AgriAI

Case Study :

In modern Agriculture and Forestry purposes Plant Pathological Research is very much essential for the crop improvement. Every year, huge amounts of crops are damaged by pathogen attack. In most of the cases, farmers and lay men suffer huge loss due to severe pathogenic infestation. For crop improvement, the first and foremost essential key point is disease identification. After proper identification of diseases we will be able to treat the diseases. Most of the cases lab oriented research is time taking and very much expendable but instant identification of plant diseases with its causal organism by using modern technology is most suitable and cost effective and also authentic. This type identifying methodology will help us to identify the diseases with its proper control measures. Above 95 % accuracy and authentic identification is possible by using this methodology.

In general, the impact of pesticides consists of the effects of pesticides on non-target species. Pesticides are chemical preparations used to kill fungal or animal pests. Over *98%* of sprayed insecticides and 95% of herbicides reach a destination other than their target species, because they are sprayed or spread across entire agricultural fields.Runoff can carry pesticides into aquatic environments while wind can carry them to other fields, grazing areas, human settlements and undeveloped areas, potentially affecting other species. Other problems emerge from poor production, transport and storage practices.Over time, repeated application increases pest resistance, while its effects on other species can facilitate the pest's resurgence.

Each pesticide or pesticide class comes with a specific set of environmental concerns. Such undesirable effects have led many pesticides to be banned, while regulations have limited and/or reduced the use of others. The global spread of pesticide use including the use of older/obsolete pesticides that have been banned in some jurisdictions, has overall increased .

Thus we can conclude that even though pesticides and insecticides are pretty useful but nowadays the reverse of it i.e. harmfulness is getting increased at a rapid rate.

Problem :

So the point arises - " What can we do to control usage of pesticides ? ".

Over here we should keep one thing in mind is that we need to make a system which can lower down or even maybe track down the usage of pesticides cause we can't pull it down for some obvious reasons to maintain the agriculture industry.

Solution :

Proposed Theory :

In our project we will be using the feature and characteristics of being warm-blooded mammals called thermoregulation.
So basically we will scan up our environment using the AMG8833 Sensor Camera breakout using the balenaFin/Raspberry Pi 4B and it will show up whether the pest is present there or isn't. And it will then work according to the Python script that controls two Servo Motors to close/open the lid of the Pesticide bottle.

Experimental Work :

Hardware & Software Used :

Hardware :-

- balenaFin/Raspberry Pi 4B
- NVIDIA Jetson Nano (for future upgrades)
- Raspberry Pi Camera V2

Software :-

- Edge Impulse Studio
- TensorFlow Lite & Edge Impulse’s EON Compiler (MobileNet V2 SSD 0.35)
- WebAssembly Package Generated from Edge Impulse Studio
- balenaOS
- GitHub Actions
- Jupyter Notebook
- Docker Containers(for pushing code to balenaOS)

Selection of optimised ML models :-

Since we are using Edge Impulse which is a web-based IDE that simplifies developing complex machine learning models for Edge/Embedded devices, even for MCU’s that run in a few mW’s , we decided to use Google’s TensorFlow (TensorFlow Lite, to be specific) for the development of our model rather than using Facebook’s PyTorch as because TF Lite is extremely lightweight as compared to PyTorch and even uses very less amount of CPU

space. As for our end product we are only using CPU-only based machines so we decided to go for nothing else than TensorFlow Lite.

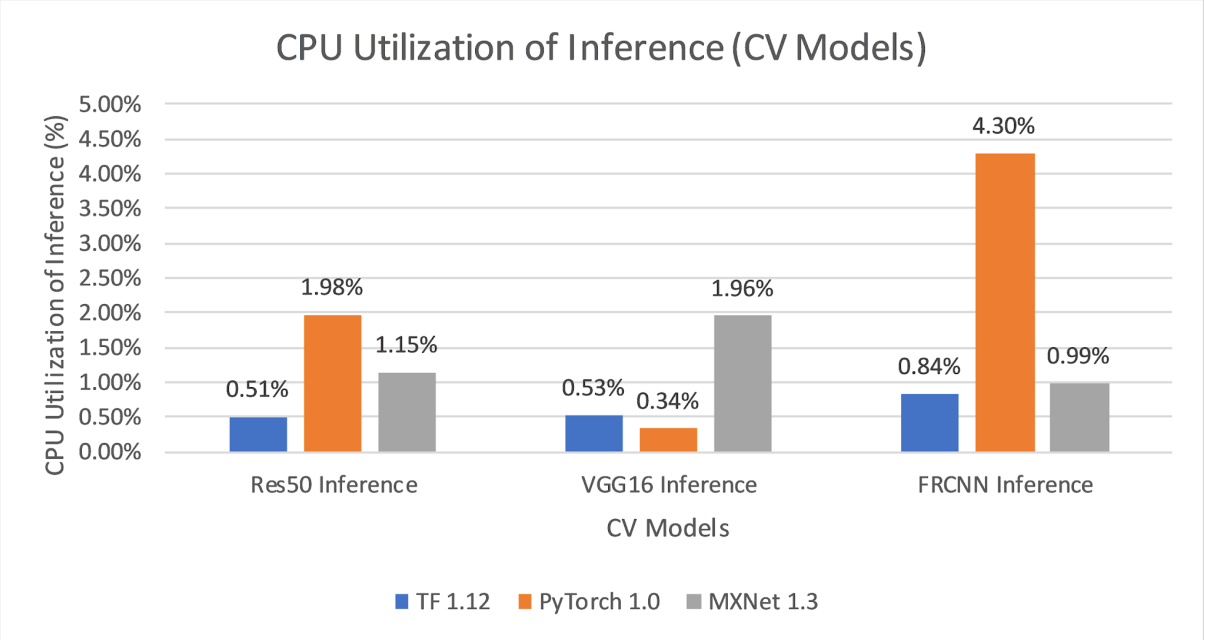


Fig 1. CPU inference by various versions of TensorFlow, PyTorch and Microsoft's MXNet.

Edge Impulse allows us to use these following models for our project development :

Fig 2. Types of ML models available for usage from Edge Impulse.

We went on to use MobileNet V2 0.35 as it provides us a far greater amount of accuracy as compared to other ML models while even taking less than a few megabytes of model data size and even has classification ability for both RGB and Grayscale images.

