UNIVERSITÀ DI PISA

DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

Msc in Artificial Intelligence and Data Engineering

Multi-class semantic segmentation of satellite bodies imagery

Project report

Luca Caprioli

Valerio Secondulfo

Academic Year: 2022/2023

Contents

1 Introduction

Space situational awareness (SSA) systems play a significant role in space navigation missions. One of the most essential tasks of this system is to recognize space objects such as spacecrafts and debris for various purposes including active debris removal, on-orbit servicing, and satellite formation. The complexity of object recognition in space is due to several sensing conditions, including the variety of object sizes with high contrast, low signal to-noise ratio, noisy backgrounds, and several orbital scenarios. Existing methods have targeted the classification of images containing space objects with complex backgrounds using various convolutional neural networks. These methods sometimes lose attention on the objects in these images, which leads to miss-classification and low accuracy. In this work, new resources are proposed to aid the design of such missions and the implementation of new control algorithms and autonomous guidance techniques for satellites devoted to the inspection of non-cooperative targets before any proximity operation is initiated. In particular, the use of state-of-the-art Convolutional Neural Networks architectures performing image segmentation is proposed and its effectiveness in recognising features and parts of the target satellite is evaluated. The report proposes a comparison of various models, both trained from scratch and through transfer learning, all based on U-Net, one of the most famous architecture for semantic segmentation. Furthermore, the target objects were augmented by blurring, adding noise and light effects, and were then passed to the different models for training. The experiments were conducted by utilizing a recently developed space object dataset generated from realistic space simulation environments. The proposed image segmentation solution yielded sufficient performance, specifically for the transfer learning, and its feasibility, at least partially, for use in real-world SSA systems was demonstrated.

The project is available in the following link:
SatSeg

2 Space Situational Awareness

The concept of space situational awareness emerged in the 1950s with the launch of the world's first artificial satellite, Sputnik 1, by the Soviet Union. This event highlighted the need to track and monitor objects in space and from there on it became of paramount importance in our increasingly crowded space environment. More than 1 million pieces of 'space debris' are orbiting the Earth [1] and with 20,000 new satellites expected to be launched over the next decade, this number will increase – as too will the risk of collision in space. The very International Space Station has to perform regular debris avoidance maneuvers to mitigate the risk of these collisions with space debris. These maneuvers are based on space situational awareness data taken from different kinds of sensors (radar systems, optical systems, LiDAR systems etc.).

In 2009 the European Space Agency's started the SSA Programme designed to support Europe's independent space access and utilization through the timely and accurate information delivery regarding the space environment, particularly hazards to both in-orbit and ground infrastructure. In 2019 it evolved into the present Space Safety Programme with an expanded focus, also including missions and activities to mitigate and prevent dangers from space. The programme is split into four main segments:

- Space weather: monitoring the Sun, the solar wind, and in Earth's magnetosphere, ionosphere and thermosphere, that can affect spaceborne and ground-based infrastructure or endanger human life or health.
- Planetary Defence: detecting natural objects, such as asteroids and comets, which can potentially impact Earth, gathering observations from telescopes around the world and plotting their path through the sky to calculate the impact risk, as well as coordinating with the international community the response to a possible impactor.
- Space debris: Tracking active and inactive satellites and space debris to better understand the debris environment; providing data, analysis and advice to spacecraft engineers to perform collision avoidance manoeuvres as well as developing a system of automated collision avoidance.
- Clean Space: systematically considering the entire life-cycle of space activities, from the early stages of conceptual design to the mission's end of life – and even beyond, to removal of space debris.

The Space Safety programme is being implemented as an optional ESA programme with financial participation by 14 Member States while the United States Strategic Command (USSTRATCOM) is responsible for maintaining space situational awareness on behalf of the U.S. government, operating the Space Surveillance Network (SSN), a global network of radar and optical sensors to track space objects. Commercial entities are also playing

an increasing role in space situational awareness. Companies like LeoLabs and ExoAnalytic Solutions use ground-based radar systems to track space objects and provide data services to government and commercial customers. Space agencies and organizations are actively exploring methods to mitigate space debris and enhance space situational awareness.

In recent years, new technologies such as computer vision and machine learning have been applied to space situational awareness to improve tasks such as object tracking, anomaly detection, and collision prediction capabilities. Regarding computer vision, in particular semantic segmentation, its role in this domain could be summarized by the following tasks:

- Identification of Spacecraft Components: Identify and classify different components of a spacecraft, such as the main body, solar arrays, communication devices, or scientific instruments. This information helps in understanding the overall architecture and functionality of the spacecraft.
- Collision Risk Assessment: Analyzing close approaches between spacecraft and space debris by segmenting and tracking their respective pixels. This allows for the accurate calculation of collision probabilities, aiding in collision avoidance maneuvers and space traffic management.
- Spacecraft Health Monitoring: Monitor the health of a spacecraft by segmenting and analyzing specific regions of interest, such as thermal radiators or propulsion systems. Deviations from expected segmentation patterns can indicate potential anomalies or malfunctions, triggering necessary maintenance or corrective actions.

This project was born first and foremost from the interest of its members in the space industry, both from a scientific and environmental standpoint. The idea of sending man-made objects to explore space is something incredible, but it should always be done with consciousness. The goal is to create a sub-system that could implemented in one of these missions to automatize the identification and classification of space objects, in particular artificial satellite bodies. By training and comparing different models and by leveraging the general capabilities of semantic segmentation, space agencies and operators can enhance their ability to monitor, analyze, and respond to dynamic space environments, facilitating safer and more efficient space operations. Like the earth and the sea, the sky is beautiful, for this it must be preserved.

