accuracy. In the end, we reduce 80% size and make the accuracy in an acceptable range. We can obtain more efficient results for our programs through machine learning. I tried many different prune methods, unit prune, weight prune, fisher prune, dropout technology. Some technologies do not work or fit on our project. We make more tests by changing those main variables to make the result better. That technology goal is to make the model less size but still have the highest accuracy. The weakness is prune may cut some important connections and nodes. But we used lucid and neteron to avoid part of the effect.

2.3 Development Process

Agile is the process we choose to follow, the rationale being agile lets us fast iterate and it is highly adaptable to changes. We have a baseline to test efficiency and accuracy, and we experiment with different techniques, if a technique seems promising, we integrate it.

our project process could divide into several parts:

Camera:

At the beginning, we have to figure out which camera we are going to use, we consider cameras with latency, accuracy, memory usage, cost impact,schedule impact, technical opportunity, technical risk. We did a trade study and grid search on camera and decided to use two types of camera which are raspberry PI(OV 5647) and Logitech(C920s). We have successfully used a logitech camera, and for raspberry PI camera, we need a daughter card to connect the camera with our board, however due to COVID-19 we are not able to get one from industry. So we decide to design our own daughter card.

Hyper-parameter:

Before we change hyper-parameters, we have to learn how to use several tools in order to make the right decisions. At the beginning, we learned how to use tensorflow and how to change hyper-parameters. Then we started a grid search on different configurations for the ML algorithm. For now we found the best hyper-parameter for our algorithm.

Pruning:

We found a pruning way that could reduce almost half the memory usage of our board. However, after pruning the algorithm, we also lose some accuracy. For the pruning part, we made several different programs to optimize our model. Optimize the model by trimming the overall model, adjusting the sparsity, optimizing the model format and level, and reducing the input data size. Our main goal is to do everything possible to improve the model's accuracy and speed while reducing the size. We use several visualization models to increase the accuracy after pruning. Neteron and Lucid can output the visualization picture for our model, which can help us to improve accuracy. The ANN visualization can visualize the whole model layer to a pdf file. We use the pdf to see which layer can be optimized to increase accuracy. Those several projects make the prune part have a huge improvement about reducing size.

FPGA:

We first learned how to train our data on a different algorithm(BNN). Then we learned how to use Xilinx tools Vitas and Vivado by implementing a sample project. For the future works, we will work on how to connect DPU(DPU is an existing design of FPGA project) to our board. There requires lots of configurations and knowledge to implement this part. We will continuously work on it.

Data pre-manipulation

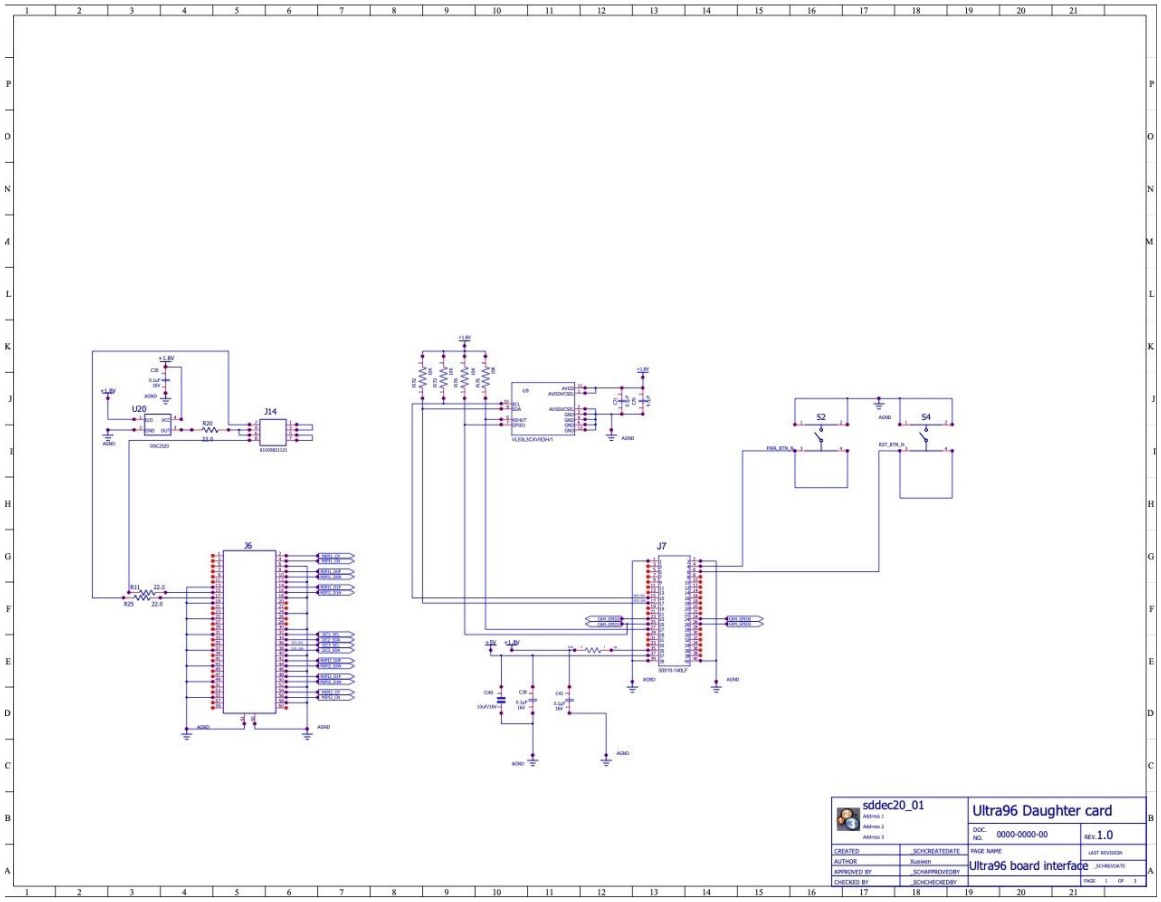
At the beginning we change our original data to 28x28 b&w and 36x36RGB, and do the test, we found that compared with original data, the pre-manipulated data reduces latency. Then we start a grid search on which types of images would be the best benefit for our algorithm. For now, we have tested edge-detection images , over explored images, noise filtering images and sobel edge detection images. For now the overexploited images work well, but we will keep looking for manipulating ways to find the best result.