According to the MobileNet V2 paper by Google authors we even get to see comparison graphs like these :

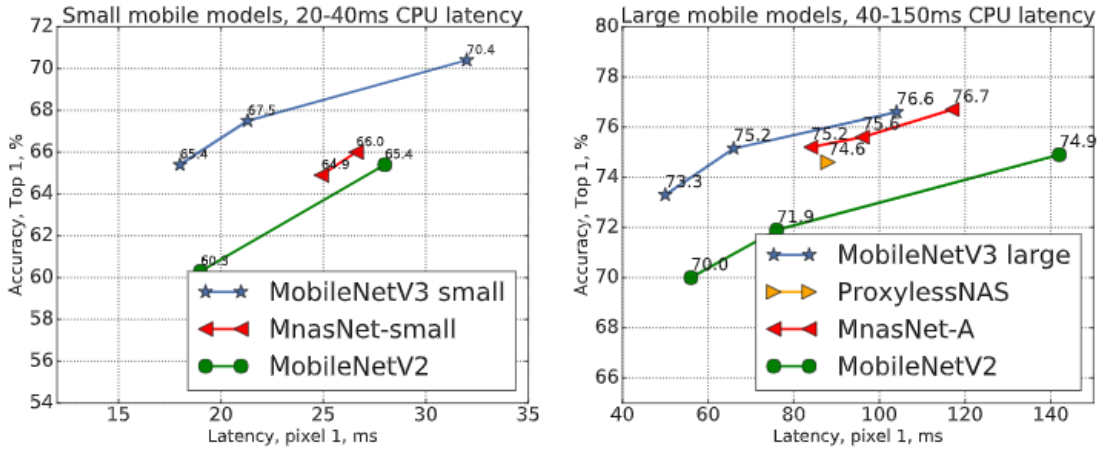


Fig 3. MobileNet V2 accuracy and latency rates in large and small CPU-only models.

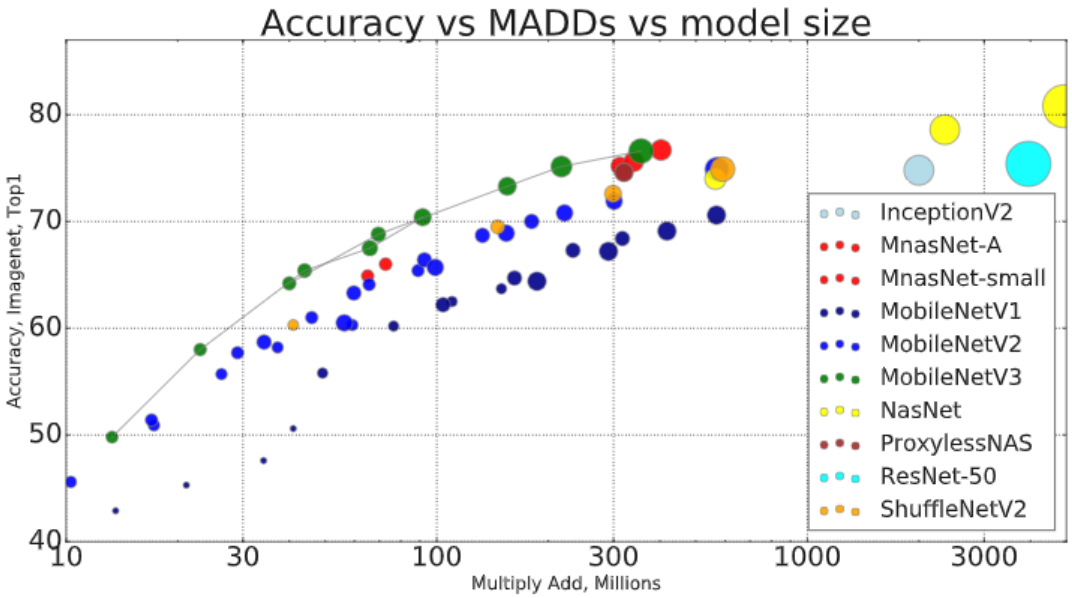


Fig 4. MobileNet V2 vs popular model MAC's/MADD's rate. Mobilenet V2 uses ~300MMadds while achieving gross accuracy of about 72%.

Thus the above comparison charts led us to the fact that usage of MobileNet V2 will be the best use case for developing our model.