3 State of the Art

Before going over the different algorithms and architectures that were considered for this work, it is appropriate to introduce the most common tasks they are asked to perform in computer vision. They are listed hereafter:

- Image classification: it is the task that has been used as an example up to now. An image classification algorithm just identifies which kind of object, if any, is primarily depicted in the image among the ones the network has been trained to identify.
- Object detection: the algorithm is capable to distinguish various objects in each image, drawing for each of them a bounding box.
- Semantic segmentation: rather than finding the envelope of each object as in the previous case, the algorithm is able to tell exactly which pixels belong to the object. However, this category of networks is not able to distinguish between objects of the same type, thus considering them as a single instance.
- Instance segmentation: it is an hybrid of object detection and semantic segmentation. It is, in fact, able to find the area of the picture covered by a certain object and distinguish among different instances of the same type.

For this project semantic segmentation was chosen as the most suitable computer vision task for the problem at hand. The next forward step could have been instance segmentation but, as for the missions described before, there was no advantage in doing so, as all of them generally gain very little value in recognizing every instance of similar objects. In any case, as it will be described in the next section, the dataset is made up mostly of single objects, so a more pragmatic limitation was posed in this regard.

3.1 Training method and metrics

Cross-entropy loss and focal loss are commonly used loss functions for the semantic segmentation task. Both loss functions aim to measure the discrepancy between predicted segmentation outputs and ground truth labels, but they differ in their approaches to handling class imbalance and focusing on challenging samples.

- **Cross-entropy loss** is a standard loss function used for multi-class classification tasks. In the context of semantic segmentation, the cross-entropy loss considers each pixel as an independent classification problem. It calculates the loss for each pixel individually and then averages the losses across all pixels in the image as follows:

$$L_{CCE}(y, p) = -\frac{1}{N}\sum_{i=1}^{N}\sum_{c=1}^{C} y_{i,c}\log(p_{i,c})$$

where $y_{i,c}$ uses a one-hot encoding scheme of ground truth labels, $p_{i,c}$ is a matrix of predicted values for each class, and where indices c and i iterate over all classes and pixels, respectively. However, cross-entropy loss can be sensitive to class imbalance, which is common in semantic segmentation tasks. If certain classes are underrepresented in the training data, the model may prioritize the dominant classes and struggle to learn accurate representations for the minority classes.

- **Focal loss** is a modified form of cross-entropy loss that addresses the issue of class imbalance and focuses more on challenging samples. The focal loss introduces a modulating factor called the "gamma". This factor down-weights the loss for well-classified pixels, reducing the impact of easy samples and emphasizing the importance of hard, miss-classified samples. The formula for focal loss incorporates the gamma factor and modifies the standard cross-entropy loss. It can be expressed as:

$$L_{CF}(y, p) = \alpha(1 - (p_{t,c}))^{\gamma} \cdot L_{CCE}$$

where alpha is now a vector of class weights, $p_{t,c}$ is a matrix of ground truth probabilities for each class, and L_{CCE} is the categorical cross entropy loss defined before. Lower values of gamma increase the importance of easy examples, while higher values give more weight to hard examples.

Evaluation of semantic segmentation models is typically done using metrics such as Intersection over Union, pixel accuracy, and class-specific metrics. These metrics measure the accuracy and quality of the segmentation results by comparing the predicted segmentation with the ground truth. Intersection over Union (IoU), also known as the Jaccard index, is a commonly used evaluation metric for assessing the performance of semantic segmentation tasks. It is calculated by dividing the area of overlap between the predicted segmentation and the ground truth by the area of union between them. The formula for IoU is as follows:

$$IoU = \frac{TP}{(TP + FP + FN)}$$

Where TP, FP and FN are obviously computed per pixel. The IoU metric ranges from 0 to 1, where a value of 1 indicates a perfect overlap between the predicted segmentation and the ground truth, while a value of 0 indicates no overlap. Additionally, IoU can be computed per class to assess the performance for individual classes in multi-class segmentation tasks.

3.2 U-Net architecture

U-Net is a convolutional neural network (CNN) architecture [2] specifically designed for semantic segmentation tasks in computer vision, where the goal is to classify and segment different objects or regions within an image. The name "U-Net" comes from its U-shaped

architecture, which consists of an encoder pathway and a corresponding decoder pathway 1.

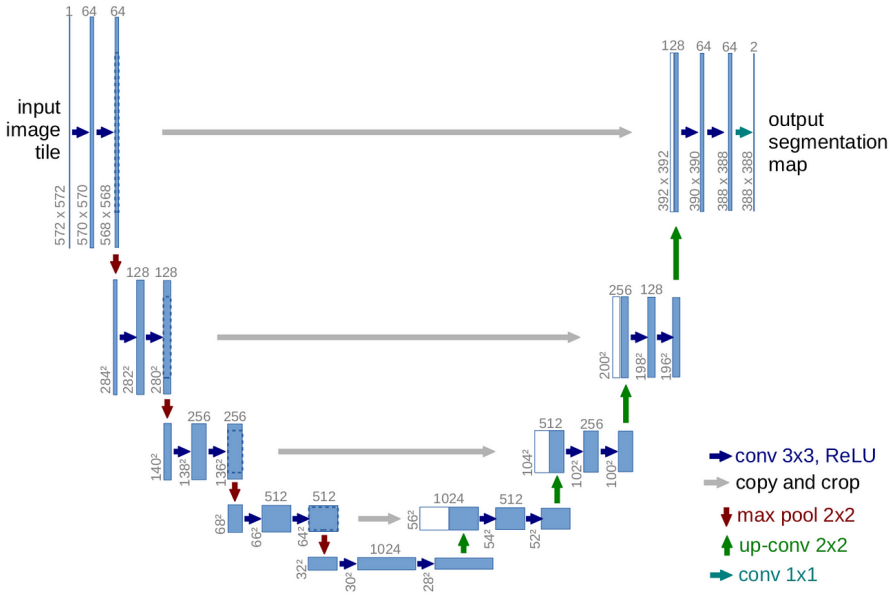

Figure 1: U-Net architecture

The encoder pathway captures the context and learns high-level features from the input image, while the decoder pathway recovers the spatial information and generates a pixel-wise segmentation map. The U-Net architecture is known for its skip connections, which are a crucial component of the network. These skip connections enable the transfer of feature maps from the encoder to the corresponding decoder layers. By doing so, U-Net combines both coarse and fine-grained features, allowing the network to leverage low-level and high-level information during the segmentation process. This helps in preserving spatial details and improving the segmentation accuracy.