We have a group meeting with our client each week, and if we have some unsolved problem, we will make a group meeting on the weekend, at the end of this semester we at least have 15 meetings in total. And we want to make sure that everybody in our team is on the same stage, so we explain our work, discovery, studies to our teammates every week.

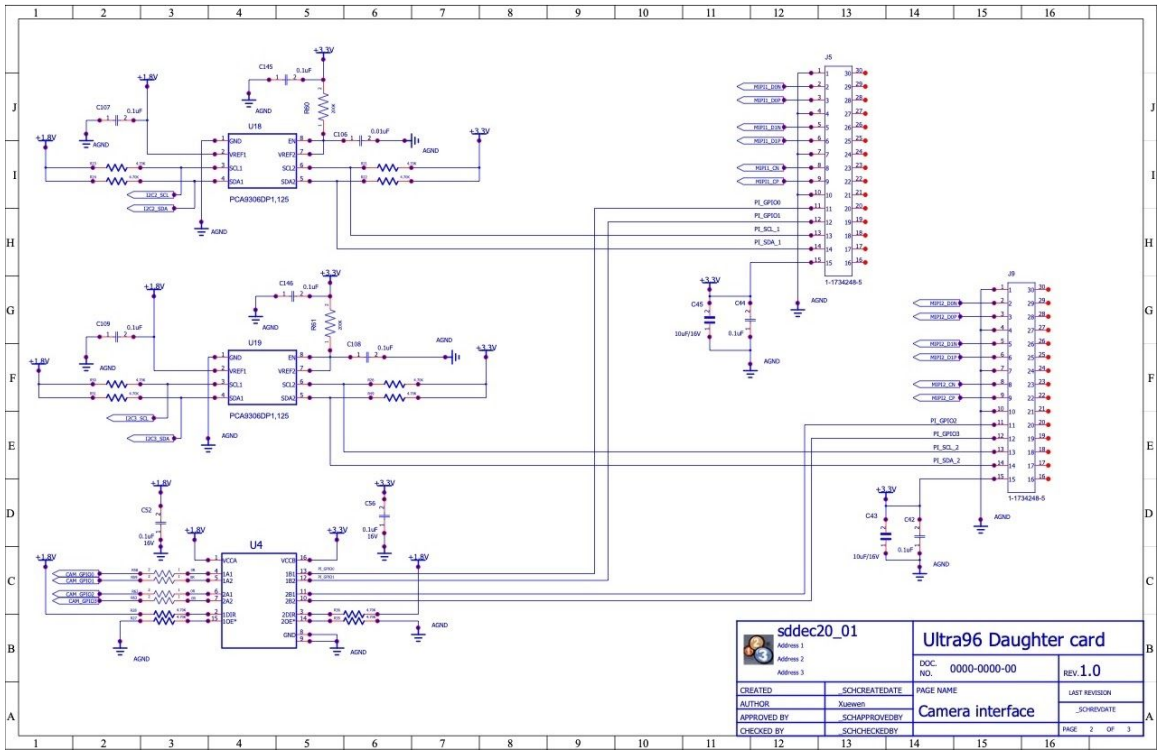
Daughter Card Design:

It was a little difficult for us to start on the daughter card design because we did not have experience on the PCB design. So all the things we need to learn by ourselves. We learned how to use the tool PCB 123. Although we have the prototype of the MIPI adapter, we want to add a ToF module (Time-of-Flight ranging sensor with multi target detection, on figure[2]U9) in order to increase the accuracy of the detection. Also, some functions we did not want to use on the prototype would change or delete during the design. We have 2 schematic reviews and 1 layout review during the design process to make sure the daughter card was designed correctly. We have several versions of the schematic and the layout, and here are the final versions of the design. We send the final design to the industry and waitting for the board and testing.

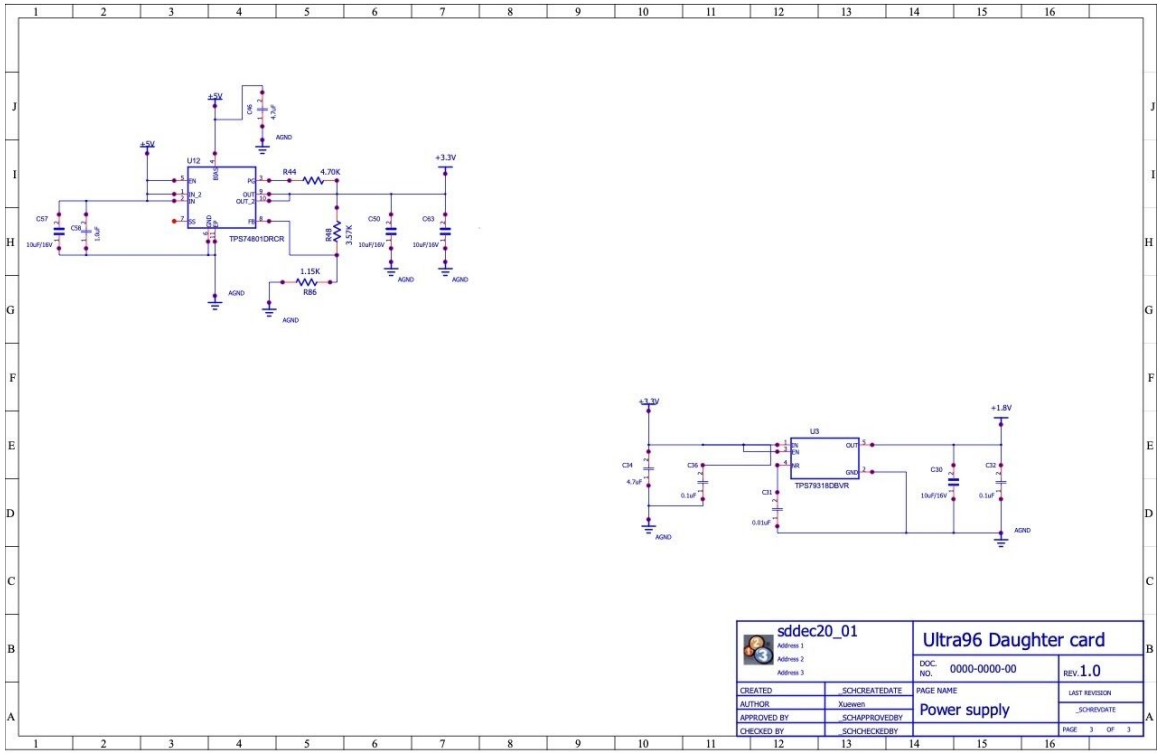
figure[2],[3] and [4] are the schematic design by Xuewen Jiang.