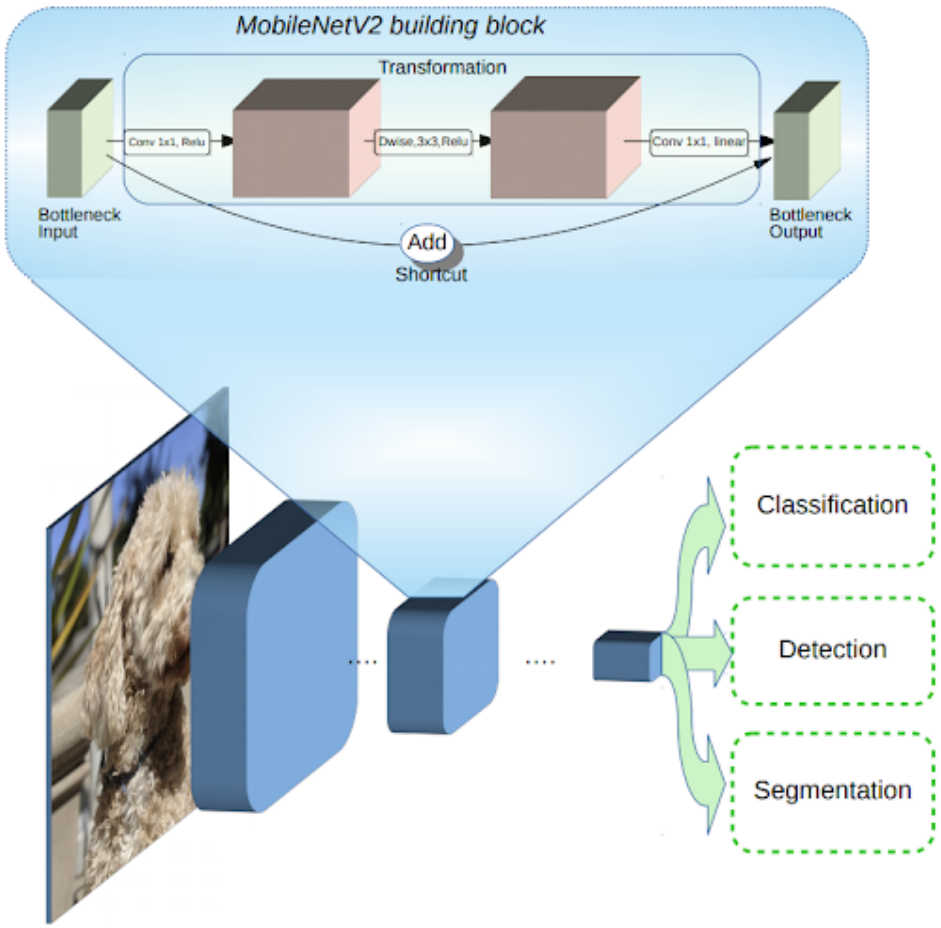


Fig 5. Google's MobileNet V2 architecture.

Development of the ML model :-

Step 1 :-

We have collected thermal imagery data for both butterfly and damselfly (for this prototype) of about 1200 of them which includes around 620 training images of damselfly and 600 for butterfly.

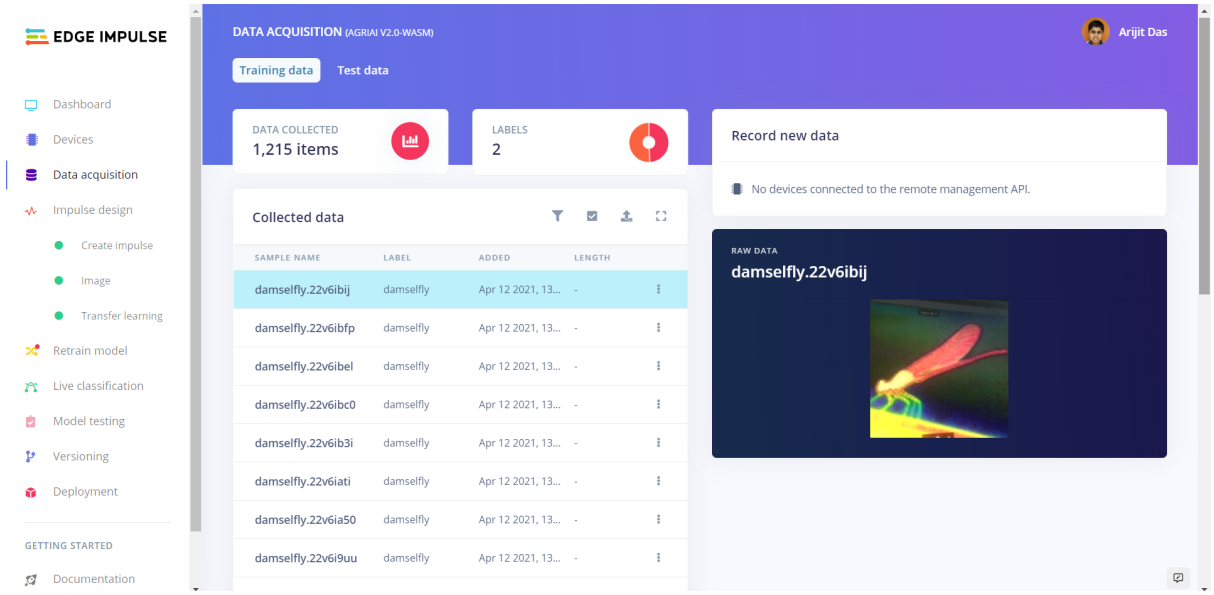


Fig 6. Data Acquisition page in Edge Impulse Studio.

Step 2 :-

Then we have gone on designing our Impulse (aka the ML model) as shown in the pictures laid down below.

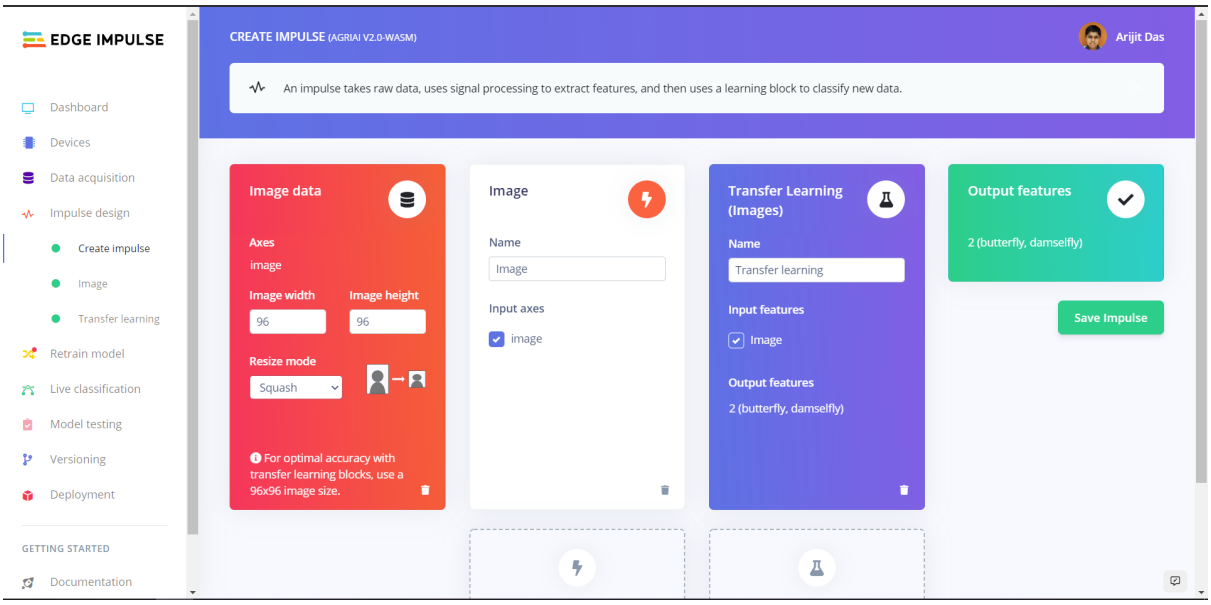


Fig 7. Designing our Impulse in the EI Studio.