The U-Net architecture has been widely adopted and has achieved remarkable performance in various medical image analysis tasks such as tumor segmentation, cell segmentation, and retinal vessel segmentation but in this case was adapted and tested for a completely different kind of task to see how it could perform. Over the years, several variants and modifications of the U-Net architecture have been proposed, each aiming to improve its performance or address specific challenges in different applications. These variations include U-Net++, Attention U-Net, and U-Net with residual connections, among others. This report will analyze specifically two kinds of U-Net architectures, the original and the Attention one.

3.3 ResNet-34

After building-up and training a U-Net base network from scratch, the second part of this project utilized another kind of encoder backbone to train the decoder block of a U-net

architecture. The chosen encoder was a ResNet-34 [3] network architecture, as from the possible choices it is the best bet considering its complexity and number of parameters related to the shape of the chosen dataset. ResNet-34 is a variant of the Residual Neural Network architecture specifically designed with 34 layers, including convolutional layers, pooling layers, and fully connected layers. It's main innovation, as the other ResNet architectures, is the use of residual connections or skip connections. These connections allow the network to learn residual mappings, which capture the difference between the desired output and the current output of a layer. By propagating this difference through the network, it alleviates the vanishing gradient problem and enables the training of much deeper networks.

In ResNet-34, the basic building block is the residual block, which consists of two or three stacked convolutional layers with batch normalization and rectified linear unit (ReLU) activations. These layers perform feature extraction and transformation. The residual connection bypasses these layers and directly adds the input to the output, creating a shortcut path for the information to flow through the network. Its depth and residual connections allow it to capture intricate patterns and learn highly discriminative features. It strikes a balance between model complexity and performance and has achieved competitive results on benchmark datasets.

3.4 VGG16

As a third part, the backbone encoder of U-Net was changed with a VGG16 encoder. VGG16 [4] is a convolutional neural network architecture widely used for image classification and object recognition tasks. It was developed by the Visual Geometry Group at the University of Oxford. The architecture of VGG16 consists of 16 layers, including 13 convolutional layers and 3 fully connected layers. The convolutional layers are stacked one after another, and they employ small 3x3 filters with a stride of 1 pixel. This design choice allows VGG16 to learn complex features and capture fine-grained details in images.Between the convolutional layers, VGG16 uses max pooling layers with a 2x2 filter and a stride of 2. This downsampling process helps increase the receptive field of the network and makes it more robust to translation variations. The fully connected layers at the end of the network serve as the classifier. They take the high-level features extracted by the convolutional layers and map them to the desired output classes. For this project, these last convolutional layers were used as the bridge for the U-Net decoder.

4 Dataset

The dataset used for this project [5] was crafted specifically for object detection and semantic segmentation tasks. It consists of 3117 images with uniform resolutions of 1280 × 720 2. It includes, for each image, a mask 3 of at most three parts (body, solar panels and antennas) of usually a single spacecraft.

Figure 2: Example of training image (resized 512x512)

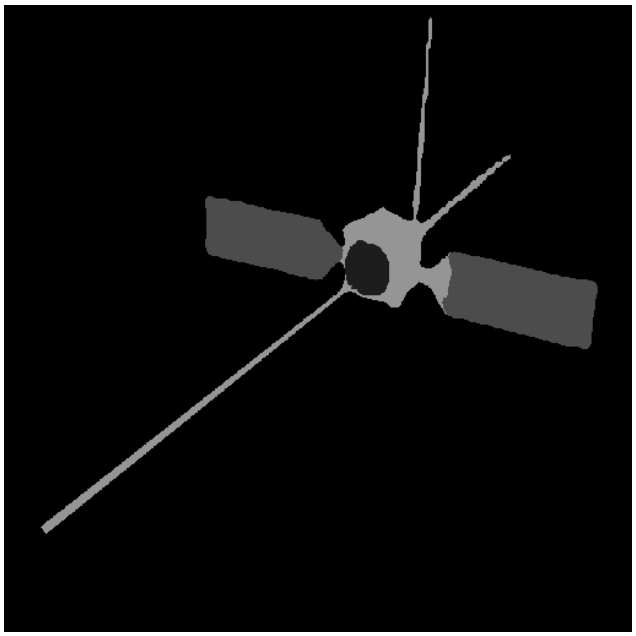

Figure 3: Example of training mask (resized 512x512)

The spacecraft objects are also various in range, they can be as small as 100 pixels

to as large as occupying nearly the whole images. As already mentioned each image of the dataset contains one or more artificial satellites and for each image a corresponding mask i present, labeled with a similar name. Each mask contains pixels that identify, with different colors (red, yellow, blue) parts of the corresponding, the different modules of the spacecraft. The background is identified by black pixels

4.1 Pre-processing

The original dataset was divided into a training and a test subsets, which consisted of 2517 and 600 images respectively, but it proved to be a problem because there was a strong imbalance between the class (pixels) identifying the antennas and the ones pinpointing the other two classes 4, plus obviously the background, as always in semantic segmentation problems.