sddec20_01
Ultra96 Daughter card
0000-0000-00
1.0
Ultra96 board interface
U20
J14
J6
J7
S2
S4

figure[2]

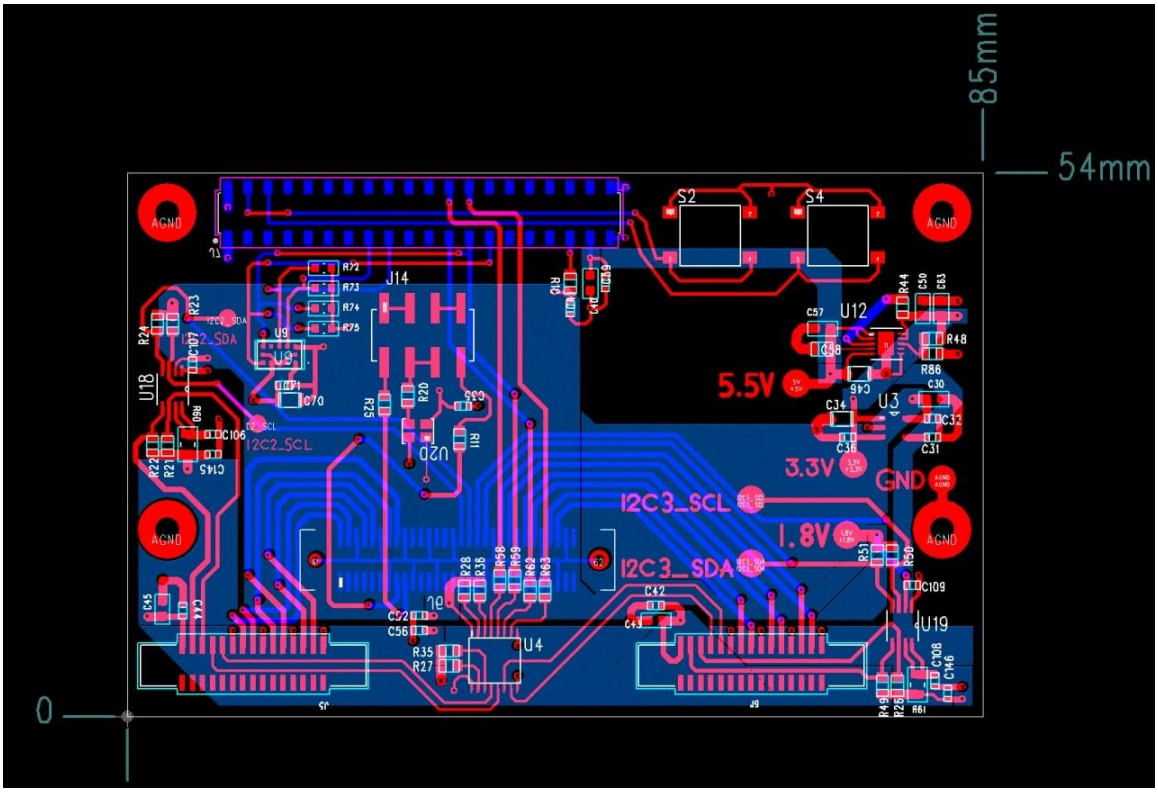


figure[3]



figure[4]

Layout designed by Jianhang Liu and Kazambe Isaac



figure[5]