Step 3 :-

After creating our impulse we have started training the ImageNet block which lays out the following conclusions after generating features out of the provided testing images :

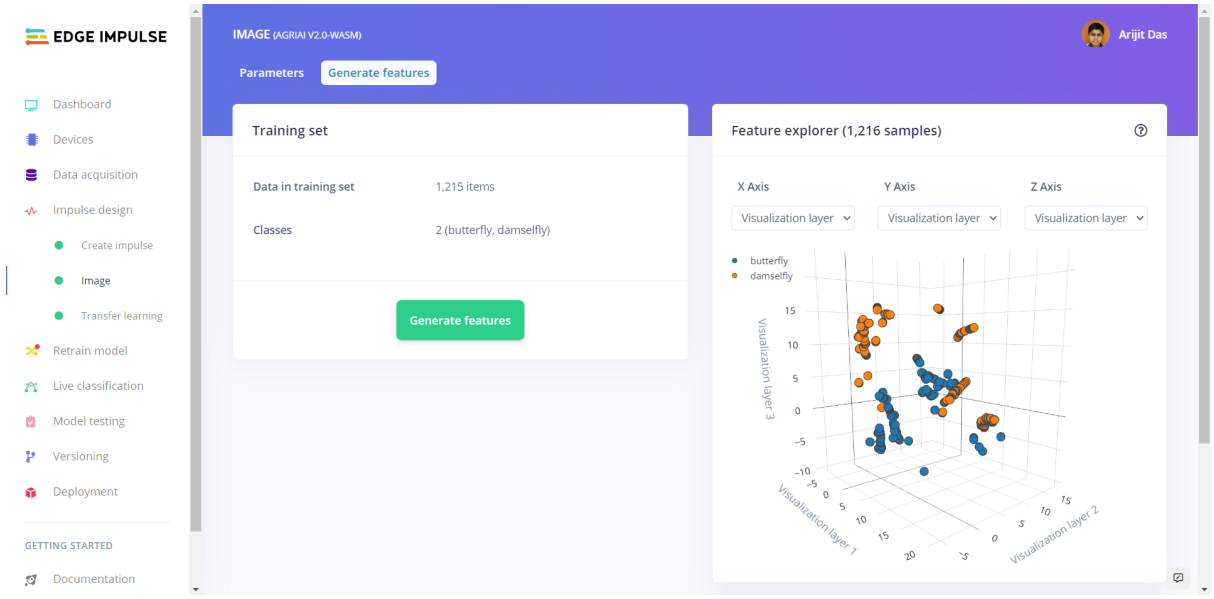


Fig 8. Image block in EI Studio after Feature Generation.

Step 4:-

Now we proceeded to the main step of training our model using Transfer Learning. Transfer learning is a research problem in machine learning that focuses on storing knowledge gained while solving one problem and applying it to a different but related problem. For example, knowledge gained while learning to recognize cars could apply when trying to recognize trucks.

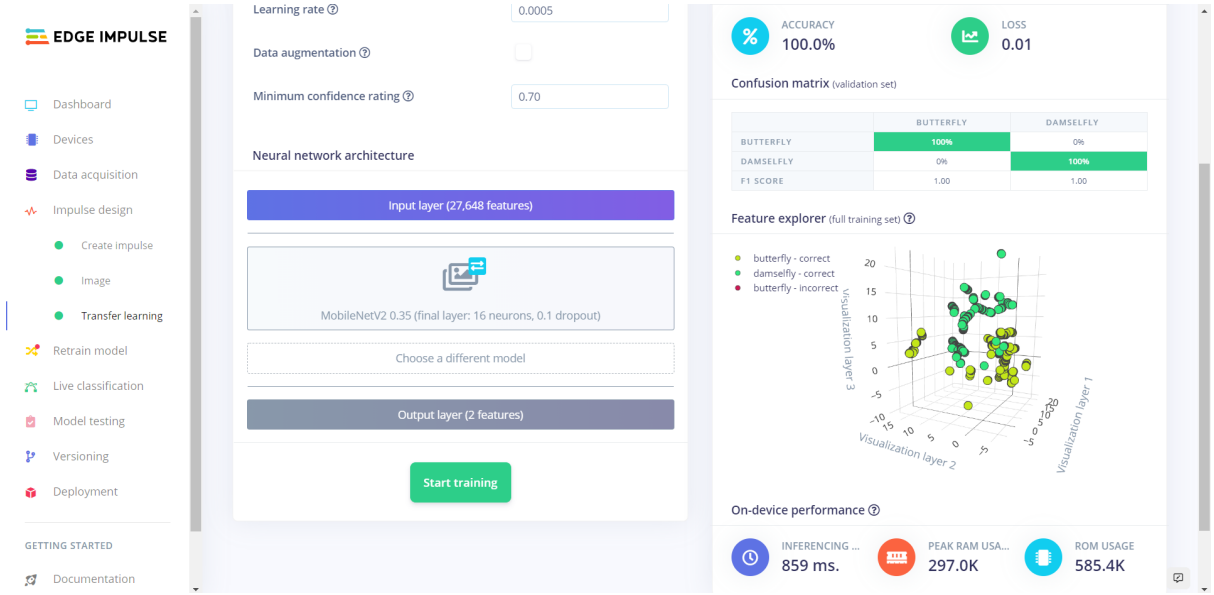


Fig 9. Yahoo….we get an accuracy of complete 100% with a loss rate of as low as 0.1 which is incredible. (Thanks to Transfer Learning!)

So we get a record on-device performance rate of 297K peak RAM usage and 585.4K peak ROM usage which is just extreme lightweight for ARM based SBC's having gigabytes of RAM and ROM availability.

Step 5 :-

Now that we have trained the model, it's time to test our model. We already have 200-300 images for testing the dataset. So the ML model will test out those and will lay down results to us.