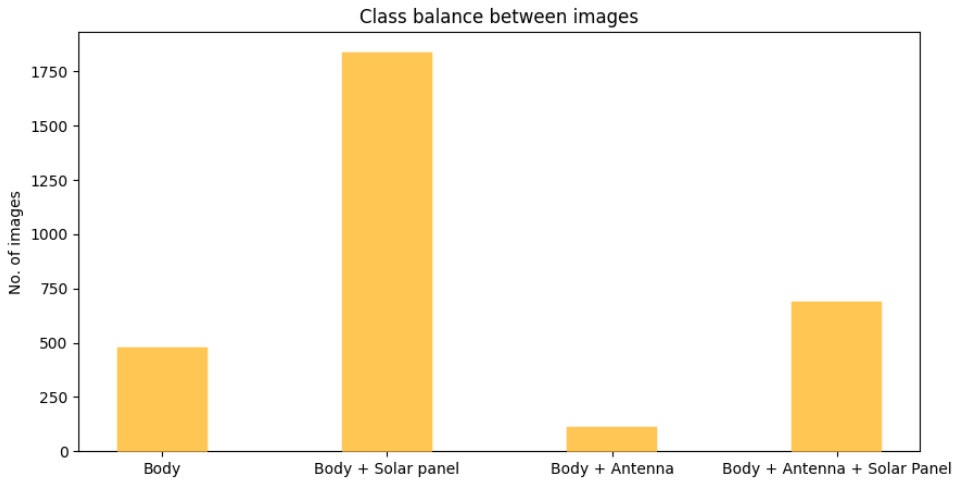

Figure 4: Class presence

Hoping to partially overcome this problem, the dataset was regrouped in a single folder and then split in train set, validation set and test set following a stratified K-fold technique, to at least have a coarse-grained control of the imbalance between the images. In the end the three sets contained respectively 2455, 624, and 38 samples, with each split containing 25% of images with the minority class. Obviously them same procedure was executed at the same time for the masks. While executing the pre-processing, the new images where resized to 512x512 to speed up their loading during the training step and given the fact that most models, and in particular U-Net, were trained on image of even smaller size.

4.2 Data augmentation

After splitting the dataset in train, validation and test set, another problem arose. The number of images (and masks) to properly train even the most basic of U-Net based neural network was too small. To mitigate this issue, and to tackle yet again the problem of the

imbalanced dataset, several passes of data augmentation were executed. The training set contained, as already mentioned, 2455 images of which 613 contained at least one subject with pixels of the minority class. With this in mind the three passes of data augmentation were executed only on the images of the training set containing the minority class 5 6. This helped the training set reach a more suitable number of samples, 4297. The data augmentation pipeline was specifically crafted for the task at hand in order to make the synthetic images as realistic as possible, albeit randomly generated. The pipeline starts with spatial transformations, choosing randomly at most two of the ones described below, all with the same probability:

- Vertical flip on the x-axis.
- Random rotation of 90°/180°/270°.
- Random rotation of 45° max.
- Random maximum translation of 20% the image dimensions on both axis.
- Random crop of an image portion containing a non-empty mask, 70% of the image dimensions.
- One between adding a defocus or a motion blur effect to the image.

After applying a combination of these transformation the images went through another kind of augmentation pipeline, much more specific to the problem of computer vision in a space environment, this time aiming at changing the colors and applying filters on the image. In this case only one of the filter listed below was chosen randomly, again all with the same probability:

- A random brightness shift in a tolerable range.
- A shadow of random dimension on the target object
- A random number of sun flares (max 10) in shades of one between three different colors (white, yellow, cyan).
- A random Gamma shift, again in a tolerable range.

For the filter containing more than one choice, all were considered with equal probability. Having almost doubled the dimension of the training set, the whole dataset was ready for experimenting, hopefully reaching good results.

