October University for Modern

Sciences and Art

Faculty of Computer Science

Graduation Project

Title

Supervisor: Dr Islam sharawy

Name: aly ashraf mohamad mosselhi

ID: 210267

Abstract

This project aims to solve a huge problem which is the money spent on spacecraft maintenance equipment like depth cameras and lidars and replace them with more convenient and less costly equipment like normal monolithic camera and a program that estimates the pose of the spacecraft. Most spacecraft maintenance missions require either sending astronauts or even send another spacecraft carrying heavy equipment in space to fix or refuel the spacecraft, however this pose either high risk onto the astronaut or high money cost onto the space station, that is why space stations around the world often go to the process of autonomizing the proccess.in the light of that this project motivation is to develop a program that can detect the coordinates + the orientation of the spacecraft to help automising the refueling and the fixing of such spacecrafts and if it succeeds not only will it cut the cost of the huge equipment needed to mount the fixer spacecraft. Furthermore it will also help keep the space missions linear and no side distraction will be present to distract the space station from the said mission.

Table of Figures

Chapter 1: Introduction

1.1 Introduction

Spacecraft pose estimation have been one of the trending topics among space agencies these past years. Due to its rising importance in the field. spacecraft pose estimation is necessary for multiple tasks such as refueling and supervision of uncooperative spaceships that are out of order.

it had always been expensive to use the traditional methods such as depth cameras and using multiple cameras around the satellites. Or even in the worst cases sending actual astronauts to refuel or try to bring back the uncooperative satellites themselves.

Some terms that the reader must be familiar with:

Depth Cameras: Depth cameras are a kind of camera that produce 3D images of the target.

Monolithic Cameras: They are the normal everyday camera present in mobile phones etc.

Tango satellite: The used satellite in the dataset of the pose estimation.

Quaternions: They are a coordinate system that offers a wide variety of benefits with respect to the cartesian coordinate system. (Mukundan 2012) The main difference between Quaternions and cartesian coordinate system is Quaternions avoid a problem called gimbal lock.

CLAHE histogram: a histogram equalization technique made to give images better contrast while avoiding making noise in the image.

Currently they tend to use LIDARS and depth cameras to do this task however while they are mounted on a navigator spaceship to get the depth and the coordinates of the targeted spaceship. These cameras are very huge in size and require specific spaceships to carry them.

MB Convolutions: these are a type of convolutions that are less expensive than normal convolutions and they are mostly found in models that value performance more than accuracy usually found in Efficentnet and MobileNet model family.

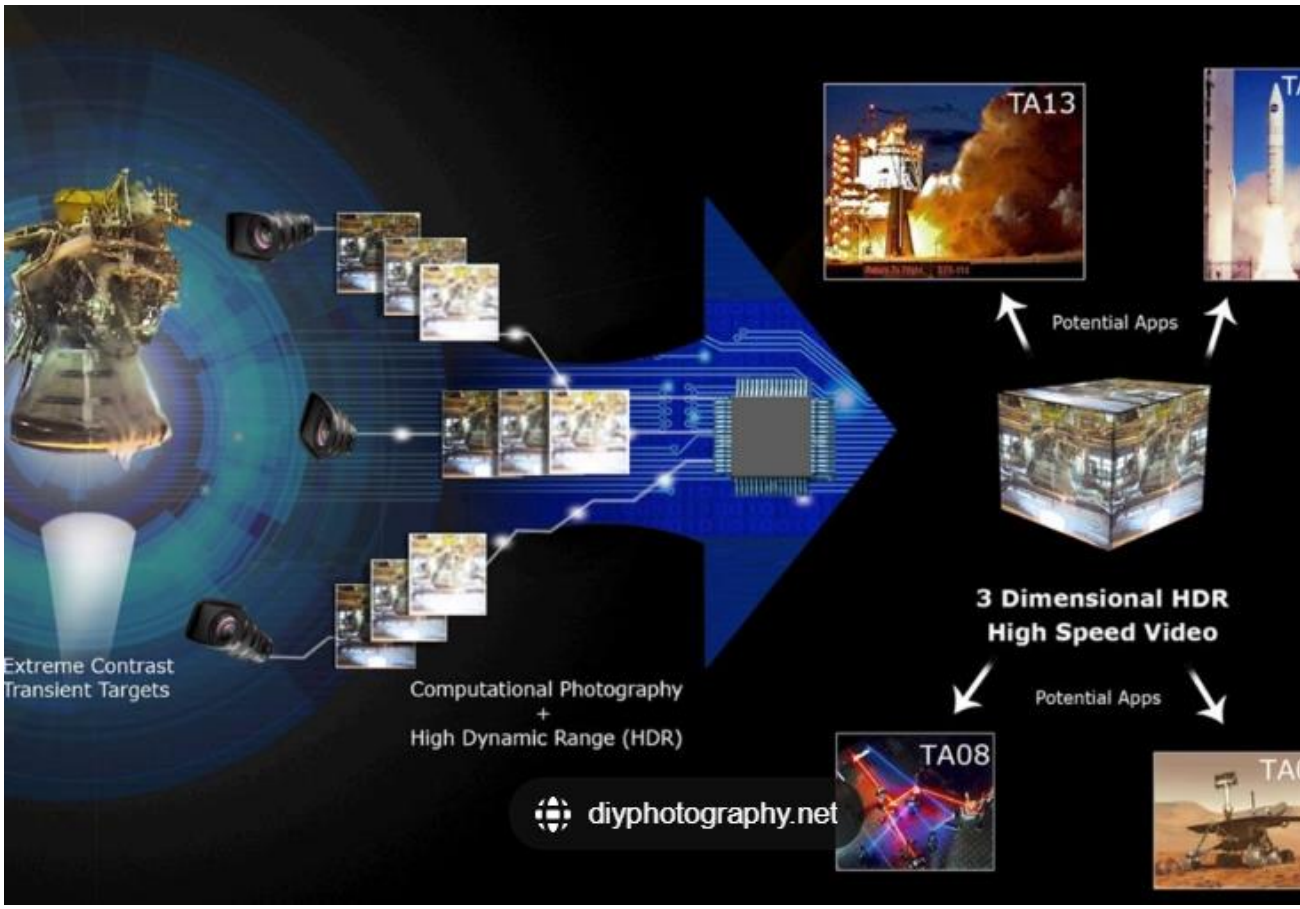


Figure 1: the depth camera

While the current situation offers high accuracy and resistance to occlusion, The equipment needed to do such task is very challenging to get.Moreover,the equipment is very big and expensive not only that but they are also power hungry and fixing them is even harder to pull off due to them being stranded in space.

1.2 Problem statement

The Challenge in this approach is the model needs to predict acceptable results within a good time frame.However,this model must also be robust and small enough for the satellite to run the model without stuttering or having trouble in the calculation because space equipment must always be real-time runned.

1.3 Objective

This project Aims to develop a light weighted program that find the spaceship and then regress the coordinates and the rotation of the spaceship

1.4 Motivation

Space Agencies have been proposing challenges to this topic for years now due to its rising importance because autonomizing the spacecraft pose estimation can in turn help reduce the cost of labor by not sending astronauts to check the satellites themselves or even worse lose the satellite in space. Furthermore, Computer vision on pose estimation can help them save the money of the heavy tedious equipment such as LIDAR sensors and Depth cameras.

The motivation of the author is to improve and facilitate the space exploration missions by making the tracking of the spacecraft pose easier and in turn this will make the space agencies focus their money more on the space missions themselves rather than focus on buying expensive equipment to maintain the spacecraft from damage.

1.5 Thesis layout

In this thesis, the first chapter will provide an introduction about the project and its aim. Then, the second chapter will provide a literature review and a background of the previous work in the same area of research. The third chapter will provide the methods used on this project. The fourth chapter will provide the system architecture and building blocks of the project. The fifth chapter will provide the results and evaluations found while working on this project. Finally, the sixth chapter will be the conclusion of this project.

Chapter 2: Background and Literature Review

2.1 Background

As mentioned before this is not the first time people worked spacecraft pose estimation projects. Therefore, there are so many different algorithms used however the approach that are used in this topic is often the same one and that is to use CNNS to compute the spaceship location on the image (Bounding box) this is important because it is needed to pin point the exact location of the spacecraft so when the key point detection algorithm tries to detect the KeyPoint it can search in a specific area rather than the whole image which will in turn save time and computational power. However, Some others also prefer using pure machine learning model (which will be this project's approach for this project) while it does not offer as great results as the hybrid method,it saves a lot of computational power IF used right.

Figure 2: Hybrid method(Bounding boxes)

Then the use of a KeyPoint detection network to compute the key points needed for the next step this step is crucial to the problem because key points are always needed to get a good representation of where the edges of the object is which is then fed to a PNP solver.

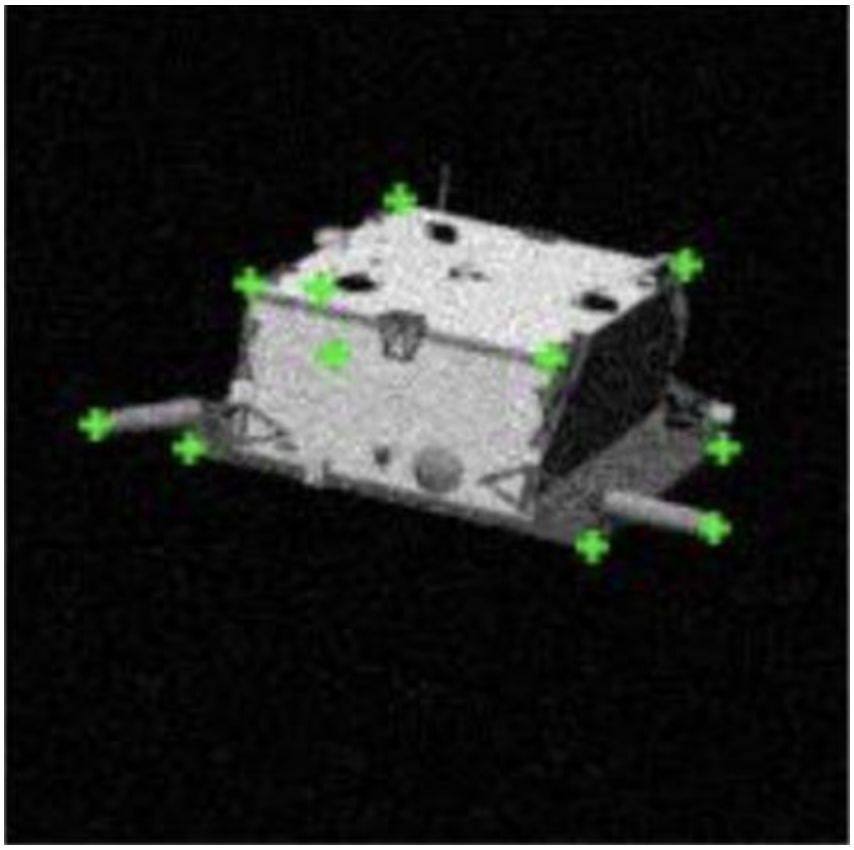


Figure 3: Hybrid method(key points)

Then lastly the pose estimation is done using a PNP Solver

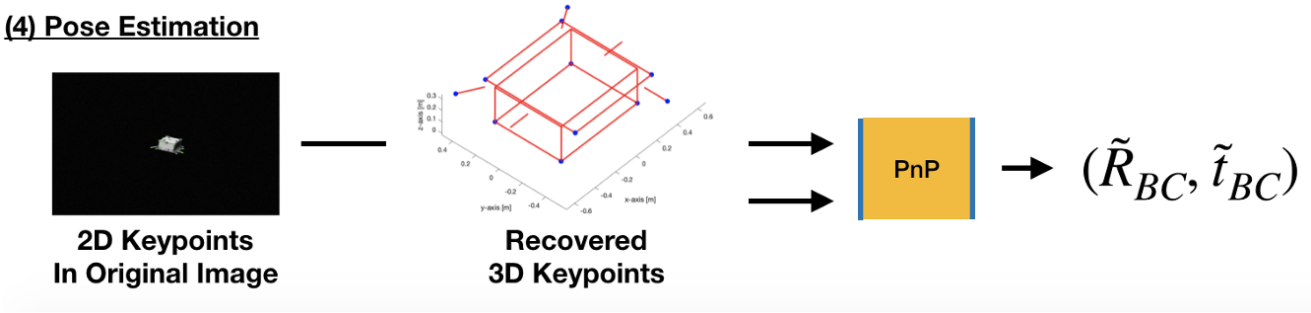


figure 4: hybrid method (PNP solver)

This is the most used approach possible to this problem due to the 3D plane of the problem and how the problem is designed. However, the algorithms used differ from one researcher to another for example:

In the ROI detection algorithms: normal CNN were used to detect the object while this approach is great in accuracy it trades of in the computational complexity of the algorithm.