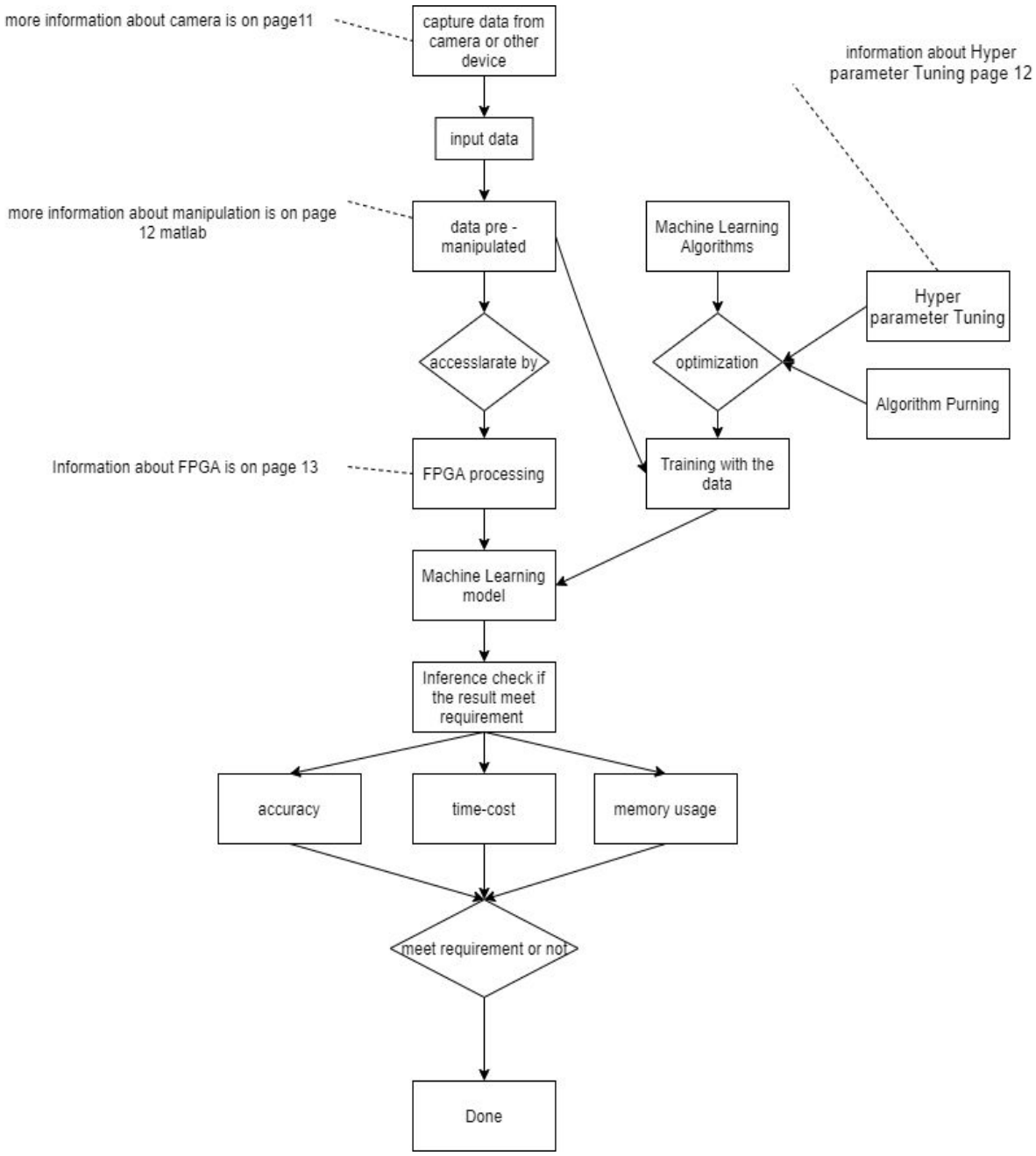
BOM for the Daughter card design

Link:https://drive.google.com/open?id=1mXkfkNlMq9EA_Je-l3QsiqfseRGwOratzknDpmXhS5Y

The metrials based on the AISTARVISION MIPI Adapter V2.0								datasheet
Schematic :https://github.com/Kevin-WSCU/96Boards-Camera/blob/master/AISTARVISION%20MIPI%20Adapter%20V2.1/AISTARVISION%20MIPI%20Adapter%20V2.1.pdf								
Daughter card for Ultra96		Product Type	reference designator	Quantity	Cost($)	Total	web to buy	
clock selection: clock from adapter board, 96 Boards or sensor board								
2520 standard clock oscillators	OSC2520	Clock Oscillators	U20	1	1.47	1.47	https://www.mouser.com/ProductDetail/Kyocera-Electronic-Components/MC2520Z800000C19XSH?qs=GBLSl2AkiruYmi3baNLTwA%3D%3D	
Clock Source Selection 2.54mm Pitch header	61000821121	Headers	J14	1	1.72	1.72	https://www.digikey.com/en/products/detail/w%C3%BCrth-elektronik/61000821121/4846967?s=N4IgTCBcDaIGwEYAMKAcYElyAugKyA	
switchs								
LFS C&K switchs tactile switchs	PTS645SH50SMT	Tactile Switches	S2, S4	2	0.24	0.48	https://www.mouser.com/ProductDetail/CK/PTS645SH50SMTR92LFS?qs=%2Fha2pyFaduime26pVejlh2oyXnoglpe5bAY3bnaSg5fX3hPgLBbAH5%2FZCsoluaw	
Connect to ULTRA 96 LS and HS expansion								
To LS CONN of 96Boards	55510-140LF	Header	J7	1	4.6	4.6	https://www.digikey.com/en/products/detail/amphenol-icc-fci/55510-140LF/1090236?s=N4IgTCBcDaIKwIwAYC0iAsyAyAxEAugL5A	
To HS CONN of 96Boards	61083-063402LF	Header	J6	1	5.93	5.93	https://www.digikey.com/en/products/detail/amphenol-icc-fci/61083-063402LF/1001993?s=N4IgTCBcDaIGwEYAMAOAzAWXNAWJYAMgGIgC6AvkA	
Raspberry PI camera Interface								
	PCA9306DP1,125	Translation-Voltage Levels	U18,U19	2	0.68	1.36	https://www.digikey.com/en/products/detail/nxp-usa-inc/PCA9306DP1125/1642395?s=N4IgTCBcDaIAoGECCBOAzABgGwBE4EYAafMAVhAF0BfIA	
	74AVC4T245PW,118	Translation-Voltage Levels	U4	1	0.41	0.41	https://www.digikey.com/en/products/detail/nexperia-usa-inc/74AVC4T245PW118/2296477	
Mate with PI camera 1	1-1734248-5	FFC & FPC Connectors	J5,J9	2	1.06	2.12	https://www.digikey.com/en/products/detail/te-connectivity-amp-c	https://www.te.com/commerce/DocumentDelivery/DDEController?Action=srchrtrv&DocNm=1734248&DocType=Customer+Drawing&DocLang=English&PartCntxt=1-1734248-5&DocFormat=pdf
Power supply								
3.3V for Camera Board	TPS74801DRCR	5V to 3.3V linear regulator, 1.5A	U12	1	1.56	1.56	https://www.digikey.com/en/products/detail/texas-instruments/TPS74801DRCR/1426916	
1.8V IO Power	TPS79318DBVR		U3	1	0.56	0.56	https://www.digikey.com/en/products/detail/texas-instruments/TPS79318DBVR/411990	
Time-of-Flight ranging sensor with multi target detecti	VL53L3CX		U9	1	4.85	4.85	https://estore.st.com/en/products/imaging-and-photonics-solutions/proximity-sensors/vl53l3cx.html	
					Total: 20.21+4.85 = 25.06	25.06		
Bill of metrials for Ultra96								
http://zedboard.org/sites/default/files/documentations/AES-ULTRA96-G%20BOM%20REV%201%20180315.pdf								
4 Mounting Hole								
*Y = Custom made symbol/footprint								
BOM for capacitor								
material: Ceramic								
Type		link	reference designator	how many	price$	Total		
0.1uF 16V 0603	C0603C104M4RACTU	https://www.digikey.com/product-	C109,C35,C39, C41, C1	16	0.1	1.6	1.6	
10uF 16 V 0603	GRM188R61C106KAALJ	https://www.digikey.com/products	C40, C43,C45,C30,C57	7	0.37	2.59	2.59	
0.01uF 16V 0603	C0603C102J4RACTU	https://www.digikey.com/en/produ	C106, C108,C31	3	0.39	1.17	1.17	
4.7uF 16V 0603	GRM188R61C475MAAJD	https://www.digikey.com/products	C46,C34,C70	3	0.34	1.02	1.02	
1uF 16V 0603	GRT188C81C105ME13D	https://www.digikey.com/en/produ	C58	1	0.33	0.33	0.33	
					6.71	0	0	
BOM for resistor						0	0	
22 Ohms 1/4W	CRGP0603F22R	https://www.digikey.com/en/produ	R20, R11, R25	3	0.14	0.42	0.42	
0R 1/8W	CRCW08050000Z0EA	https://www.digikey.com/en/produ	R18,R58,R59,R62,R63	5	0.1	0.5	0.5	
4.7K Ohms 1/8W	CRCW08054K70FKEA	https://www.digikey.com/en/produ	R23,R24,R21,R22, R50	12	0.1	1.2	1.2	
1.15k 1/10W	CRCW08051K15FKEA	https://www.digikey.com/en/produ	R86	1	0.1	0.1	0.1	
3.57K 1/10W	CRCW08053K57FKEA	https://www.digikey.com/product-	R48	1	0.1	0.1	0.1	
200k 1/6W	CRCW0805200KJNEA	https://www.digikey.com/en/produ	R60, R61	2	0.1	0.2	0.2	
10k	RE0603FRE0710KL	https://www.digikey.com/en/produ	R72,R73	2	0.1	0.2	0.2	
1K	RE0603FRE071KL	https://www.digikey.com/en/produ	R75, R74	2	0.1	0.2	0.2	
					Total: 9.63	9.63	9.63	
					Total Price for parts: 25.06+9.63=34.69			
From PCB123 prediction		Quantity	Unit Price	Thickness	Total			
PCB board	4 layer	2	198.89	0.062"	397.78+34.69*2=467.16			
solder mask (enables silk screen)		4	101.08		543.08			
JR: click mezzanine link	https://www.avnet.com/shop/us/products/avnet-engineering-services/aes-acc-u96-me-sk-3074457345638713309/							
FPC 1MM THROUGH HOLE TE	https://www.te.com/usa-en/product-1-84984-6.html		https://en.maritex.com.pl/connectors/zif_and_non-zif_connectors/zif_connectors/zif_connectors_vertical_for_through-hole_mounting_1_00mm_pitch.html					
FPC 1MM through hole molex	https://www.mouser.com/ProductDetail/Molex/52806-1510?qs=wUVoC1r15A%2F%252BUg72%2F4W6g%3D%3D							
TO220 3.3V regulator, 800mA	https://www.digikey.com/en/products/detail/texas-instruments/LM1117T-3.3/2602474							
TO220 3.3V regulator, 1A	https://www.digikey.co.za/product-detail/en/texas-instruments/LM3940IT-3.3-NOPB/LM3940IT-3.3-NOPB-ND/308006							