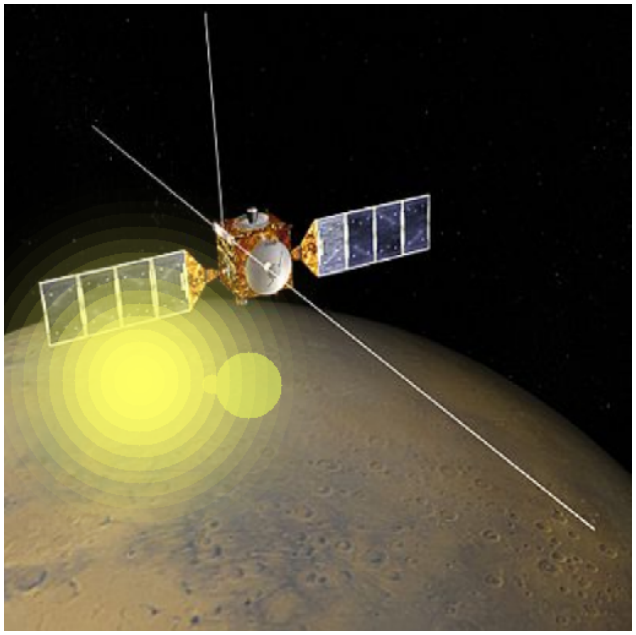

Figure 5: Augmentation of the example image

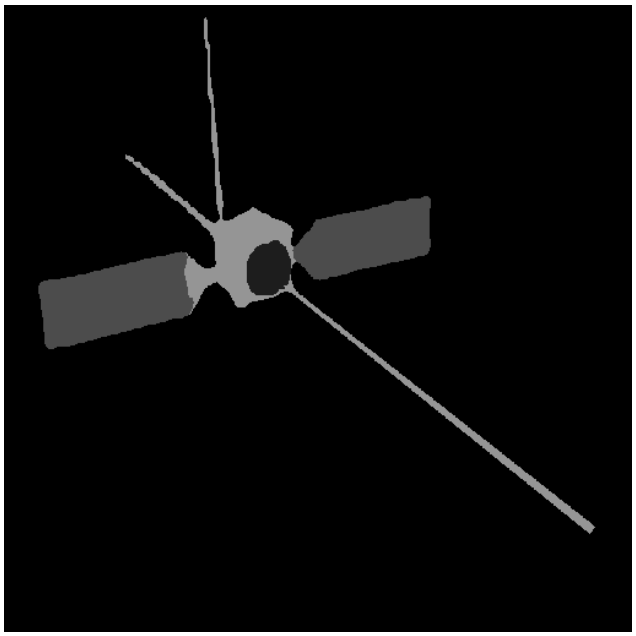

Figure 6: Augmentation of the example mask

5 U-Net network from scratch

The first part of this work revolved around training a U-Net based network from scratch to evaluate its performance on the task described till now. Generally the idea was to start from an architecture as small as possible, increasing its dimension when its full potential was reached, both in terms of training and validation results, after taking into account all the possible measures. Each successive step carried the results and the implementation of all the steps before.

5.1 Experiment One

For the first experiment the smaller possible U-Net architecture was chosen, composed of only one encoder block and on decoder block. It was trained for 200 epochs, and even with the relatively small starting dataset (to remember, 2445 training images) it reached pretty quickly its full potential 7 8. The results are reported below:

	mean IoU score	**Loss**
Training	45%	0.2533
Validation	44%	0.2634

Table 1: Experiment One report

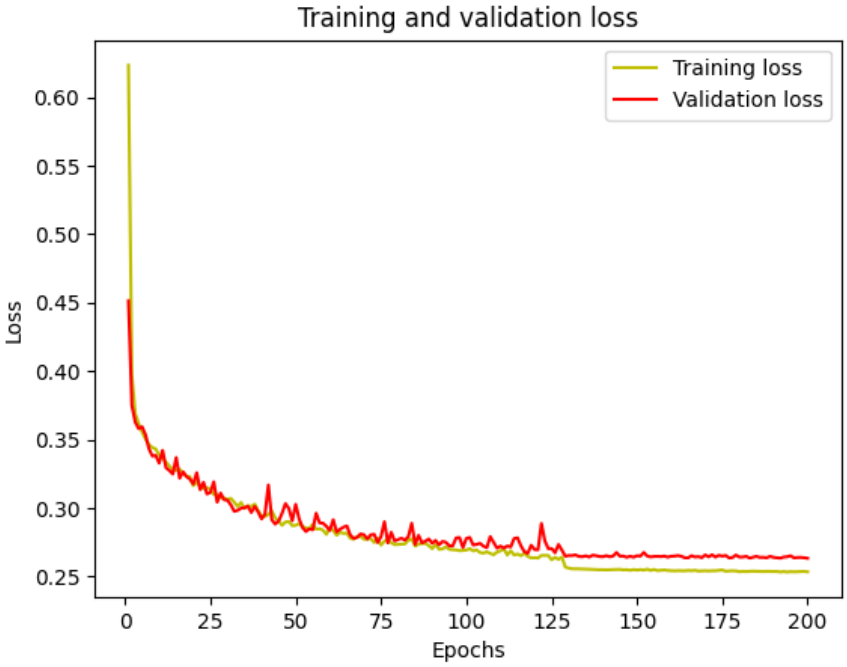

Figure 7: Training and Validation losses

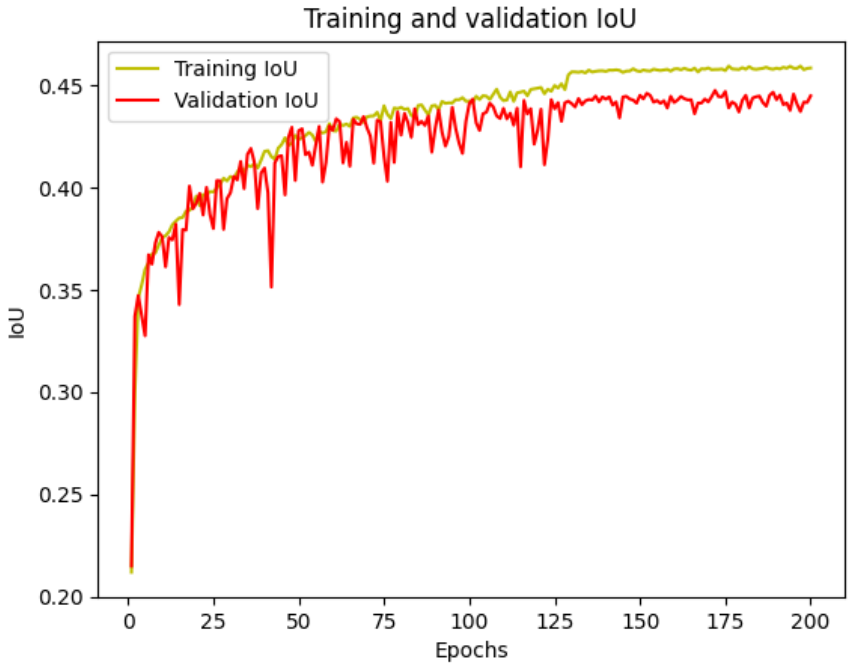

Figure 8: Training and Validation IoU metrics

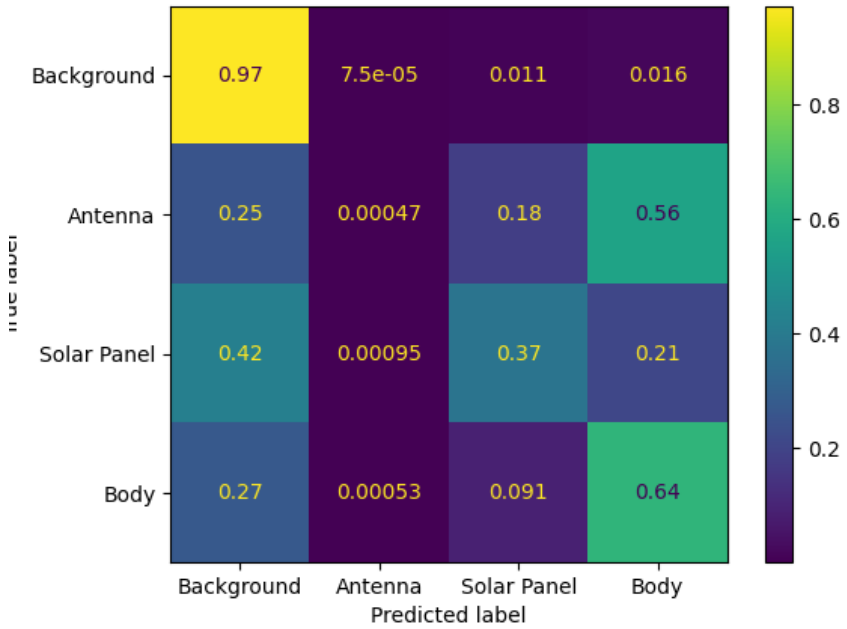

Figure 9: Confusion matrix

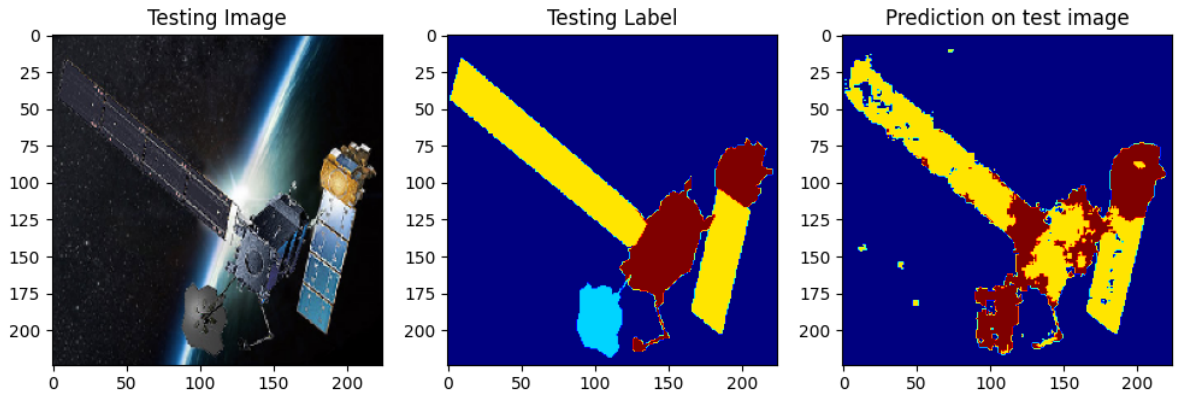

Figure 10: Prediction on test image

This first experiment results were relatively good as the network can more or less locate and segment the subject 10, but as seen from the confusion matrix 9 it mix basically all the classes and obviously always mistake the minority class for the other two.

5.2 Experiment 2

In the second experiment the previous architecture has been improved by adding a second block. This decision was taken considering that, with only one block in the encoding/decoding part of the network, the training saturated it's capacity and no additional improvement could be done on it. Here we have two blocks for encoding and decoding.