MobileNet was also used in ROI detection. However, while it had low computational complexity it in turn suffered from low accuracy.
Older versions of YOLO were used too while they maintained a great balance between accuracy and complexity there was still room for improvement.

On the KeyPoint detection side
Several algorithms were used too like:
M-LSD while this algorithm is great for the KeyPoint detection it proved troublesome in the training process.

Lastly a great deal of PNP solvers were used each of them offered great results and with minimum downsides:
Ransac-PNP was utilized in this problem before while it had high complexity it proved great results in dealing with outliers.
EPNP was also tried, and it proved good complexity results in fact it proved the greatest out of all the PNP algorithms in terms of complexity however it suffered a lot in terms of accuracy.
And of course, last of all normal PNP was also tried on this problem. However, it proved mediocre results in terms of both complexity and accuracy.

For the machine learning approach mentioned earlier it is a much simpler pipeline all there is to it is to apply a machine learning model attached to neural network to regress the translation and the orientation of the satellite also while this method harnesses worst scores, it provides very fast and light models which are suitable on satellites.

2.1.1 Machine Learning

Machine learning “ML” is a kind of algorithm that falls under the umbrella of Artificial Intelligence “AI”, but the difference between them is that machine learning can adapt to various situation as long as the dataset is present

2.1.2 Neural Network and Deep Learning

A neural network is a cluster of algorithms that are made to copy the human brain to be able to strand out patterns and absorb the data in terms of machine perception, classifying, or clustering this raw input data.

2.1.3 Key point detectors

Key point detection algorithms are kind of algorithms that classify some pre chosen points that it learnt and implement it on the targeted object via different means it consists of two head regression head that find the key points itself and a segmentation head that crops the object from its frame to find better the key points.

2.1.4 PNP solvers

PNP solvers are a purely mathematical matrices algorithm that takes three inputs which are the 3d model of the object,2D key points and the 3D key points. After that it tries to solve multiple matrices problem then when it succeeds it outputs the Coordinates and the rotation of the chosen object.

2.2 Previous Work

2.2.1 end to end neural estimation of spacecraft with intermediate detection of key points

2.2.1.1 Strategy & Structure

The proposed strategy for this research is to use a neural network to regress the key points of the spacecraft itself then feed it to a pose inference network. Furthermore it also proposed that obtaining a region of interest can be done with normal convolution network.

This project theorize that RANSAC-PNP is the perfect PNP type to solve this problem due to its resistance to outliers.moreover,this research innovates a new way to work on the problem due to previous researches using convolution neural networks to directly regress the pose of the spacecraft however this proves to be an unreliable solution to this problem due to the noise of the images and the pose itself

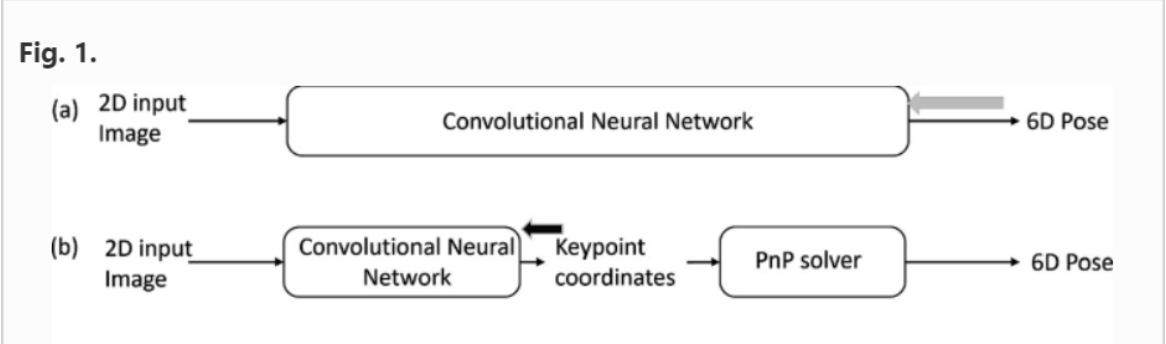


figure 5: architecture for research one

2.2.1.2 Data

The model is applied on the Speed+ dataset it consists of three types of images of the satellite tango most of them are images that were made using OpenGL due to the scarcity of satellite images. The size of the images in total is 46k images that have no labels nor ROI. However, it comes with the Rotation and coordinates labeled.Moreover,it has the camera parameters along with it too. The three types of images are:

Synthetic: these are images that have no noise nor anything special about them.

LightBox: these consists of very bright images that match the occlusion of space near the sun.

DarkLamp:these consists of very darkened images to match the occlusion of space near the earth atmosphere.

2.2.1.3 Method Evaluation

Using CNN to find the ROI is good in terms of accuracy, however it proved bad in terms of this specific problem due to the computational power of looking through the whole image and trying to find the object. Furthermore, using the normal PNP is not the optimal solution to this problem due to its mediocre accuracy and high complexity.

2.2.1.4 Results Evaluation

The conclusion of this study is a different variation of PNP solvers could be used to reach either better results in terms of its accuracy or get better complexity time.

The results achieved by this research was

S(ipos) is for the position

S(ori) is orientation

S(total) is the total of both

Team name	S^j_{pos}	S^j_{ori}	S^j_{total}
VPU (Ours)	0.0166	**0.0646**	**0.0812**

figure 6:results for research one

2.2.2 toward robust learning-based pose estimation of noncooperative spacecraft

2.2.2.1 Strategy & Structure

Unlike the previous research this research suggests using convolution network to instead regress the ROI of the object after that it follows the same pipeline of getting the 2D key points from the object using a dedicated KeyPoint detection network after the 2D key points is extracted from said image it is then fed into a PNP solver along with the 3D model of the object. This research also states that obtaining a 3D bounding box can also increase the accuracy of the pose estimation.

This project is built on pre-defined structure which is the 2D key point structure suggested in the earlier research.However,it adds the 3D bounding box and the 3D model to the equation.they believed that it will offer greater results when the 3D parameters is added to the pnp solver.they also one upped the previous research by applying YOLOv3 and mobilenetv2 model to the project which has very high speed compared to normal CNN yet it offers good results too.

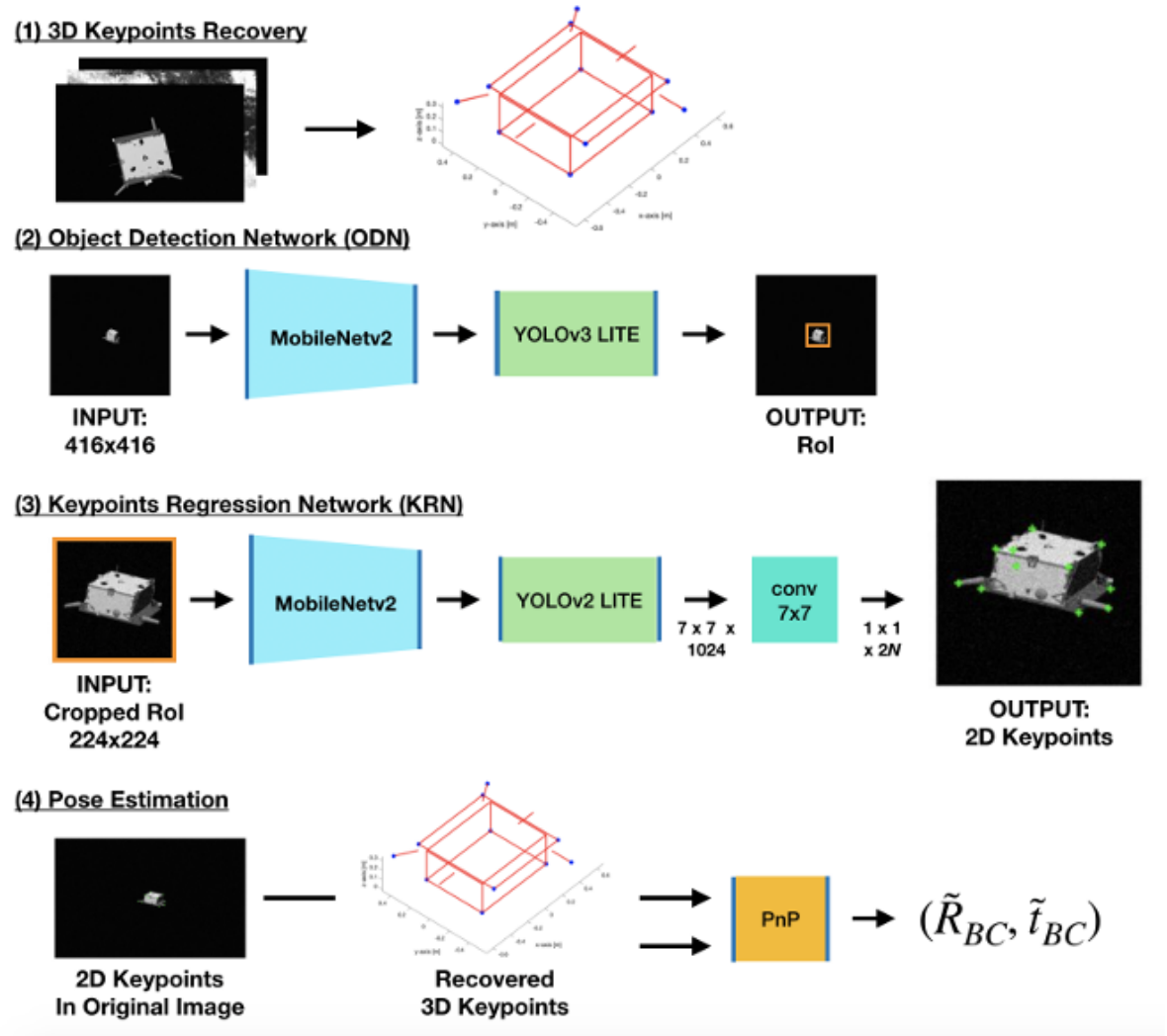


figure 7:network for research two

2.2.2.2 Data

This research uses an upgraded dataset to the previous research which is the speed+ dataset. This dataset is the same, however it offers even more synthetic images to train onto not only that, but it also offers the camera parameters for a better and more accurate representation of the pose.