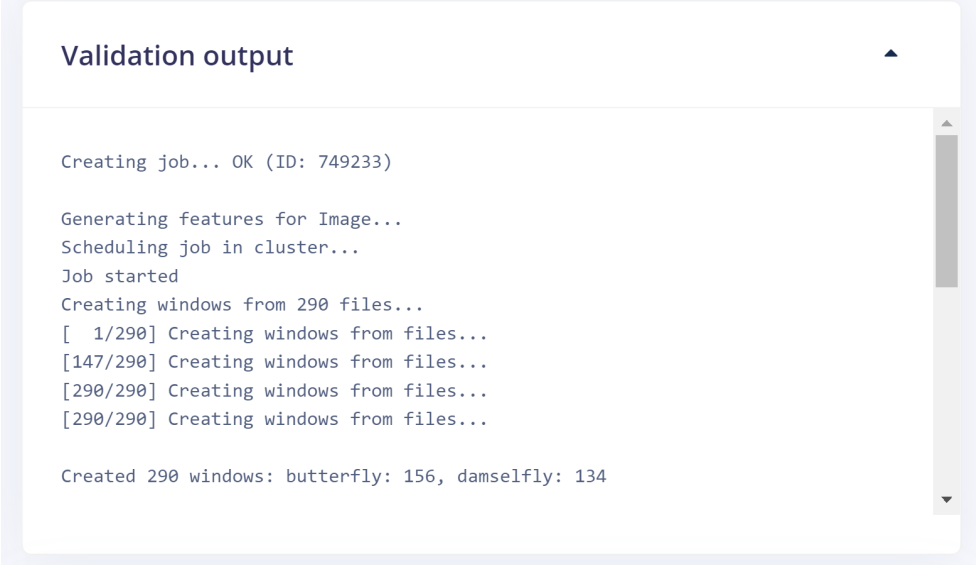


Fig 10. Validation outputs are looking good.

We will have to wait for a few minutes for the model to test all our test datasets completely.

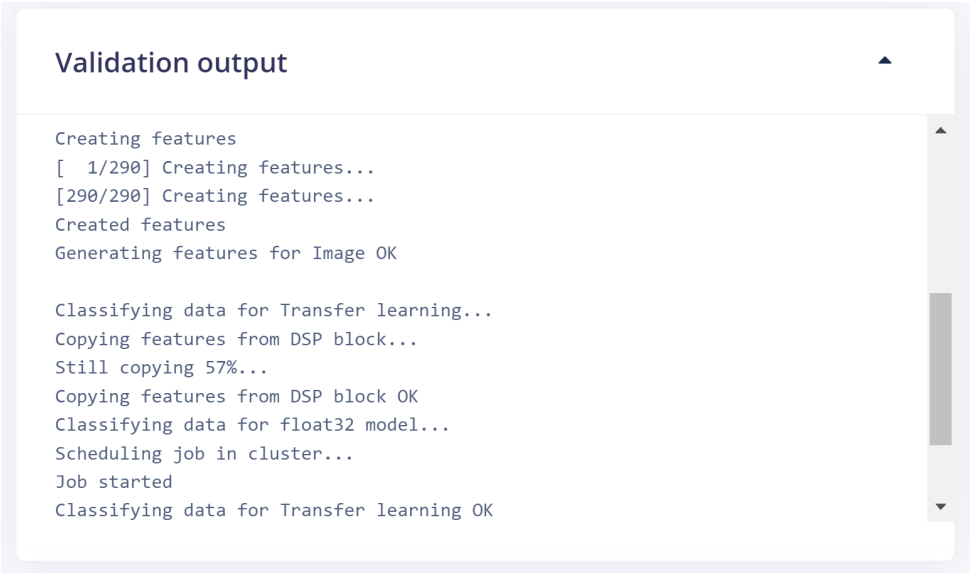


Fig 11. Model testing = done! :)

Let’s see what results do we get ?

3
2
1
.
.
.

And…….

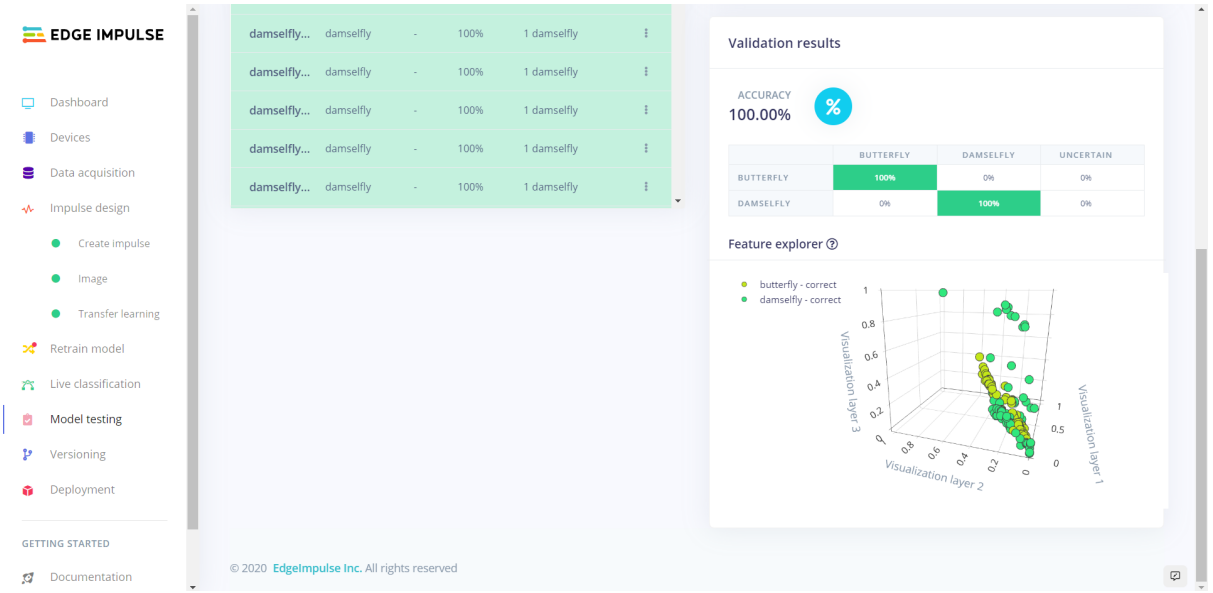


Fig 12. WOW...we got an accuracy of 100 % :D!

So now you can even see the 3-axis graph that has been generated that tells us about the generated features from the images.

So our model training is complete now let us look at how we will deploy this onto our Raspberry Pi's.