	mean IoU score	**Loss**
Training	60%	0.1369
Validation	52%	0.1979

Table 2: Experiment 2 report

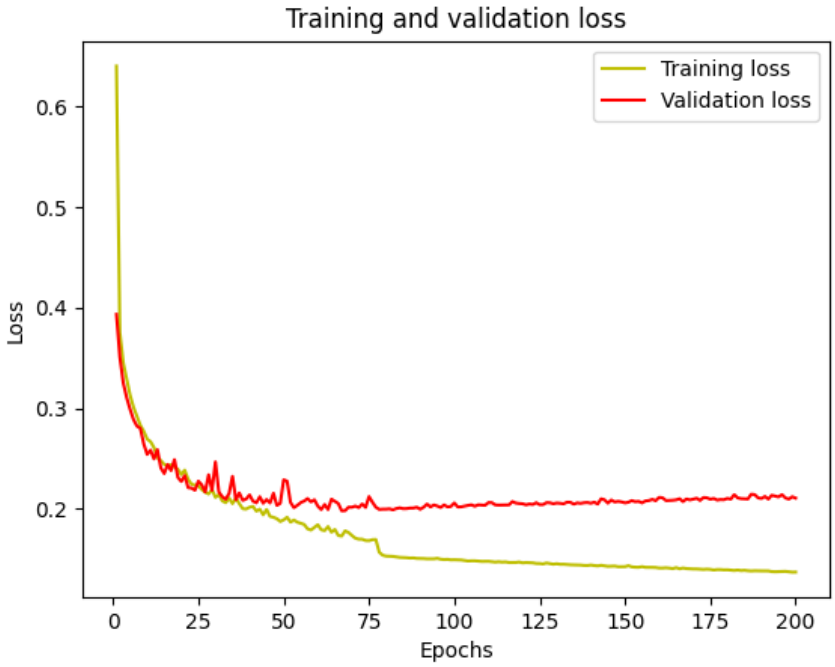

Figure 11: Training and Validation losses

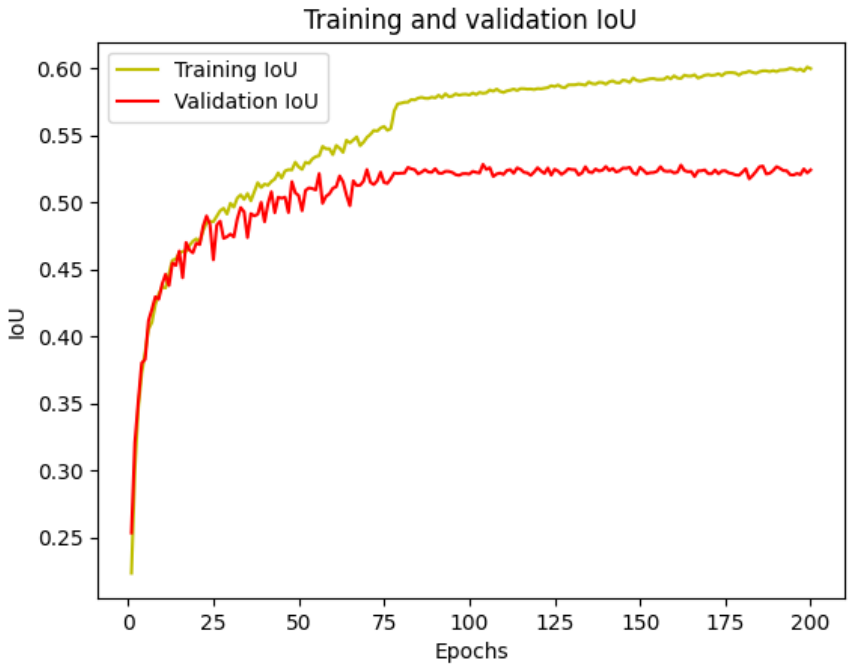

Figure 12: Training and Validation IoU metrics

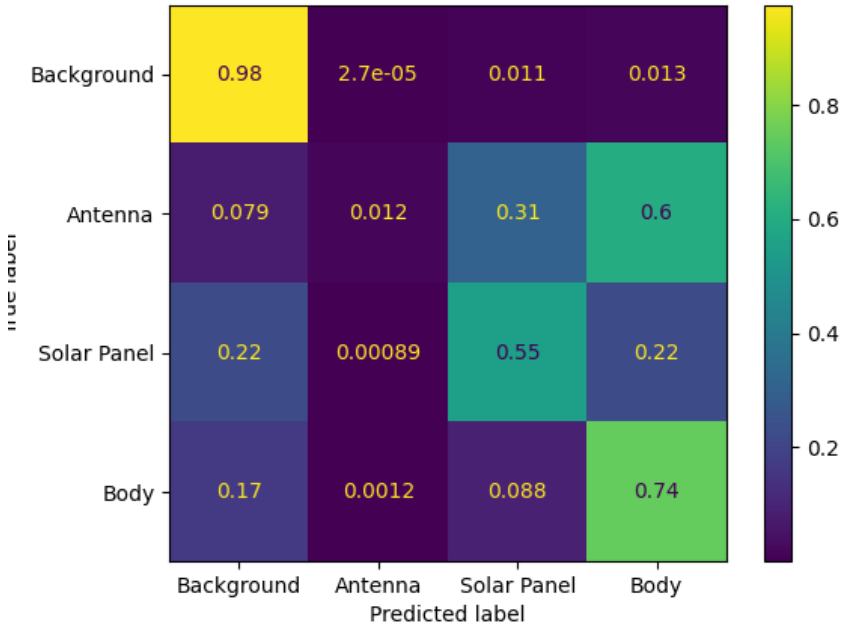

Figure 13: Confusion matrix

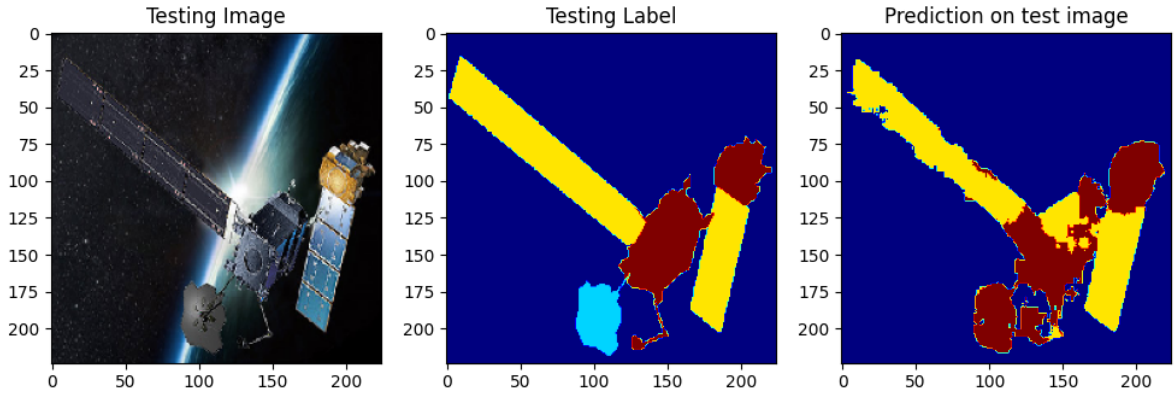

Figure 14: Prediction on test image

11 and 12 show a huge problem of overfitting with a training loss going down to zero and the validation loss slowly going up. This can also be checked easily by observing the IoU of the two function.