2.2.2.3 Method Evaluation

Using the yolov2 and yolov3 is a smart approach due to their very low computational power not only that but applying mobilenetv2 also adds more accuracy to the project however, while these models have great balance between accuracy and complexity they are very old models which have a lot less optimization compared to newer MobileNet and yolo models moreover, while the 3d key points adds a lot of accuracy it is still hard

to obtain them and solving in 3D key points might prove troublesome in terms of complexity

2.2.2.4 Results Evaluation

The conclusion of this study is: adding the 3d regressed KeyPoint + two very light weighted model can prove a very good counterbalance between accuracy and complexity moreover, they proved that CNN is not the only go to solution when obtaining the KeyPoint or the ROI.

They also scored these results which are very good in terms of accuracy

Mean $\boldsymbol{E}_{\mathrm{T}}$ [m]	[0.010, 0.011, 0.210]
Median $\boldsymbol{E}_{\mathrm{T}}$ [m]	[0.007, 0.007, 0.124]
Mean E_{R} [deg]	3.097
Median E_{R} [deg]	2.568

figure 8:results for research tw0

ET:The transformation error results

ER:The rotation error results

2.2.3 toward robust learning-based pose estimation of noncooperative spacecraft

2.2.3.1 Strategy & Structure

In contrary to the previous two researchers this one focuses on a purely machine learning aproach.what is different here is these researchers dropped the KeyPoint regression and the pnp solver in Favour of a more robust and lighter model that purely runs on a mobilenetv2 backbone and two feedforward network for the regression of the poses. They also stated that a pure machine learning approach can speed up the training process and eat up less memory in the process. This is proved as they compared their model with a previously worked on model which is the original URSOnet model in which this model was based on. They said, “Our lightest model has 178 times fewer parameters while degrading accuracy by no more than four times compared to URSONet”.and this further proves their lightweight model.

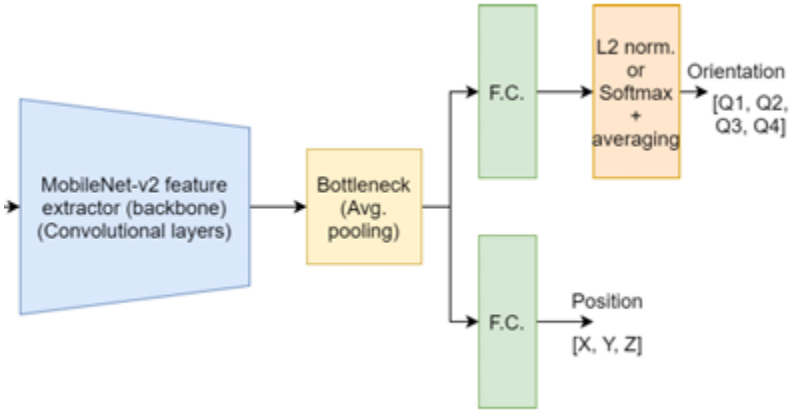


figure 9: network for research three

2.2.2.2 Data

These researchers use the standard speed dataset that consists of 12k images that contain training synthetic images and real images for testing. This dataset was created using OpenGL and augmentation to create even more images from the OpenGL made images, and the real images were captured of the TANGO satellite.

2.2.2.3 Method Evaluation

Using mobilenetv2 was a smart idea for developing a robust small network due to its small size and acceptable accuracies. Moreover, the usage of just a avg pooling bottle neck and two feed forward network further assists the robustness of the model. Not to mention that they treat the orientation as a SoftMax classification instead of the normal regression that other researchers tend to do.However,the model struggles with the generalization of the images which is a huge problem if the model is applied to real case scenarios which could lead to problematic encounters.

2.2.2.4 Results Evaluation

In conclusion this study suggests a purely machine learning model that regresses the pose of the spacecraft using minimal complexity.in which they succeeded but they suffered from the problem of generalization because the model over fits on the data of the number of classes in the orientation department increases.

Participants	# params (M)	E_{synth}	E_{real}	G_{factor}
Black *et al.* [9]	6.9	0.0409	0.2918	7.13
Sharma *et al.* [6]	11.2	0.0626	0.3951	6.31
Proença *et al.* [1]	500	0.0571	0.1555	2.72
ours (regression)	2.2	0.6160	0.7997	1.30
ours (8 bins)	2.8	0.2520	0.7868	3.12
ours (12 bins)	4.4	0.2104	1.2231	5.81
ours (16 bins)	7.4	0.1947	1.2074	6.20

figure 10:results for research three

Chapter 3: Material and Methods

3.1 Materials

− In this section we will explain the used materials in this project which varied between data & tools & environments.

3.1.1 Data

This dataset presented by the European space agency and it consists of 12k images of the Tango satellite, it can be found via this link [2110.03101] SPEED: Next-Generation Dataset for Spacecraft Pose Estimation across Domain Gap (arxiv.org).

This data is made using OPENGL which is made to create a patch of synthetic images of the satellite. Some Gaussian noise pre processing was made to try to replicate the space environment it also comes in grey scale to facilitate working with the data.

It consists of two types of data:

Synthetic: Generated images of the satellite using OpenGL and adversary networks

Real: Which is the other type of images that are not generated (the original real images)

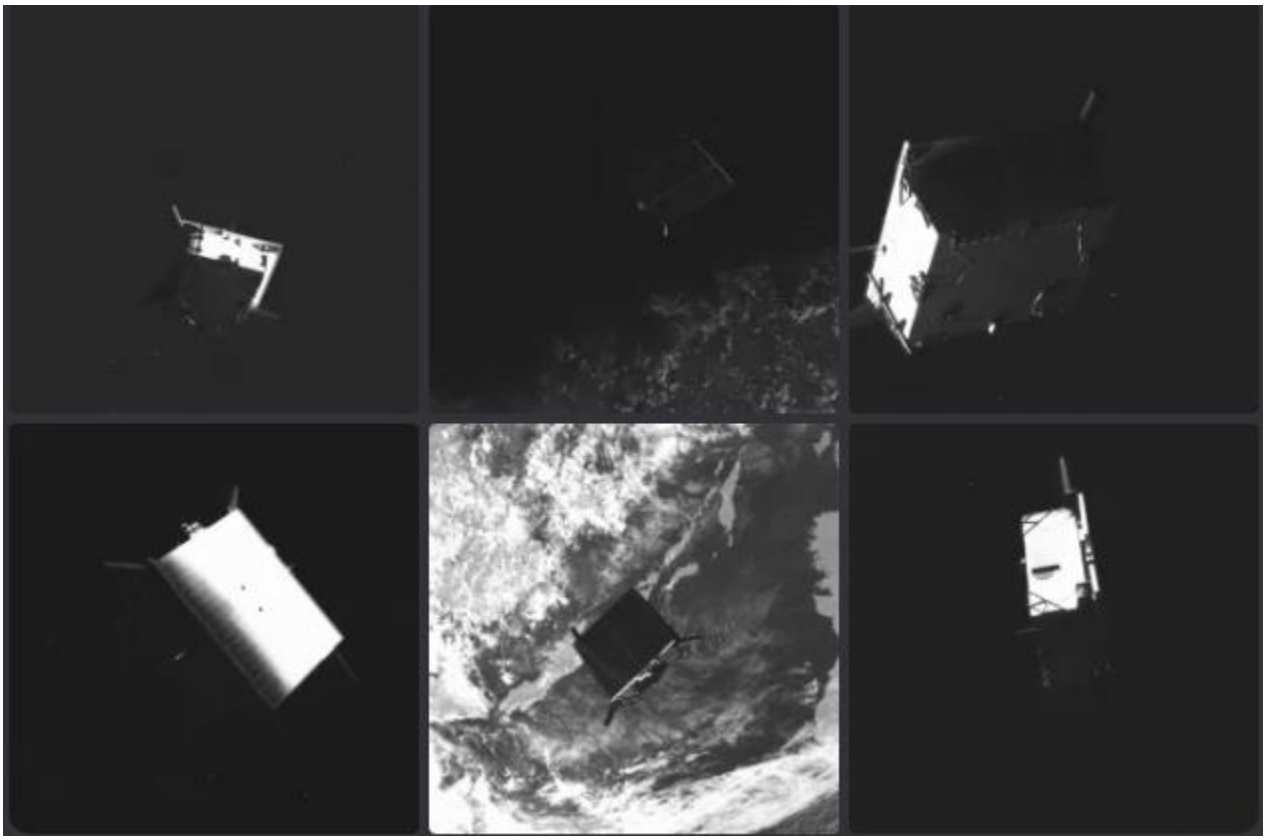


Figure 11: Sample of the SPEED dataset

3.1.2 Tools

− Anaconda: is a platform for efficient developing and applying AI and machine learning models.

− Python 3: An open-source programming language that enables developers to work and integrate their systems quickly and effectively.

-OpenCV: open cv is a library that can manipulate 2D images and it contains most of the algorithms needed to create the pose estimation program.

-Visual studio code: a complier developed by Microsoft to facilitate programming on the programmers.

-Efficentnet: a machine learning model that was developed to be a fast robust model that also achieves relatively good accuracy compared to mobilenetv2.

-torch vision: machine learning helper that was made to operate on image related data instead of numerical data.

-matplotlib: package that is used for visualization of the results of the model

3.1.3 Environment

Local GPU was used for this experiment for various reasons being GPU is a lot faster than CPU in training and testing the models and time is always of essence. However, this begs the question why not use cloud it is a lot faster and to answer that this sole experiment is made to develop a robust model that can regress without being a heavy duty model so to add authenticity the findings of this project local machine was used instead of the unrealistic state of cloud because no satellite can run cloud.

3.2 Methods

3.2.1 System architecture Overview

First the image is resized to 255x255, passed onto a CLAHE filter to get a better contrast of the image, then will pass onto Efficentnet back bone that is connected to two feed forward networks one regresses the translation of the satellite while the other regresses the orientation of the satellite.