Table[2]

2.4 Conceptual Sketch



figure[6]

This figure shows how our project works. First we get data from a camera interface, we would use two cameras, one is a MIPI camera the other one is a USB camera. For the camera selection, one of our teammates did a grid search and we did a group trade study for several selections of camera. Then we would test which format of image is best for the ML algorithm. For example 28x28 black

and white image, 36x36 RGB image, 28x28 overexposure image, edge detection image. for this step we are still in a search stage, we are still looking for other image manipulation methods that could work best for our ML algorithm, for know we got the data like memory usage latency and accuracy for 28x28 black and white image, 36x36 RGB image, 28x28 overexposure image, edge detection image. The FPGA part, we use DPU(deep learning process unit) to do complex matrix math for the algorithm in order to improve the performance of the algorithm. For hyper-parameter, we did a lot of tests via different parameters for the algorithm such as layers in neural network, epoch, batch size, optimize algorithm. After finding the best parameters, we want a pruning for the algorithm. then we will test if the algorithm meets our metrics based on accuracy, latency and memory usage.

3. Statement of Work

3.1 Previous Work And Literature

(Include relevant background/literature review for the project

– If similar products exist in the market, describe what has already been done

– If you are following previous work, cite that and discuss the **advantages/shortcomings**

– Note that while you are not expected to "compete" with other existing products / research groups, you should be able to differentiate your project from what is available

Detail any similar products or research done on this topic previously. Please cite your sources and include them in your references. All figures must be captioned and referenced in your text.)

BackGround:

-- Although floating points are a good choice for handling the small updates that during Convolutional Neural Network training, the resulting parameters can contain too much redundant information.

-- BNN, in which some of the arithmetic involved in computing outputs are represented in only a single bit. Kim and Smaragdis[1] has published their work on the full binarization with a preset portion of the synapse having zero weight, and all other synapses with a weight of one. They report 98.7 % accuracy with a fully-connected network on MNIST dataset.

-- DoReFa-Net by Zhou et al. [2] explores reduced precision utilizing both the forward pass and the backward pass, he observes that this opens opportunities for training the neural networks on FPGAS. Their results include the best-case ImageNet Top-1 accuracies of 43% for full and 53% for partial binarization.

-- Major advantages of using BNN is the drastic improvement in efficiency. However, we can also expect a large amount of drop in accuracy. For our project, we can utilize the inference on the FPGA with either full or partial binarization.

3.2 Technology Considerations

--- Major strengths using quantization, pruning and hyper-parameter tuning is the ability to inference at a faster speed, also taking less memories at the same time.

--- However, this could somehow affect our inferencing accuracies with unseen data. We are also facing the classic accuracy-computation tradeoffs.

--- One alternative is using k-fold validation to validate with our pruning, quantization, image-pre manipulation and hyper-parameter tuning. So the test set can give unbiased feedback.

--FPGA acceleration. For our inference step, it will require a lot of CPU resources to do the complex math for inference. We will use FPGA to do the math and send the result back to the CPU in order to accelerate the algorithm.

3.3 Task Decomposition

-- Image pre manipulation

apply different filters on input images and adjust its format (e.g. grayscale, color, resolution) to find the method that can benefit the system most.

-- Hyper-parameter tuning

using different parameters for the algorithm such as filter number filter layer hidden layer , we want to find the best number for those configurations.

-- pruning

understand each layers in NN and remove some of them in order to accelerate the algorithm

-- Quantization of weights

a way to manipulate data string in algorithm in order to reduce workloads of the algorithm

-- FPGA design for acceleration

FPGA can do parallel computation, which will be much faster than a CPU

The tasks are taking different angles in effort to reduce the operations needed for the inference. On the training side we pre manipulate images, fine tune the layers, pruning the unnecessary connections; On the infereing side, we quantize the neural network and map onto the FPGA.