Deploying onto the hardware :-

Raspberry Pi's are small little computers with a built-in high-grade mobile processor that can perform just as a daily computer for daily-life tasks. But because it has an ARM v7/v8 based processor it can even handle resource intensive tasks like Deep Learning that we are doing.

balenaFin :-

The balenaFin has been designed with field deployment in mind. It is a carrier board for the Raspberry Compute Module 3 and 3+ Lite (CM3L/CM3+L), that can run all the software that the Raspberry Pi can run, hardened for field deployment use cases.

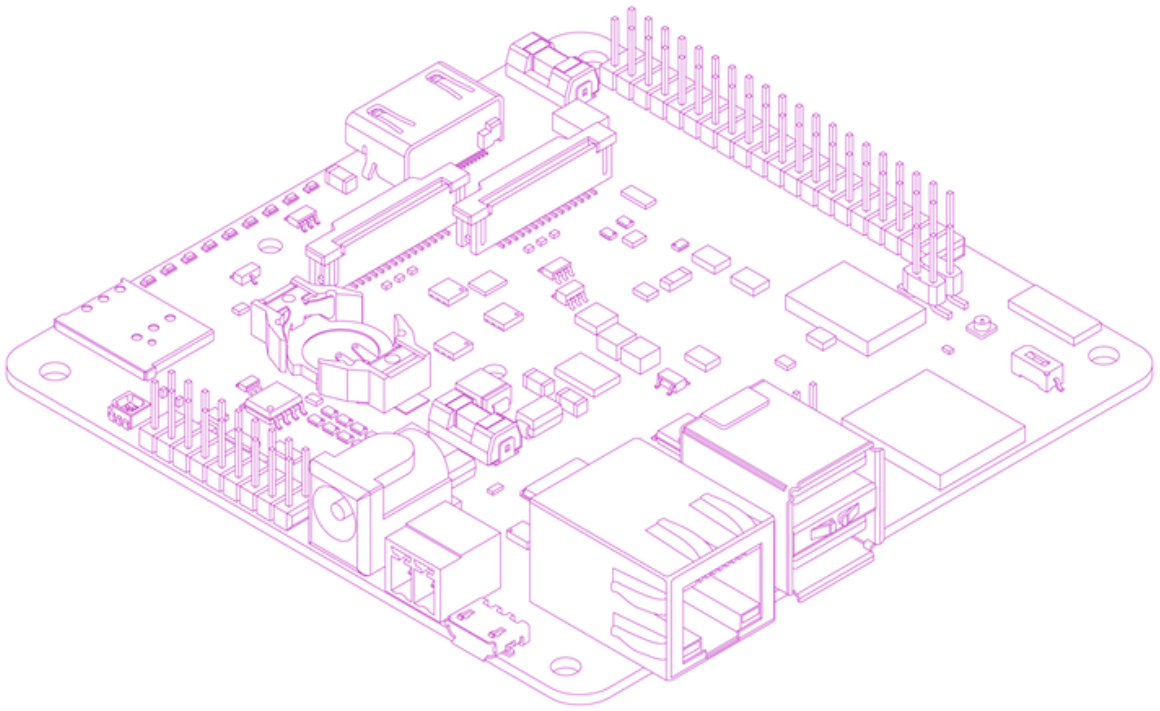


Fig 13. balenaFin schematic

Raspberry Pi Compute Module

The balenaFin supports the Raspberry Pi Compute Module 3 and 3+ lite (CM3L/CM3+L).

Storage

The storage on the balenaFin is based on an industrial grade eMMC storage with 8GB, 16GB, 32GB and 64GB options available.

Power

The balenaFin features a wide input voltage range (6V-24V), especially suitable for applications where a reliable 5V is usually not available.

Co-processor

The balenaFin includes a low-power co-processor (32-bit ARM® Cortex M4) with Bluetooth support. The co-processor can run on its own or in parallel and allows the main processor to be powered on and off programmatically. This is especially useful in applications where low power consumption or real-time processing is required.

Connectivity

The balenaFin's wireless chip supports 802.11ac/a/b/g/n WiFI and Bluetooth 4.2 (including SMART features). There is a dual-band embedded antenna included in the board and an external antenna connector for improved signal coverage.

I/O

The Mini PCI Express port on the balenaFin brings seamless connectivity to a number of different modules. Third party modules are readily available for LTE, Zigbee, LoRA and CANBus and extra storage can be achieved by leveraging the USB interface on the mini PCI Express connector (this will probably require a custom design). The balenaFin HAT header can be used to connect any Raspberry Pi HAT compatible module (PoE, RS232, ZWave, etc). A smaller 18-pin header exposes the co-processor's analog and time-sensitive I/O. V1.1 includes an extra 4-pin USB header that allows wire-free design applications such as additional storage, secondary ethernet port, multimedia readers, additional radio interfaces, etc.

Raspberry Pi Compute Module 3+ Lite :-

The CM3+ Compute Module contains the guts of a Raspberry Pi 3 Model B+ (the BCM2837 processor and 1GB RAM) as well as an optional eMMC Flash device of 8GB, 16GB or 32GB (which is the equivalent of the SD card in the Pi).

This is all integrated onto a small (67.6mm × 31mm) board that fits into a standard DDR2 SODIMM connector. The Flash memory is connected directly to the processor on the board, but the remaining processor interfaces are available to the user via the connector pins. You get the full flexibility of the BCM2837 SoC (which means that many more GPIOs and interfaces are available than with a standard Raspberry Pi), and designing the Module into a custom system should be relatively straightforward because we've put all the tricky bits onto the Module itself. This needs to be connected with the balenaFin.

Adafruit AMG8833 IR Camera Breakout Module :-

This sensor from Panasonic is an 8x8 array of IR thermal sensors. When connected to your microcontroller (or Raspberry Pi) it will return an array of 64 individual infrared temperature readings over I2C. It's like those fancy thermal cameras, but compact and simple enough for easy integration.

This part will measure temperatures ranging from 0°C to 80°C (32°F to 176°F) with an accuracy of +- 2.5°C (4.5°F). It can detect a human from a distance of up to 7 meters (23) feet. With a maximum frame rate of 10Hz, It's perfect for creating your own human detector or mini thermal camera. We have code for using this breakout on an Arduino or compatible

(the sensor communicates over I2C) or on a Raspberry Pi with Python. On the Pi, with a bit of image processing help from the SciPy python library we were able to interpolate the 8x8 grid and get some pretty nice results!