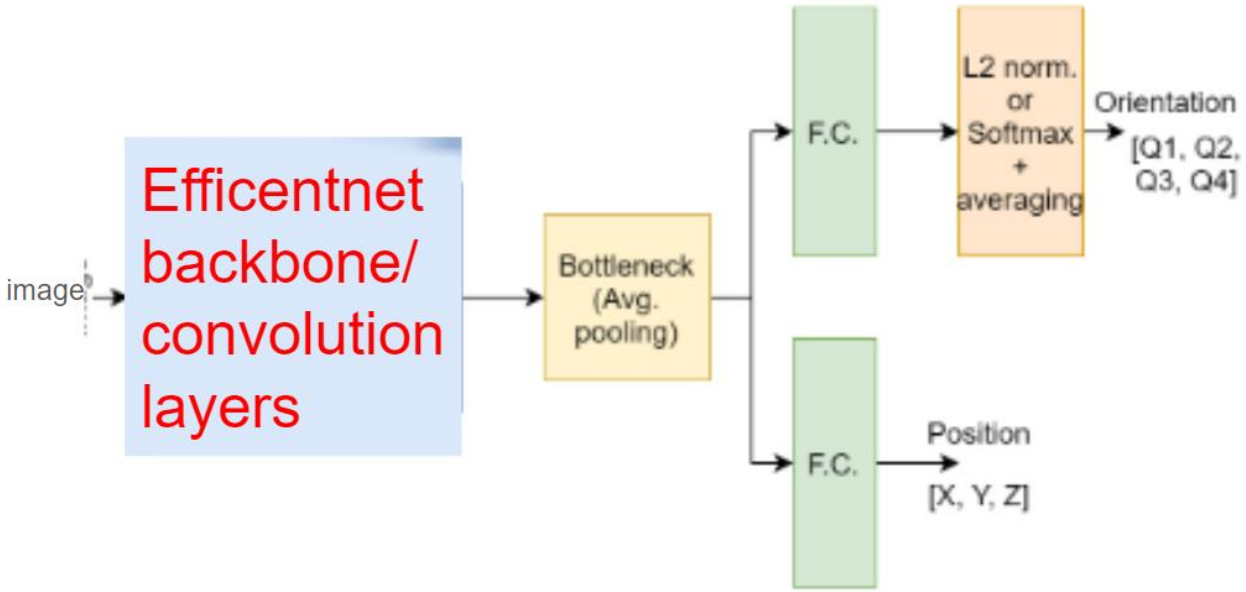


Figure 12: POSE estimation network

Chapter 4: System Implementation

4.1 System Development:

Alot of testing went into the system with multiple experiments as this project is based upon previously mentioned project which is mobileURSOnet by Posso et al. Multiple assumptions were made while put into practice 15 epochs were used for the evaluation of the edits.

-first of all, we noticed that the image size was relatively big, and we thought that it could be resized before entering the pipeline of the model to improve the speed of the training and inference.

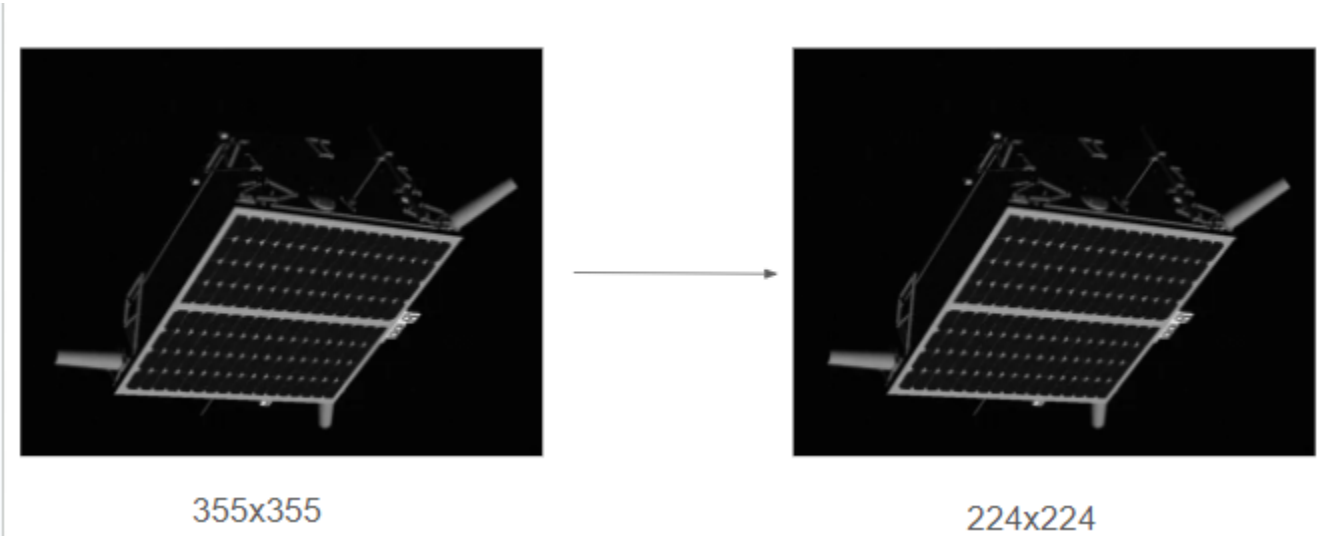


figure 13: Resizing of the image

While resizing proved effective in the speed of the model we faced another problem which was the score of the model degrading due to smaller images on the mobilenetv2 backbone.However,when efficentnet was placed instead of the mobilenetv2 the score loss was not as bad as it was with mobilenetv2.

There was more to improve so various histogram functions were tested upon the images to enhance their contrast and quality. After testing these multiple functions,CLAHE was the best result histogram function there was as seen in figure 12 while the image might seem not that diffrent.it is better due to details of the image being more visible in the equalized environment.

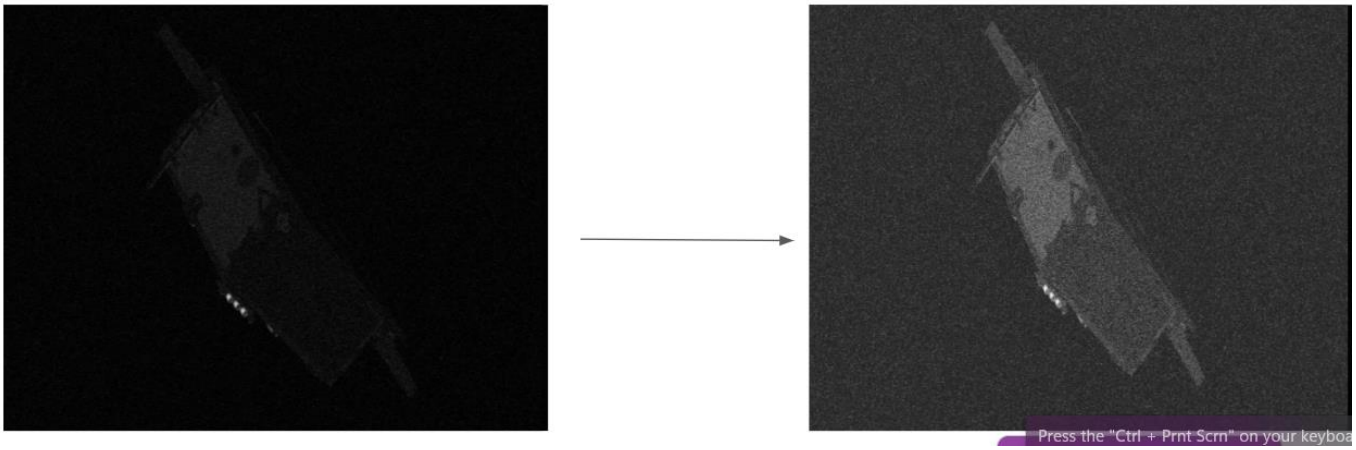


figure 14: CLAHE histogram

Furthermore, addition of Poisson noise, Gaussian noise was made to try to simulate the noise in space so a good results on the real images could be achieved which was one of the struggles of the original project due to the model over fitting when more orientation bins where added however only Poisson proved effective in that matter while gaussian made the results even worse.

And lastly, an Efficentnet backbone was put in charge instead of the mobilenetv2 of the original project, this update proved tremendous improvement in the results of the model which will be discussed in the results and evaluation section.

During the deployment four platforms were tried to run the project [Kaggle notebook, google colab,pycharm,and visual studio code].

4.2 System Structure

In this section we will talk about the system that was deployed and what the system overview of this project. In the first subsection we will talk about the final system and the stages that the system passes through to get our product, in the second subsection we will illustrate the system building blocks using tensor board and mention how they are related to each other's.

4.2.1 system overview

This model consists of three main stages, the first stage is: data preprocessing, this stage is responsible for the preprocessing of the data before it enters the model, and it consists of multiple steps to reach the needed dataset preparation.

In our 1st main stage, the data is loading the dataset into the memory to make the required preprocessing steps.

After that image resizing will be implemented then the CLAHE histogram is applied to the images of the dataset to get better contrast

On the image the third step is to apply some image rotation to avoid over fitting on the data. Furthermore, rotation will add some noise due to little false labels on the data to improve the Generalization of the model. And finally, Poisson noise is added to the image to complete The data augmentation proccess.When all these steps are done the images are then copied to a Folder so that the model becomes ready to use the dataset for the next stages.

In our 2nd main stage, the model starts the training process by loading our predefined weights From the efficentnet family, then the model is created using these weights the model. After that the model is passed onto 7 sequential blocks each containing number of MB-CNN's (mobile inverted convolutions) ,the first block contains one 3x3 convolution then on block two, two convolutions are present then on block three two 5x5 convolutions are present then on block four three 3x3 convolutions are present, on block five three 5x5 convolutions are found, on block six, four 5x5 convolutions exist, however on the last block of the training process which is sequential block number seven only one 3x3 convolutions exists. Then it is passed onto the feed forward network for the predictions on our 3rd main stage.

There exist two feed forward networks in the process, one to regress the orientation of the satellite and one to regress the translation of the satellite within the image to demonstrate further here is a figure so the reader may visualize the System.

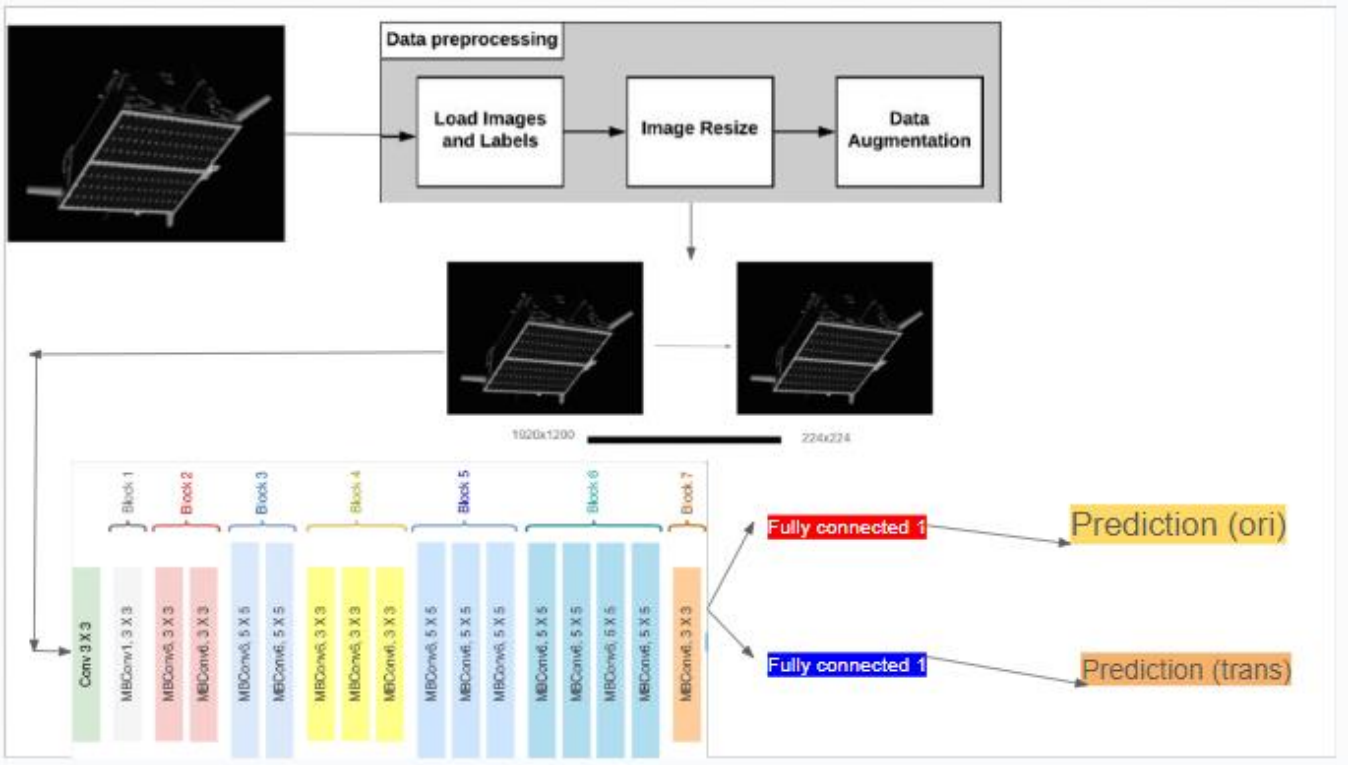


figure 15: system over view

4.2.2 tensor board

The following figure describes the architecture of the model in a more detailed matter and shows the sequential blocks and the convolutions of this project in a more detailed manner with illustration.