3.4 Possible Risks And Risk Management

1.Coronavirus :

Coronavirus happens in January and widespread in the United States in March. We cannot predict the virus widespread in such a bad situation, but we should think of such risk earlier and get the solution for that.

1)Risks: Delaying our camera daughter card. We cannot get the camera adapter card because all factories are closing during the coronavirus.

Risk solution: Design our own daughter card by ourselves for the Ultra 96.

2)Risks: Group working. It is hard to meet each other in the lab because the school is closing.

Risk Solution: We have group meetings online and get video chatting every week to discuss our process.

2. Reducing computation may influence our inference accuracies

3.5 Project Proposed Milestones and Evaluation Criteria

-- Key milestones: Delopying algorithm on FPGA;

Obtain baseline algorithm metrics(memory,speed,accuracy)

Improve algorithm to surpass baseline metrics

Discover some better image filters for processing

-- Tests: Run the modified algorithm on ULTRA96-v2(FPGA) with the same measurement of time, memory, accuracy. Comparing the newly obtained results with baseline metrics.

3.6 Project Tracking Procedures

-- GANTT chart. We use the Gantt chart to manage our project timeline and working status.

-- Google Doc. We use Google Doc to share our work and files.

--Weekly meeting. We have group face-to-face meetings every week (when the school is still open).

--Telegram Chatroom. We use Telegram to have a weekly meeting to share our working process with group members and clients.

3.7 Expected Results and Validation

Our desired outcome:

-- Improve speed by 50 %

-- Keep the accuracy above 97 % on unseen data

--Less size

--For Hardware, finish Daughter card design on PCB123, then order the meterials.

We will confirm that our solutions work at a **High level** with:

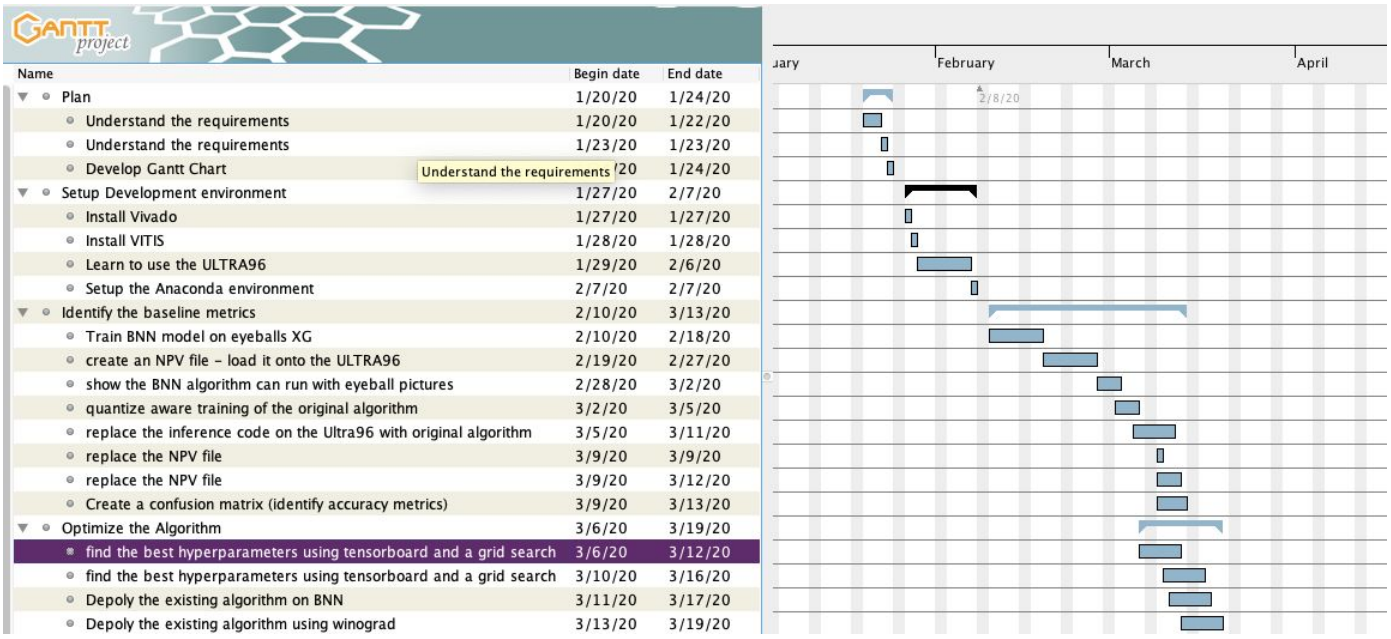
-- Comparing the newly obtained metrics with baseline metrics; on the same dataset

-- Contact clients or advisers often to get suggestions on the work process

4. Project Timeline, Estimated Resources, and Challenges

4.1 Project Timeline

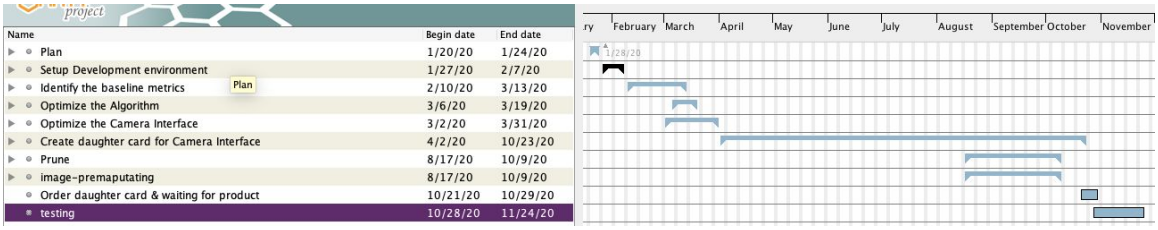
Project timeline for 491



figure[7]

-- paragraphs: The timeline is being proposed to be agile and adaptable to changes as we get familiar with the technology and may want to experiment/follow different methods to achieve our goal and capture results based on a well-defined guideline, and obtain our result(metrics) in a reliable manner.

Project timeline for 491 and 492



figure[8]

We continued our project process on 491 and tried to finish it.

4.2 Feasibility Assessment

Realistic projection of what the project will be. State foreseen challenges of the project.

-- We will make some significant improvement on latency, memory usage while keeping a pretty good accuracy.