The AMG8833 is the next generation of 8x8 thermal IR sensors from Panasonic, and offers higher performance than its predecessor the AMG8831. The sensor only supports I2C, and has a configurable interrupt pin that can fire when any individual pixel goes above or below a threshold that you set.

Running the ML model :-

Step 1 :-

First open our GitHub repository (https://github.com/arijitdas123student/agri-ai-v2).
Then click on the “deploy with balena” button.

Step 2 :-

Select your device (balenaFin as of here) and download the image.

After downloading the device image. Connect the balenaFin to your host computer for flashing of the OS as per these instructions :

- Connect a USB to Micro-USB cable between your system and the balenaFin's PRG port. Please note that you can ONLY power the balenaFin from the PRG port for flashing!
- Open balenaEtcher and select the OS image you downloaded on the previous section.
- You should see your balenaFin listed as a Compute Module under the "Select drive" menu.
- Once selected, two red (5V and 3V3) and one green (ACT) LEDs should be illuminated on the balenaFin LED status panel.
- The balenaFin is now ready to be flashed with an OS.

Now we will flash it onto our balenaFin using balenaEtcher :

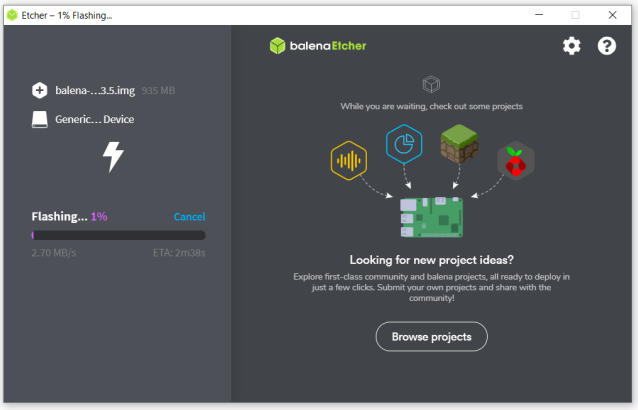


Fig 14. Wait for a few minutes for balenaEtcher to flash the image onto your SD card. In the meantime you can go and prepare a coffee for you! :)

Step 3 :-

Now connect the device to the power supply and you can see that the device will be shown as online on the balenaCloud dashboard page.

Step 4 :-

Now wait for sometime for the containers to get downloaded.

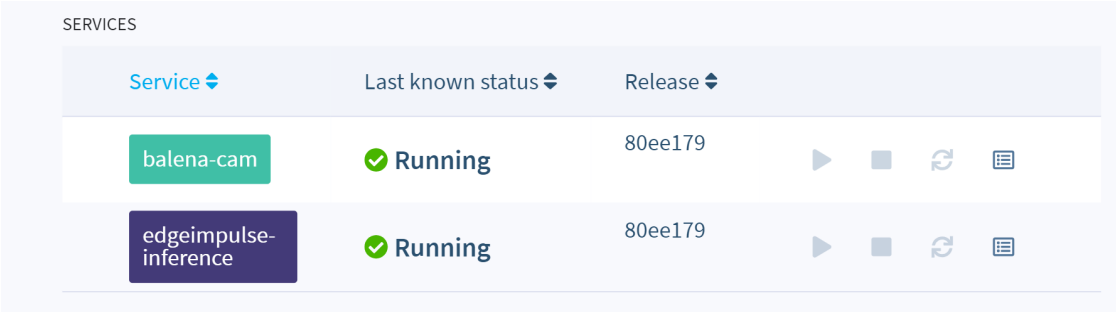


Fig 15. Containers on balenaCloud dashboard.

Step 5 :-

Now copy the Local IP Address shown on the dashboard such as the one below.

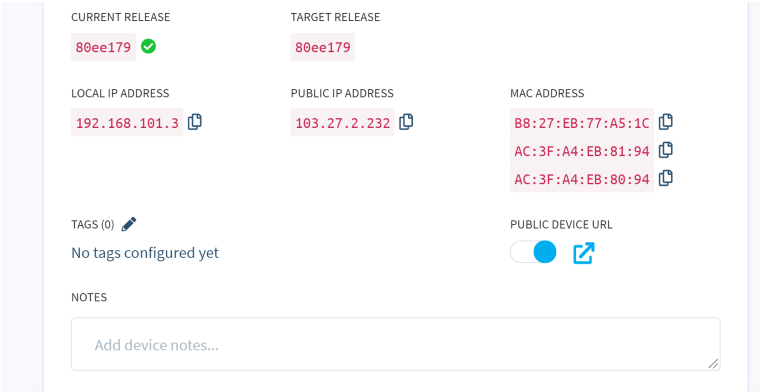


Fig 16. Networking info as per balena dashboard.

Step 6 :-

Now you will get to see an interface like this with live camera feed and real-time inference results from the balenaFin.