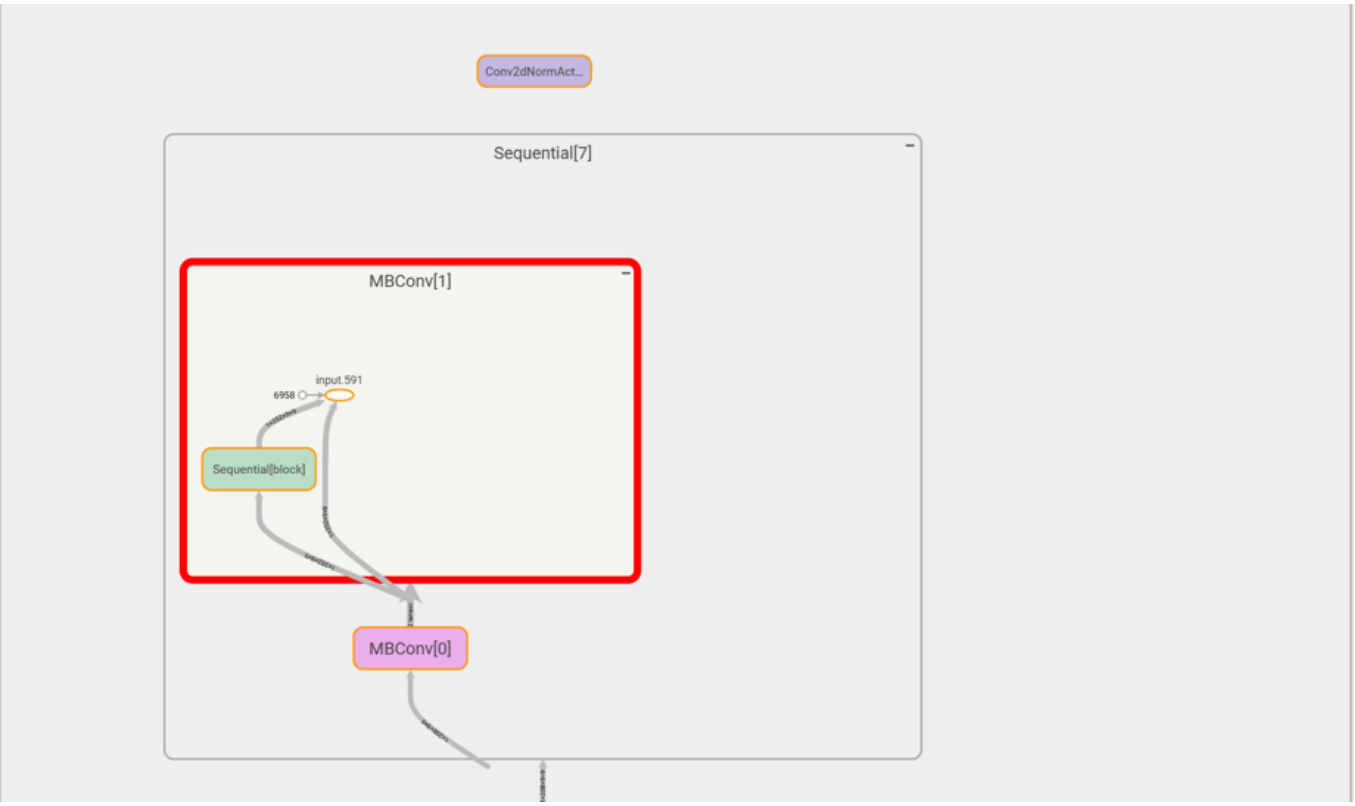


figure 16: sequential 7

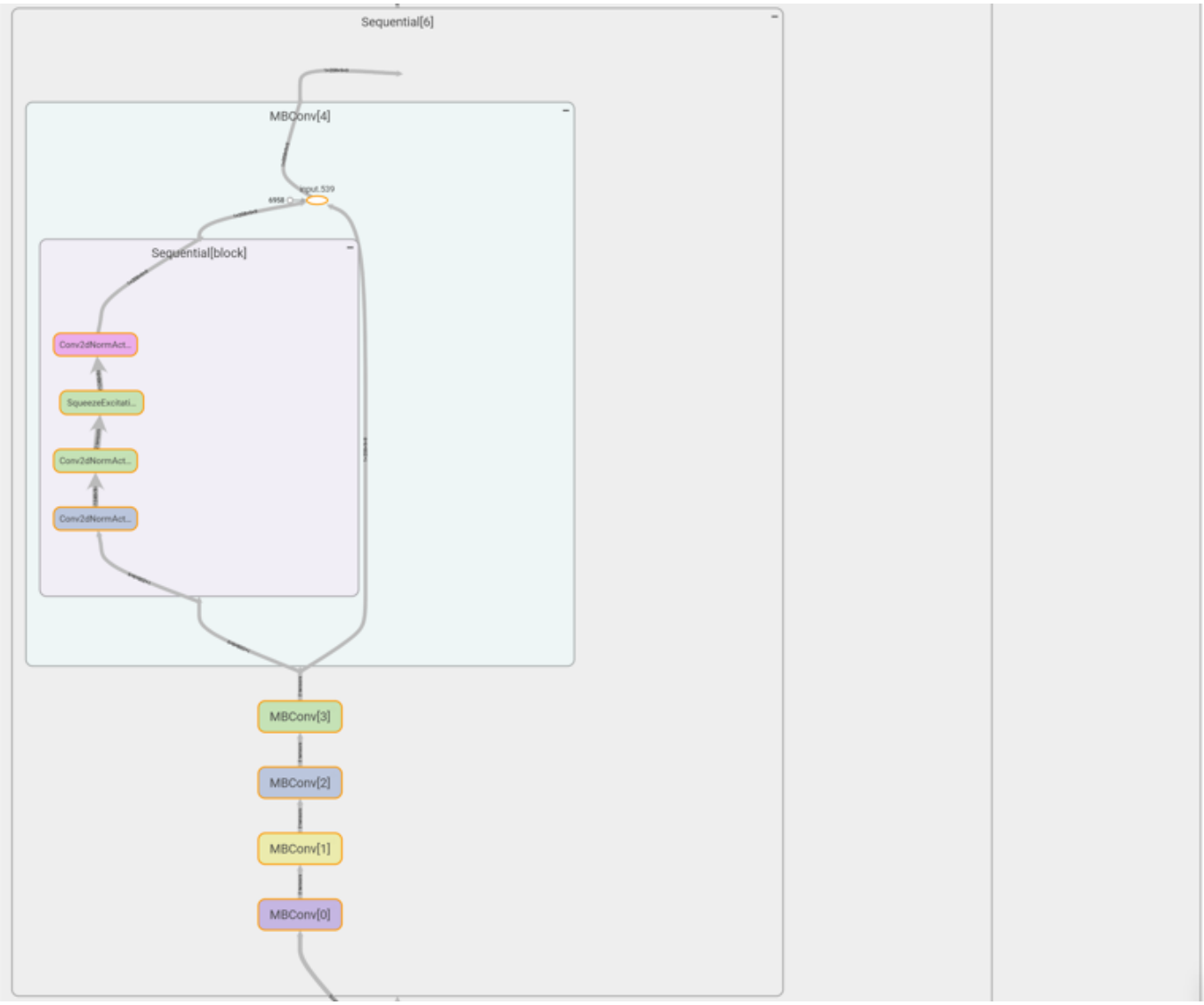
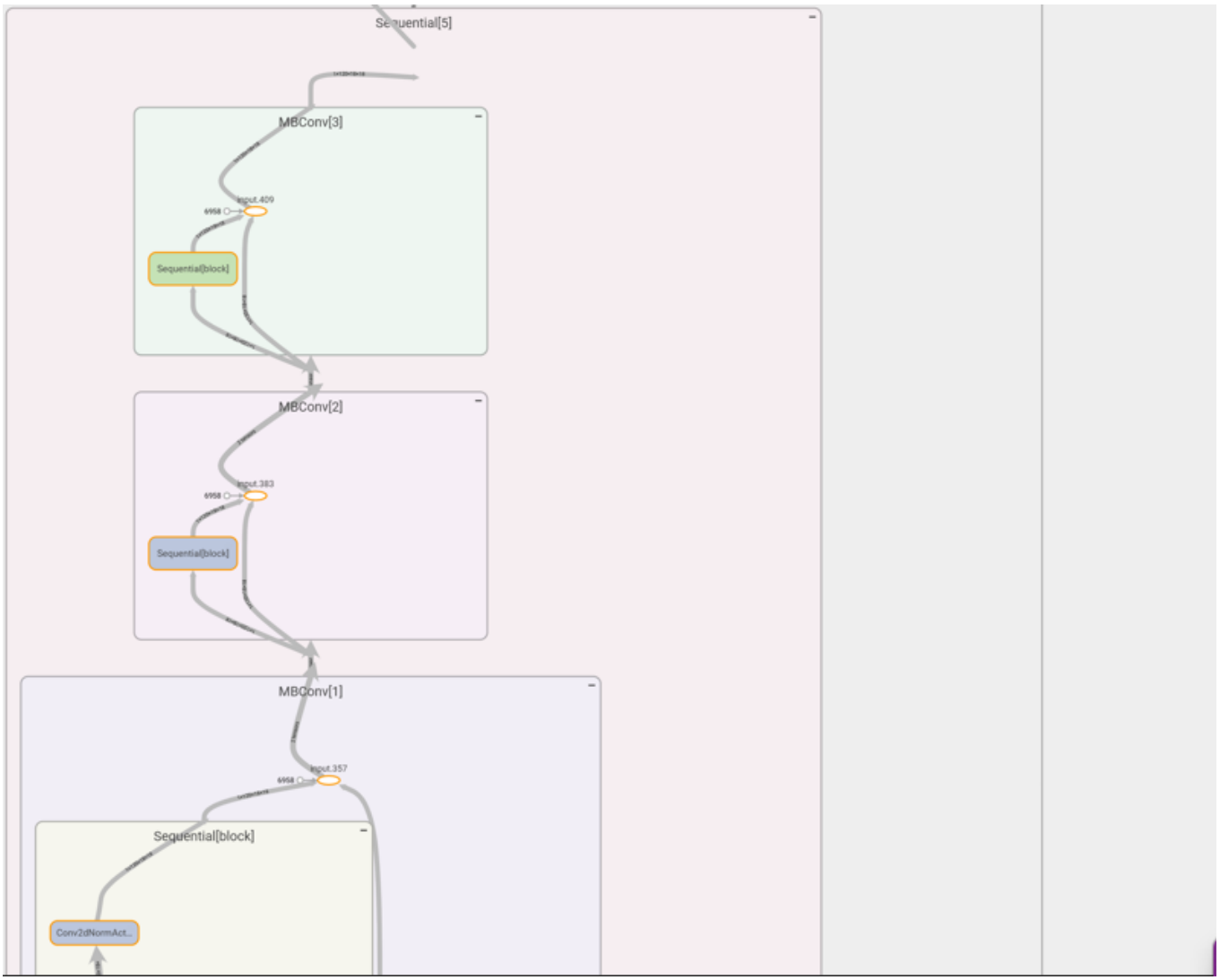


figure 17: sequential 6


Sequential[5]
MBConv[3]
input.409
6958
Sequential[block]
MBConv[2]
input.383
6958
Sequential[block]
MBConv[1]
input.357
6958
Sequential[block]
Conv2dNormAct...