4.3 Personnel Effort Requirements

Task	Explanation	Requirements
Pruning	Prune the network	143 hrs
Hyperparameter Tuning	Fine tune the network	144 hrs
Sparse Filters	Using sparse filters	147 hrs
FPGA design	FPGA hardware design	143 hrs
Camera interface Trade Study	Study trade off on camera options	30 hrs
PCB Design for the camera interface daughter card	Design a adapter between Ultra 96 and camera interface	100 hrs
BNN implementation of algorithm	Binarize the weights for algorithm	143 hrs
Camera FPGA integration	Integrate Camera with FPGA	144

table [2]

Pruning requires a lot of tests, for pruning our goal is delete those layers in an algorithm which may have no impact or little impact for the result, unfortunately we don't know which layer satisfied, so we have to test and compare the results which would cost a lot of time.

Hyperparameter is like how we configure the algorithm, same as pruning we have to find best numbers for each parameter, it's another grid search.

FPGA design, for our team, we are all new to FPGA. So our client would lead us to do this part. To start FPGA design, we studied several tools like Vitas, Vivado. The final goal for FPGA is to use a DPU do complex math calculation for the CPU on ultra-96

PCB daughter card design, because it is our first time to do the PCB design so we need more time to learn and use the PCB123, choosing the parts we need for the design.

4.4 Other Resource Requirements

-- Camera Daughter card. We are planning to use the MIPI camera interface for the Ultra 96, so we need to have a camera interface adapter card to connect those two boards.

-- Ultra96-v2

-- Xilinx License

4.5 Financial Requirements

-- Financial support for hardware purchase. All the purchase will be agreed by the client and get it from him.

Ultra 96-v2

USB camera: Logitech C920s.

MIPI cameras: Two Raspberry PI cameras OV5647 for catching images.

Camera daughter card: 96Boards DUAL MIPI Adapter Mezzanine - AiStarVision

Since we cannot get the daughter card so we will need the parts and PCB when we finish the daughter card design.

5. Testing and Implementation

Needed test for our project includes:

Unit testing for modules, integrity testing for interfaces, user-study for functional and non-functional requirements

1. Individual items to be tested for our project includes:
a. Ultra96-v2 functionality test
b. Software-hardware integration test
c. Hardware test include daughter card and camera interface
d. unit testing for pruning, quantization

Some actual test cases include unit testing for each of our software components, for instance we have thoroughly tested each of our components, ie the part for metrics measurements, quantization such before we merge everything together for integrity testing.

Challenges:

Unstable environment.

Cannot get the hardware such as the daughter card because of the coronavirus.

Cannot get the order from industry for our daughter card design. The Time is longer than expected.

5.1 Interface Specifications

Hardware interfacing testing: ultra96-v2 board, MIPI Camera, camera daughter card

Software interfacing testing:

a. IP network protocols, software drivers for the peripheral devices
b. Compatibility test with different software modules kears, tensorflow, tensorflow-lite on Coretext-arm53 system etc
c. Software hardware compatibility test with our modified bitstream on the existing FPGA overlay provided by Xilinx

5.2 Hardware and software

-- software drivers for ultra96-v2, this is useful because we want to make sure drivers is not causing us connections issues

-- IP network protocols for communication between host and embedded development board, this is useful because we want to eliminate possibility of bugs that may introduced by protocols communication issue

-- MIPI camera is useful because we can drive the MIPI camera to do the data pre-manipulation prior to the algorithm.

-- Camera daughter card is an adapter to connect Ultra96 and the MIPI camera interface.

5.3 Functional Testing

--- Unit test for software that measures metrics
--- Integration test for running the metrics on ultra96 ARM processor
--- Acceptance testing for running new algorithm and make sure the metrics is better than the baseline
--- Cross-validation test with k-fold on all datasets and calculate average loss for each fold
--- Split datasets into train, validation and test; conceal test dataset to prevent human bias added into the algorithm from hyper-parameter tuning etc

5.4 Non-Functional Testing

Testing for performance, security, usability, compatibility

--- Performance testing for keep the memory usage low

--- Compatibility test for running frameworks that support ARM53 processor architecture

----Scalability test: Classify images on eyes with different shapes/colors

---- Usability test: Mock the mounted camera on helmet environment and test usability of the system

5.5 Process

-- We tested the software drivers by connected the ultra96 to different host computer and make sure the opperatate driver was supported in different operating system

-- we tested different connectivity protocols utilizing different connection protocols like uart, usb etc.

-- Camera setting: We first use the camera on the laptop for testing the code is working. We can catch our eye in the camera and run the algorithm in the jupyter notebook to analyse if the eye is open or closed. Then we run it on the Ultra 96 and connect with the USB camera and it works.

-- Daughter card: Because we did not get the daughter card from the industry, so we did not have a chance to test it. We use knowledge and a calculator to do the self-testing during the PCB design. Make sure all the capacitor, resistor and parts will work when we test on the Ultra 96.

5.6 Results

--- All connect and drivers test indicates successes

--- What I learned is that we should stuck with the official supported operating system and software so the likelihood of running to unknown bugs are low