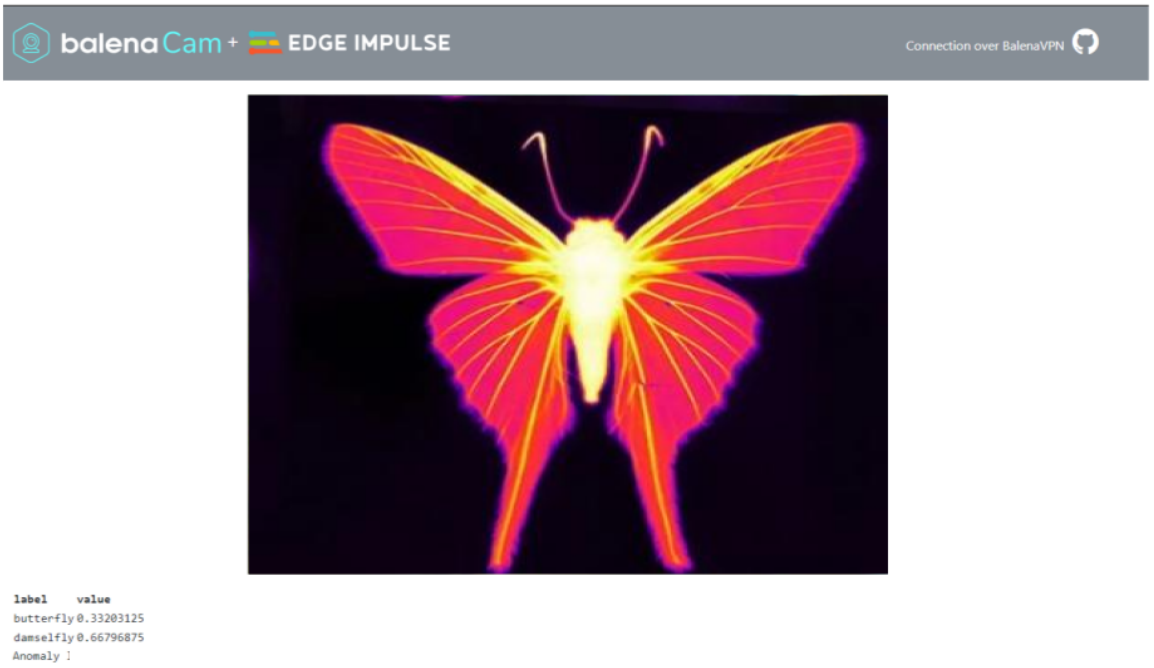


Fig 17. Local inference and real-time model outputs.

Features & Outputs :-

1. Low in cost and can be used very easily.

2. No need of a technical background to use this device

3. Will enable correct usage of chemicals in farming and thereby decrease in soil and water pollution.

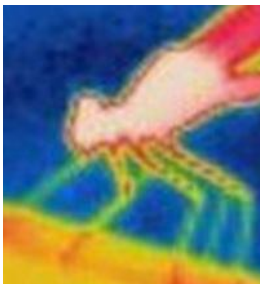
4. Prevent harmful diseases caused by spraying unwanted chemicals.

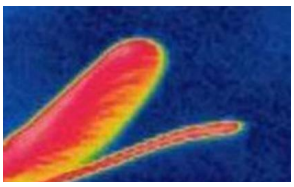
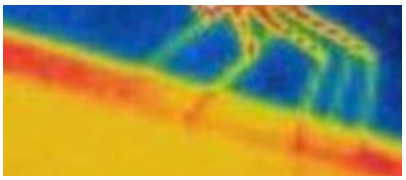
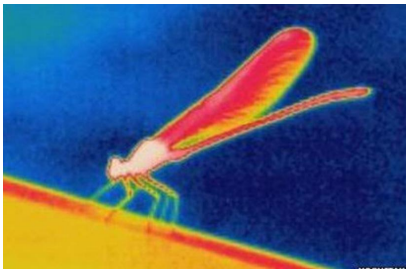
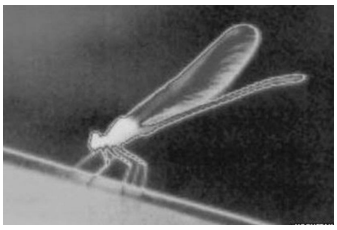
5. Less amount of required bandwidth and maintenance.

6. The model has the ability to detect around 1200 different species of butterfly and damselfly (in a thermal vision based input) with a detection time gap of 4 ms , which can also be increased to ‘n’ number of insects with ‘nx’ number of species.

Results :-

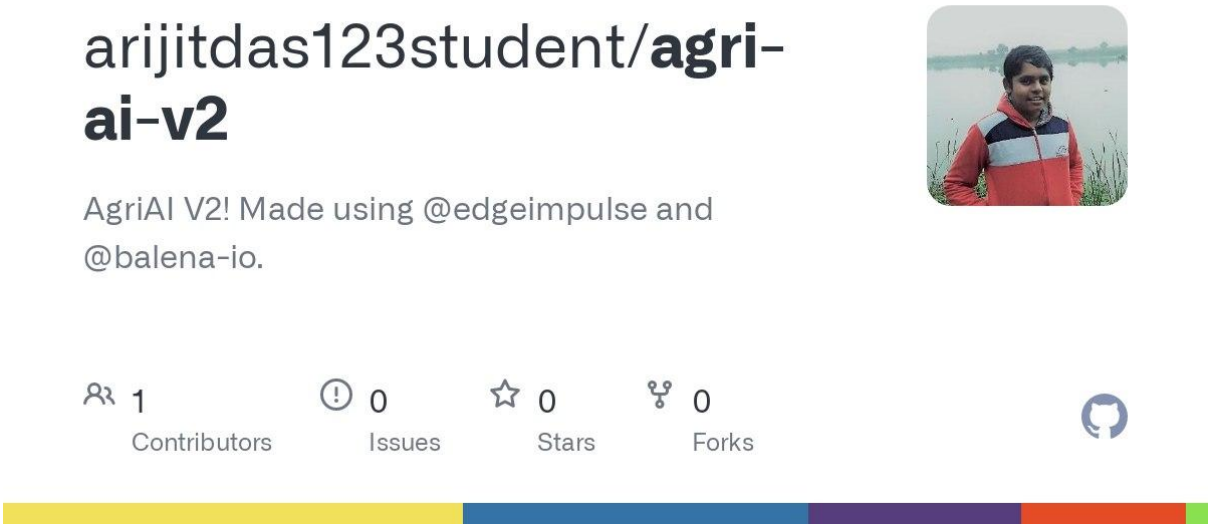
Sl No.	Part Name	Observed Values	Mean of Observations	Absolute Error	Mean of absolute error	Final way of reporting the result
1	Head (Damselfly)	100 %	97.6%	0	-2.4	97.6 ± 2.4
2	Wing (Damselfly)	100%		0		
3	Feet (Damselfly)	95%		**-5**		
4	Entire body (Damselfly)	100%		0		
5	Entire body in greyscale (Damselfly)	96%		**- 4**		
6	Head (Butterfly)	100%		0		
7	Wing (Butterfly)	98%		**- 2**		
8	Feelers (Butterfly)	91%		**- 9**		
9	Entire body (Butterfly)	100%		0		
10	Entire body in greyscale (Butterfly)	96%		**- 4**		

Sl No.	Images Used
1.	

2.	
3.	
4.	
5.	
6.	
7.	

8.	
7.	
8.	

Code :-



Check our GitHub repository in order to view the source code - https://git.io/J3Jds