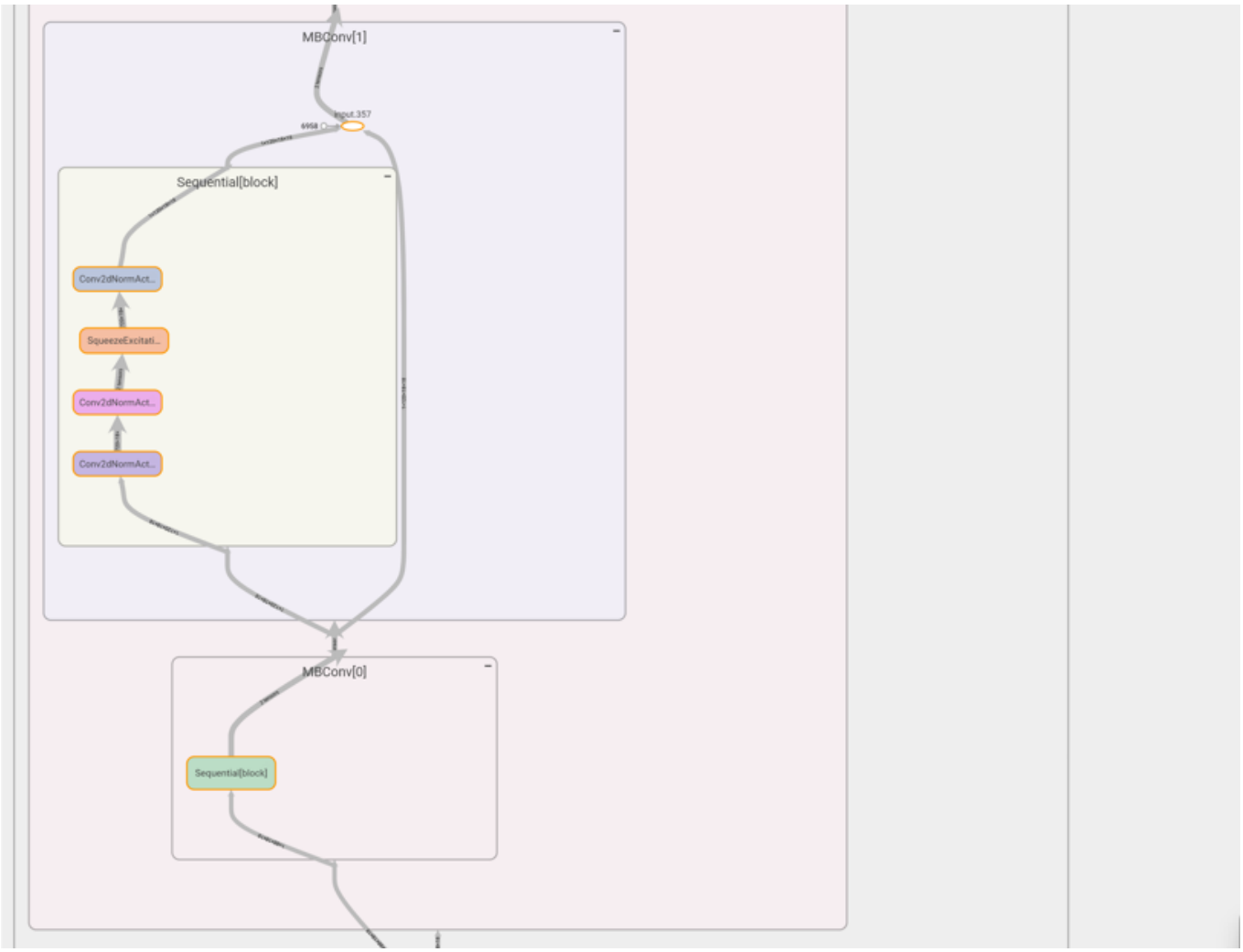


figure 18: sequential 5 (more details about sequential blocks)