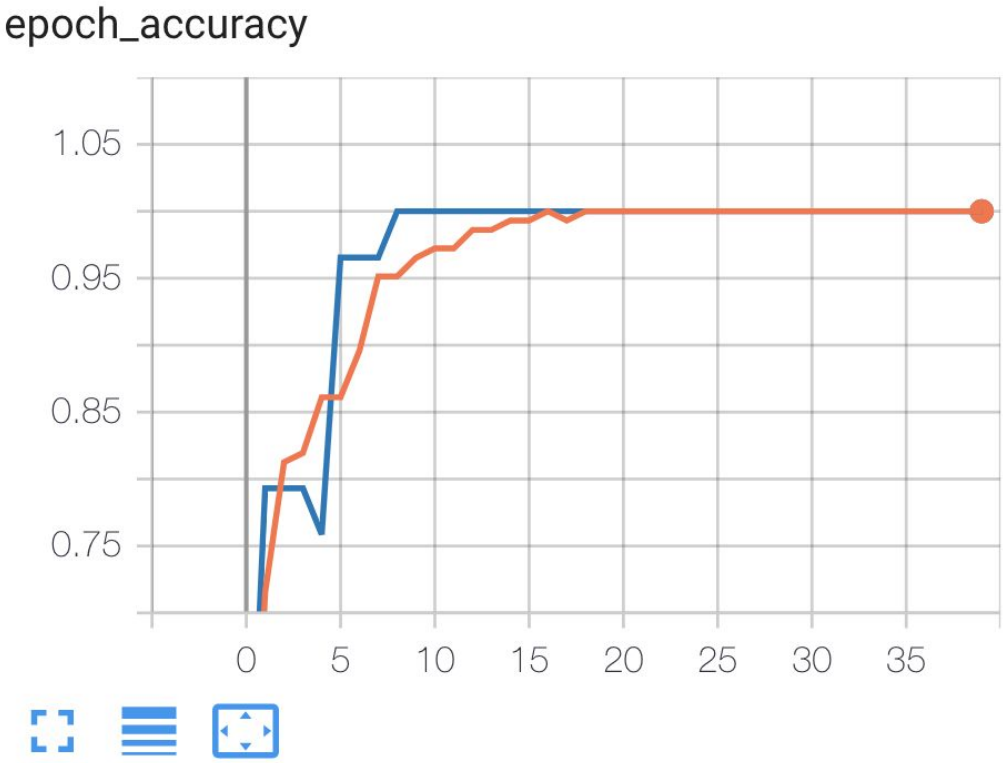
--- One implementation issues is that some of the existing frameworks supports older version of Xilinx tools that has been archived

---During training, The image input to the model is as important as the model base. In the testing, we input different size images to training and got various data. The test result shows we need to focus on input image size and dimension.

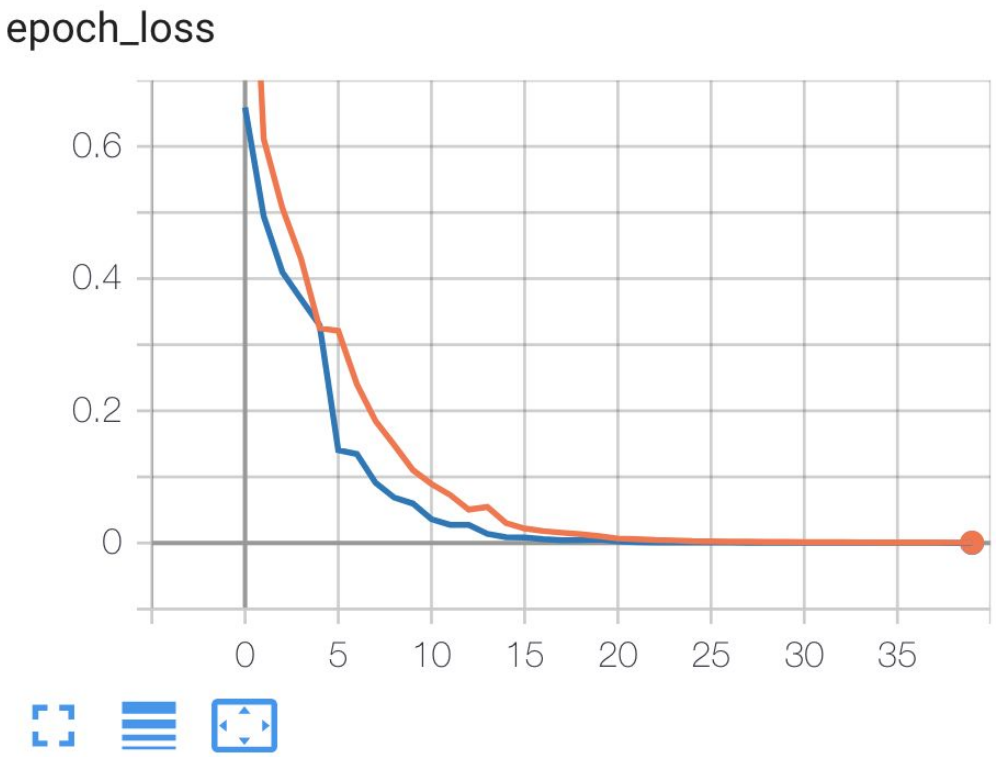
---We need to focus on which tool is best for the specific task, the most popular tool may not be useful, but the unknown skill makes enormous improvements.

figure[9] and [10]

[result for pruned model, we can observe significant reduction in latency]


epoch_accuracy
1.05
0.95
0.85
0.75
0
5
10
15
20
25
30
35

epoch_loss



accuracy for Learning rate current model is : 95.3125 %
mean latency per image = 0.03 seconds = 26513.91 microseconds

6. Closing Material

6.1 Conclusion

--- We have set up the metrics, improved the algorithm by more than 80% in terms of latency, some more improvement can be done in terms of accuracy.

--- Explore hardware acceleration and utilizing the FPGA advantage to harvest more computation power and improve our algorithm further

--- For the goal about smaller model size, we already finished part of the goal and made the size model much less than before. After that, we think we need to mix the pruning model with others' work and try to increase the accuracy in the next step. Since we not only need to decrease one model size, we will also improve other models in the future step.

--We finish the daughter card PCB design, and start on order the materials and boards. We did nott get the daughter card in the end so we did not have a chance to test it. The work will be continued to the next team.

6.2 References

[1] M. Kim and P. Smaragdis. Bitwise neural networks. [26] CoRR, abs/1601.06071, 2016.

[2] S. Zhou, Z. Ni, X. Zhou, H. Wen, Y. Wu, and Y. Zou. DoReFa-Net: Training low bitwidth convolutional neural networks with low bitwidth gradients. CoRR, abs/1606.06160, 2016.

[3]Design by GanttProject 2.8.11.

[4]Gskielian. “Gskielian/JPG-PNG-to-MNIST-NN-Format.” *GitHub*, github.com/gskielian/JPG-PNG-to-MNIST-NN-Format/blob/master/convert-images-to-mnist-format.py.

6.3 Appendices

Design document:

https://sddec20-01.sd.ece.iastate.edu/docs.html

Current situation for the PCB daughter card design:

Bill of materials and requirements:
https://drive.google.com/open?id=1mXkfkNlMq9EA_Je-l3QsiqfseRGwOratzknDpmXhS5Y

Camera interface trade study:

https://drive.google.com/open?id=1i7nz3Z4Ww1dShxYpAawH-zAwqsWl7XHK