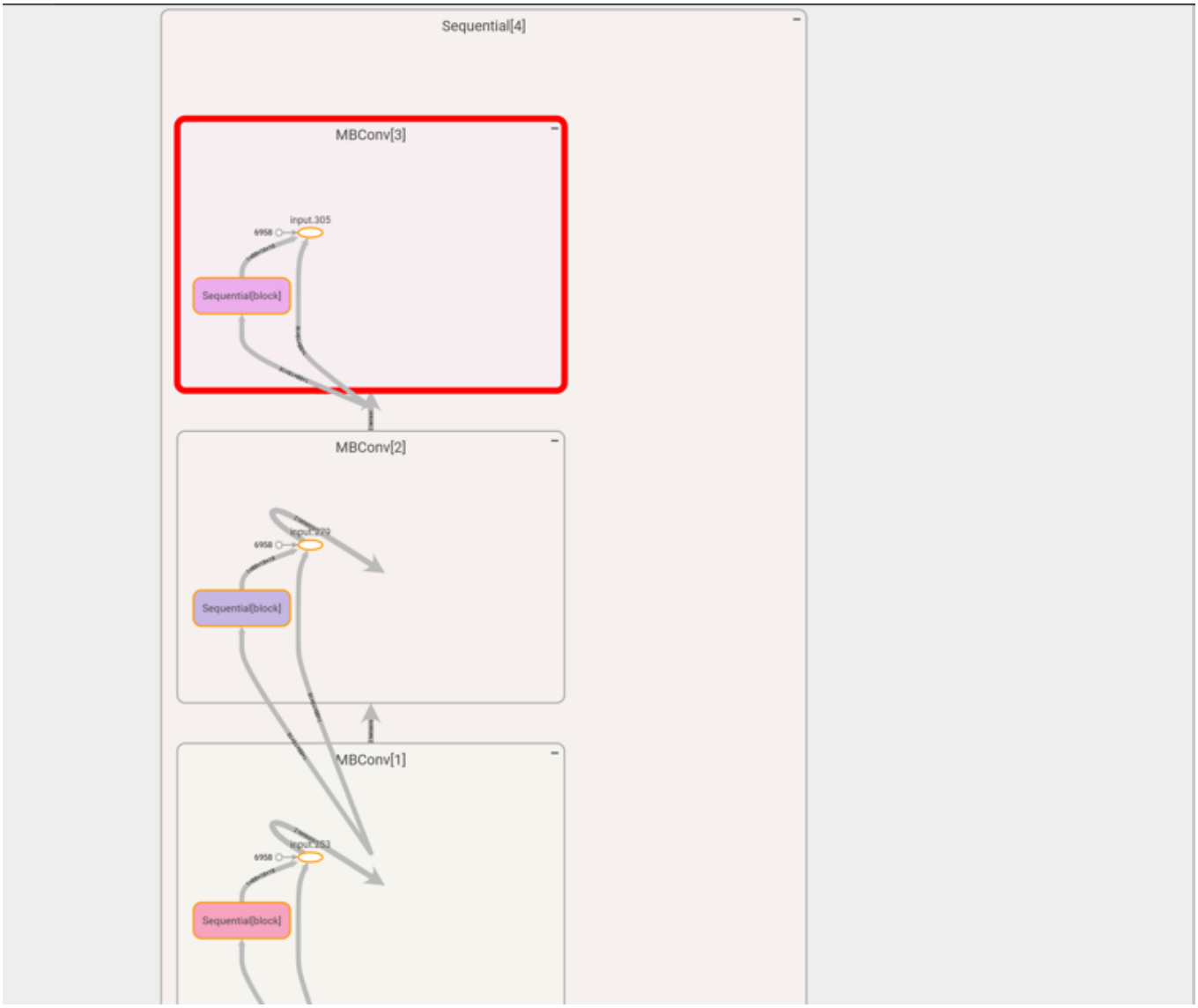


figure 19: sequential 4

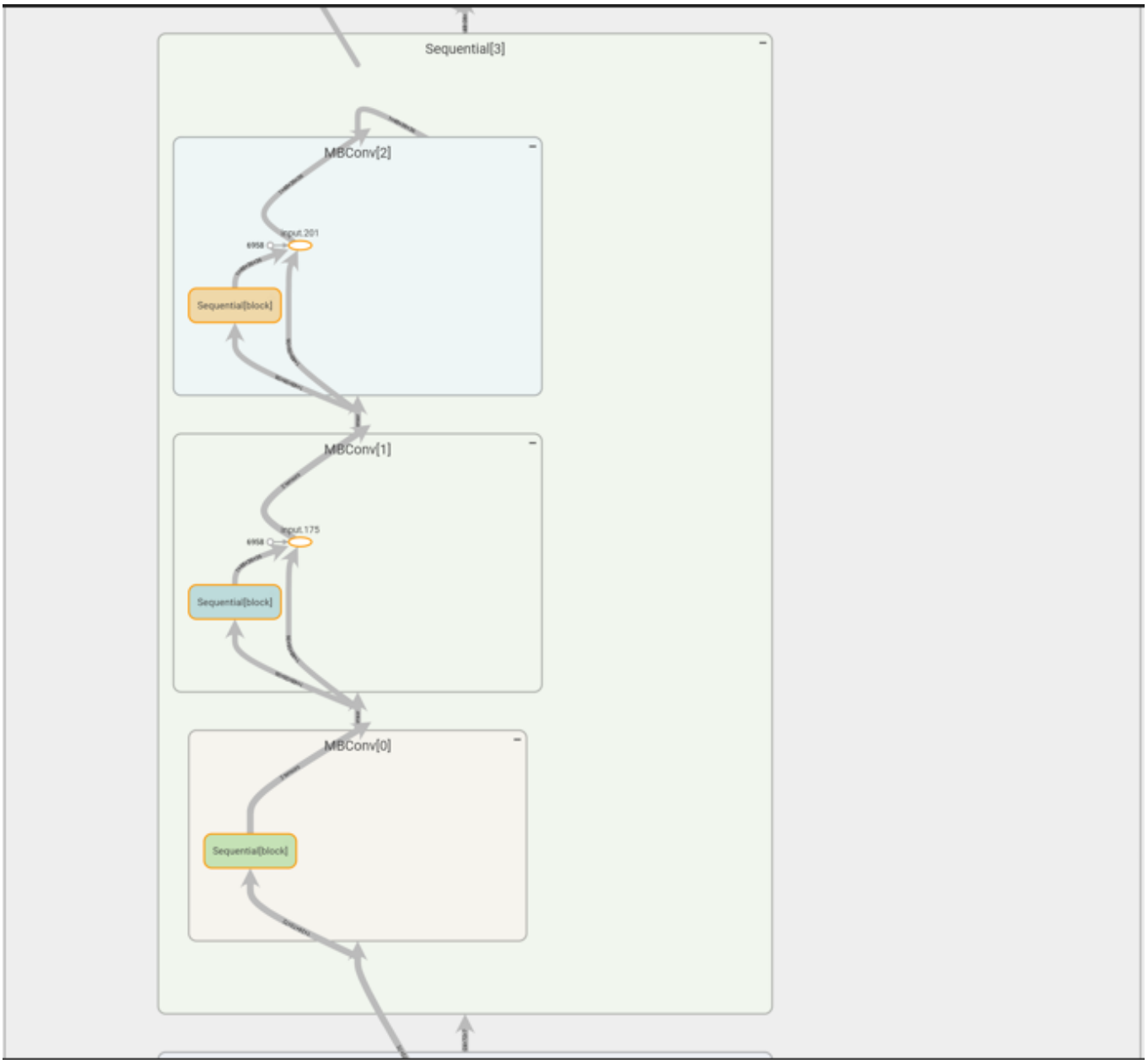


figure 20: sequential 3

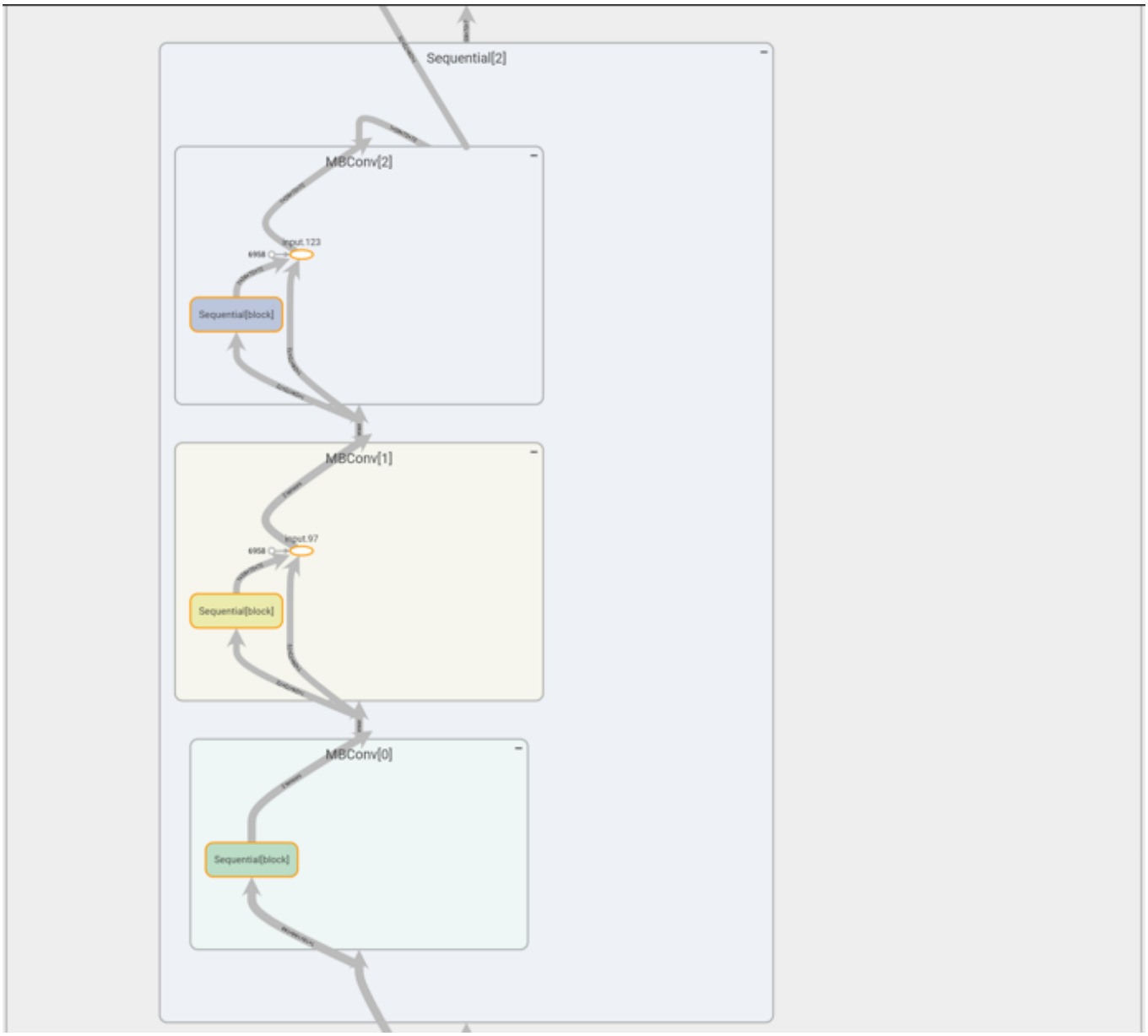


figure 21: sequential 2

figure 22: sequential 1

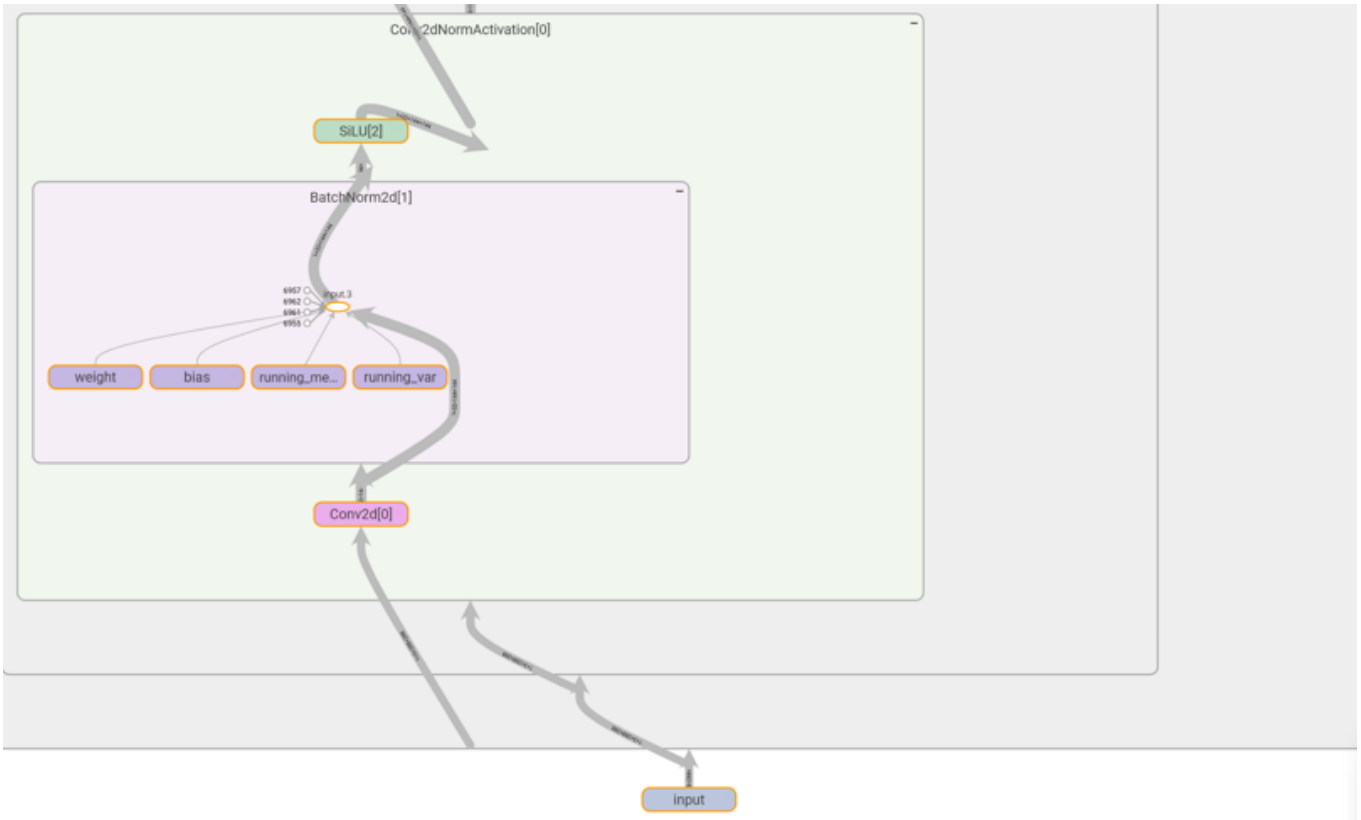


figure 23: initial convolution

4.3 system running

4.3.1 data augmentation block

This block gets the input of the image of the dataset and is responsible for the augmentation that happens in this order: Resizing-->CLAHE-->Poisson noise-->rotation with a 50% chance-->gaussian filter on the test set.

4.3.2 first convolution

This block waits for the image to be loaded and after the augmentation the images of size 1x3x288x288 are inputted into this first convolution layer. Then it outputs 1x32x144x144 tensors.

4.3.3 sequential 1

This block receives a 1x32x144x144 tensor input from the first convolution layer and then it applied one MB convolution on the received tensor then it outputs another image feature map of 1x16x144x144.

4.3.4 sequential 2

This sequential block gets passed onto a 1x16x144x144 feature map input from the sequential block 1 layer and then it applied two MB convolution on the received tensor then it outputs another image feature map of 1x24x72x72.

4.3.5 sequential 3

This block receives a 1x24x72x72 tensor input from the sequential 2 and then it applied two MB convolution on the received tensor then it outputs another image feature map of 1x48x36x36.

4.3.6 sequential 4

This sequential block gets sent a 1x48x36x36 feature map input from the sequential block 3 layer and then it applied three MB convolution on the received tensor then it outputs another image feature map of 1x88x18x18.

4.3.7 sequential 5

This block receives a 1x88x18x18 tensor input from the sequential 4 and then it applies four MB convolution on the received tensor then it outputs another image feature map of 1x20x18x18.

4.3.8 sequential 6

This sequential block gets sent a 1x20x18x18 feature map input from the sequential block 5 layer and then it applied five MB convolution on the received tensor then it outputs another image feature map of 1x208x9x9.

4.3.9 sequential 7

This block receives a 1x88x18x18 tensor map input from the sequential 6 and then it applies two MB convolution on the received tensor then it outputs another image feature map of 1x352x9x9.

4.3.10 last convolution layer

This block receives the input of sequential 7 block of size 1x352x9x9, but this time it outputs two feature maps for two different classifiers the ORI SoftMax classifier and the translation classifier.

4.3.11 orientation classifier node

This classifier receives the feature map from the last convolution layer of the efficentnet and then regress the orientation of the satellite outputting a text that identifies the orientation using the quaternions (Q1, Q2, Q3, Q4).

4.3.12 translation classifier node

This node receives feature map from the last convolution layer of the model and outputs the translation (X, Y, Z) of the satellite of from the image and put them as a text in an excel sheet ready for submission.

X Y Z Q1 Q2 Q3 Q4

img00001	0.231644	0.77466	-0.46388	0.362017	0.018649	-0.28098	7.335492
img00002	0.675757	0.352569	-0.45222	0.463187	-0.04031	-0.05594	4.995596
img00002	0.069401	-0.48254	-0.39559	-0.77836	-0.40629	-0.37355	7.046037
img00002	0.66837	0.264906	0.631009	-0.29144	0.117004	0.093756	15.07942
img00002	0.462752	-0.42023	0.760168	0.177247	-0.03282	0.000997	7.730878
img00003	0.792342	0.31862	-0.27059	-0.44436	-0.08687	0.484128	8.987622
img00003	0.305548	-0.92922	-0.0636	-0.19784	0.413922	-0.14514	17.13788
img00003	0.459784	0.743939	-0.0442	0.482907	0.036153	0.192857	4.562107
img00004	0.404937	0.745073	-0.52922	0.028533	0.468606	-0.13822	10.85755
img00004	0.045283	-0.86718	-0.07697	-0.48993	0.279724	-0.90878	12.88197
img00006	0.536348	0.189441	0.586975	0.576111	-0.03866	0.020825	8.862163
img00007	0.093254	-0.01997	0.887739	0.45036	0.175867	0.184969	6.682495
img00007	0.186534	0.395127	0.888349	0.14112	0.013738	-0.16137	9.403693
img00007	0.175585	0.297902	0.859795	-0.37573	0.336315	-0.30704	14.15538

figure 24: example of the shape of the output given multiple images to test on

Chapter 5: Results and Evaluation

5.1 Testing methodology

Multiple methodologies were used to confirm the testing and the evaluation of the model. First, a remembrance of previous knowledge presented in this paper shows that the dataset consists of real images and synthetic images. Synthetic images were used for the training process due to the low number of real images, back to our methodology synthetic images were used and split into 85% training and 15% validation and a batch size of 32 was used to train the model. After each training iteration the model is evaluated on the synthetic images then, on the real images. After the model is done training the final best model is finally tested on a second batch of real images and saved as a csv file that is then submitted to the original challenge website of the dataset for the actual evaluation of the model on the real testing data and be listed in the leaderboard. The metric used to evaluate the model performance is called the ESA score. The ESA score is calculated in several steps: first the translation score is computed by getting the norm of the translation error over the ground truth of the image.

$$score_{position}^{(i)} = \frac{\left| r_{gt}^{(i)} - r_{est}^{(i)} \right|_2}{\left| r_{gt}^{(i)} \right|_2}$$

Then the orientation score is calculated per image which is the angle of rotation that aligns the estimated angle and the ground truth angle.

$$score_{orientation}^{(i)} = 2 \cdot arccos\left(\left|\left\langle q_{est}^{(i)}, q_{gt}^{(i)} \right\rangle\right|\right)$$

Furthermore, the pose score is calculated by adding both previous elements.

$$score_{pose}^{(i)} = score_{orientation}^{(i)} + score_{position}^{(i)}$$

Then at last the total score of the ESA score is calculated by getting the average score of the pose score.

$N =$ number of images in test set

$$score = \frac{1}{N}\sum_{i=1}^{N} score_{pose}^{(i)}$$

The lower the ESA score the better the model in terms of accuracy.

5.2 Results

5.2.1 best results case

The best result was achieved when CLAHE and Poisson was in use, with a 32-batch size and onplateu learning to avoid overfitting of the data + the use of efficentnet model to increase the accuracy of the model. The model was trained for 15 epochs at first to prove its ESA score then it was trained on 50 epochs which further increased the accuracy of the model not only that but the model never over fitted when the number of bin for angles per regression increased making the model almost overfitting proof unlike the prior model which was MOBILEURSONET which is mentioned on chapter two.

5.2.2 average results case

Average finding were found when the use of the CLAHE and Poisson +the 32-batchsize and multistep learning rate using mobilenetv2, this model was tested on 15 epochs only however the results were very mediocre not to mention that it overfitted when the number of regression bins for the angle increased just like its prior model from MOBILEURSONET.

5.2.3 bad/worst results case

The usage of gaussian noise + 32-batchsize and a multistep learning rate using mobilenetv2 proved the worst results. This model was trained using 15 epochs only too however this model did not improve on learning values after the 8th epoch and it also made some of the scores NAN after the 10th epoch therefore no further testing of this model was done due to it being bad.

5.2.4 limitations

While the model proved promising results it suffered from a generalization problem in which the model had to sacrifice the number of classes for the orientation to get better results on real test images.However,this would often result in loss of overall accuracy of the model itself. However this was fixed by implementing more noise to the model fliping,gaussion filter, and Poisson noise.Moreover,the accuracy problem was fixed by using the CLAHE method to enhance the images itself. And while the model was fast it reached even faster inference by making the images smaller and exchanging the mobilenetv2 backbone with the new efficentnet backbone.

5.3.1 Evaluation

model	epochs	batchsize	No.parameters	Esa synth	ESA real	Generalization factor
Mobileursonet(8bins	50	32	2.2m	0.25	0.78	3.12
Mobileursonet(12bins	50	32	4.4m	0.21	1.22	5.8
Mobileursonet(16bins	50	32	7.4m	0.191	1.20	6.3
Mobileursonet regression	50	32	2.2m	0.61	0.79	1.2
My MODEL(best) Using 12 bins only	50	32	8.2m	0.189	0.58	3.05

As shown in the previous table: comparing our model to the previous mobileursonet models we can find that our model is superior in all aspects except the size of the model. But even that increase in size is very minimal regardless of all models except the regression model. However, while our model might be bigger, it solved the previous researchers of mobileursonet problem which was the generalization factor while maintaining good results, as seen in the comparison between the (regression model) and our best model. While the generalization of the regression model was good, but our results are better in terms of the score.in comparison to their best Esa score model in the synthetic category (the 16bin model) our model got a

slightly better results in the synthetic category. And in comparison, to their best Esa real score best model (the 8bin model) our model scores better Esa score overall than their 8bin real scores.

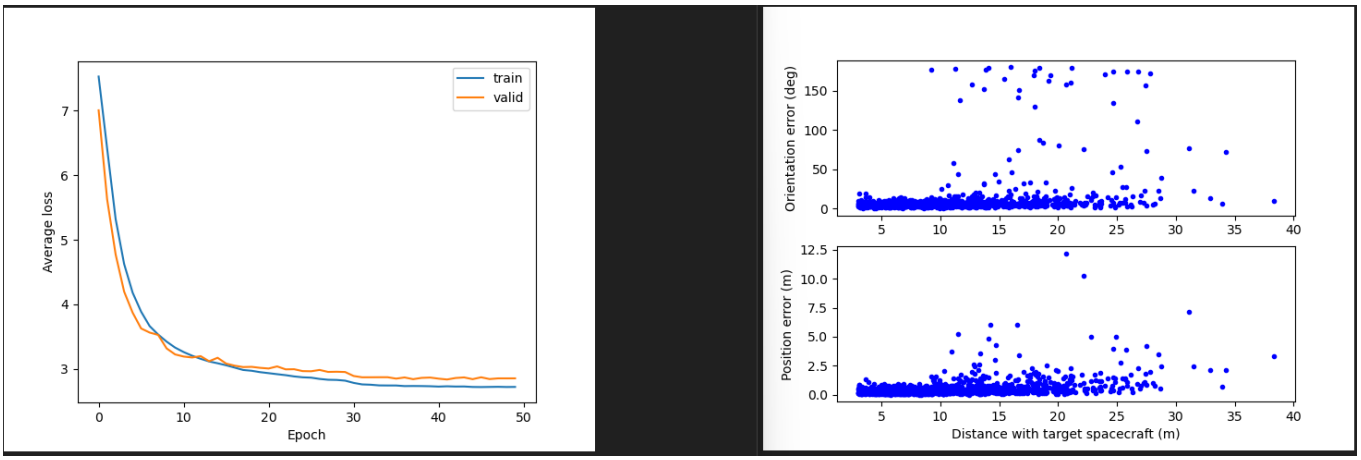


figure 25: train validation loss and error by distance for ori and trans

hamadascalob	1	May 20, 2024, 3:28 p.m.	May 20, 2024, 3:28 p.m.	0.5807256970157626	0.2223581343779088

figure 26: our model scores (Esa real left)/(best image score right)

Kelvins - Pose Estimation Challenge Post Mortem Leaderboard (esa.int) the site of the leaderboard

5.3.2 Time performance

While usage of the local machine was time consuming, it however evaded a lot of dependencies headaches and the loss of the data that could of happened if cloud was used also the model needed to be tested on a weak machine to make sure that it would be able to run on embedded small computers of the satellites.

Chapter 6: Conclusion and future work

6.1 Conclusion

In conclusion, a machine learning model was developed to estimate the pose (the orientation and the translation of a spaceship) in an image with respect to a camera. After exploring two different approaches it was concluded that the pure machine learning approach was the best in terms of applicability due to its robust and low model size and its low computational complexity with respect to the hybrid method. And in the shadow of the machine learning approach, we applied a very simple low computational complexity model even compared to normal machine learning models. And that model suggests the use of Efficentnet backbone with multiple important transformations to the image (CLAHE,Poisson noise, resizing). Furthermore, this backbone was connected to two fully connected layers that regress the position and the orientation of the sattelite.then the final pose is finally calculated and shown, and our model regressed the pose of the satellite on the speed dataset very successfully with minimal time and complexity.

6.2 problem issue

6.2.1 technical issue

-The model crashed when CLAHE was first used, this problem was fixed by adjusting the histogram values accordingly while testing.

-The model used a lot of rams that ran out when training, this problem was tackled by increasing the epoch size from 64 to 32 and decreasing the image size accordingly.

-GPU did not want to run at first using Cuda, this problem was solved by adjusting the TensorFlow and the Cuda versions to match.

-NAN values were present when using gaussian noise, fixed by using Poisson noise instead.

-Model size inflated hard when mobilenetv2 was exchanged to efficentnet,fixed by implementing an older efficentnet model that balanced between accuracy and size.

6.2.2 scientific issue

-While generalization was a problem, it was tackled by inflating the training dataset a little bit by adding noisy wrong images, so the model doesn't overfit.

-the difference between mobilenetv2 and efficentnet was slightly problematic however this was overcome by checking directories on GitHub and implementing the efficentnet from TensorFlow.

6.3 Future work

-As mentioned earlier, when the usage of newer efficentnet models is implemented, the model inflates significantly. For that problem I think there might me a solution if implemented efficentnet from a different source or to even remove efficentnet as a whole and use a newer more robust pre trained model.

-while the speed dataset is good, the model might regress better results if the upgraded version of the dataset SPEED+ was used.

-the testing on actual embedded computers like raspberry Pi would further support the applicability of the model.

-better data augmentation might yield better results on the generalization factor.

References

-Legrand, A., Detry, R., De Vleeschouwer, C. (2023). End-to-end Neural Estimation of Spacecraft Pose with Intermediate Detection of Keypoints. In: Karlinsky, L., Michaeli, T., Nishino, K. (eds) Computer Vision – ECCV 2022 Workshops. ECCV 2022. Lecture Notes in Computer Science, vol 13801. Springer, Cham. https://doi.org/10.1007/978-3-031-25056-9_11

- Park, T. H., Sharma, S., & D'Amico, S. (2019). Towards robust learning-based pose estimation of noncooperative spacecraft. arXiv preprint arXiv:1909.00392.

-J. Posso, G. Bois and Y. Savaria, "Mobile-URSONet: an Embeddable Neural Network for Onboard Spacecraft Pose Estimation," *2022 IEEE International Symposium on Circuits and Systems (ISCAS)*, Austin, TX, USA, 2022, pp. 794-798, Doi: 10.1109/ISCAS48785.2022.9937721.

Mukundan, R. (2012). Quaternions. In: Advanced Methods in Computer Graphics. Springer, London.