5.3 Experiment 3

Here the network was trained using 2 blocks for encoding/decoding and by increasing the training set by adding the first augmentation block. 15 and 16 show an increased performance with better results on the loss functions but not so good results in the IoU as the overfitting was still high.

	mean IoU score	Loss
Training	58%	0.1700
Validation	53%	0.2004

Table 3: Experiment 3 report

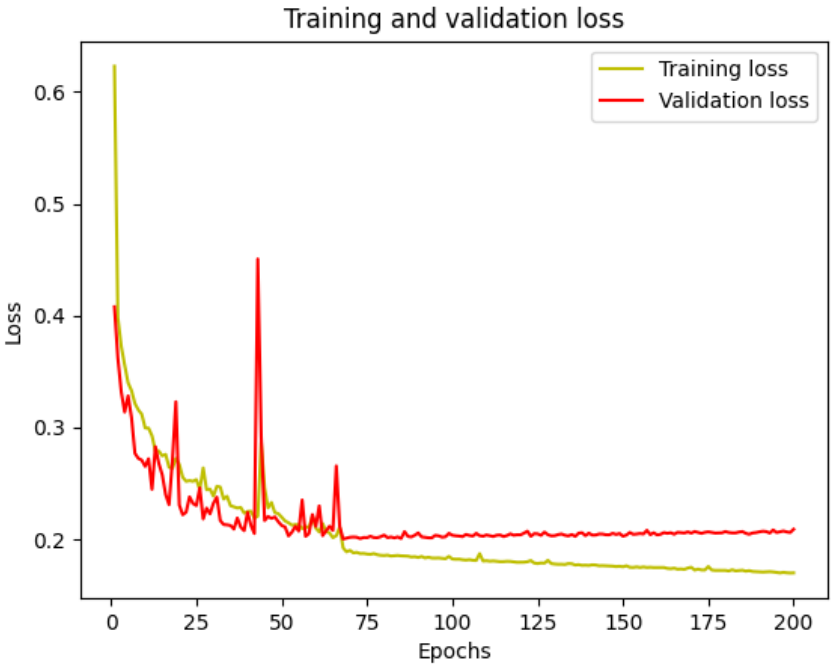

Figure 15: Training and Validation losses

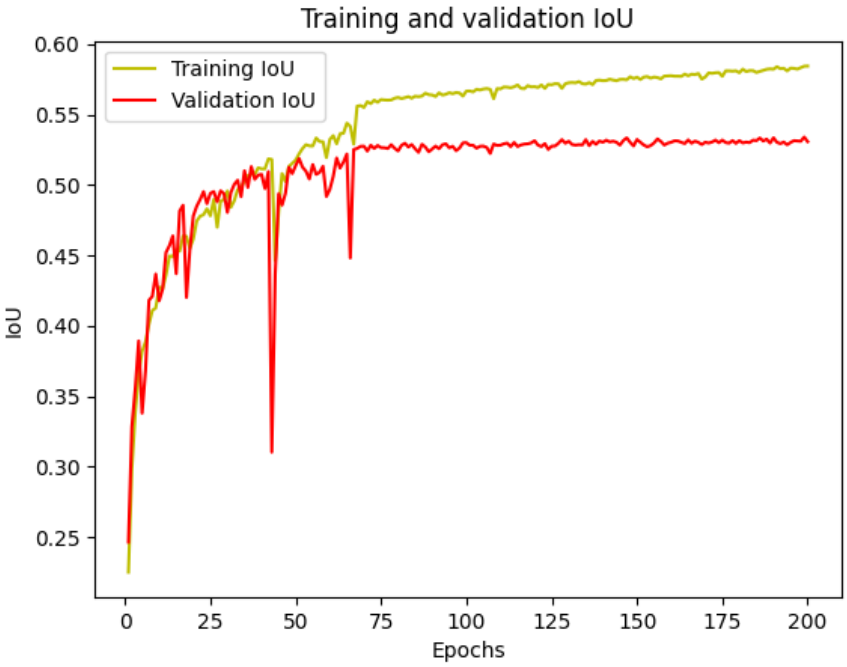

Figure 16: Training and Validation IoU metrics

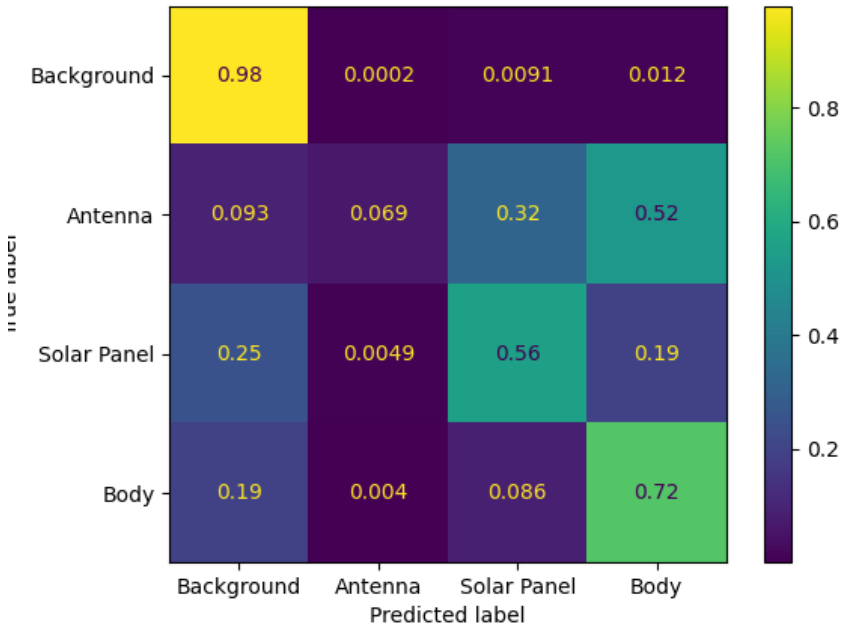

Figure 17: Confusion matrix

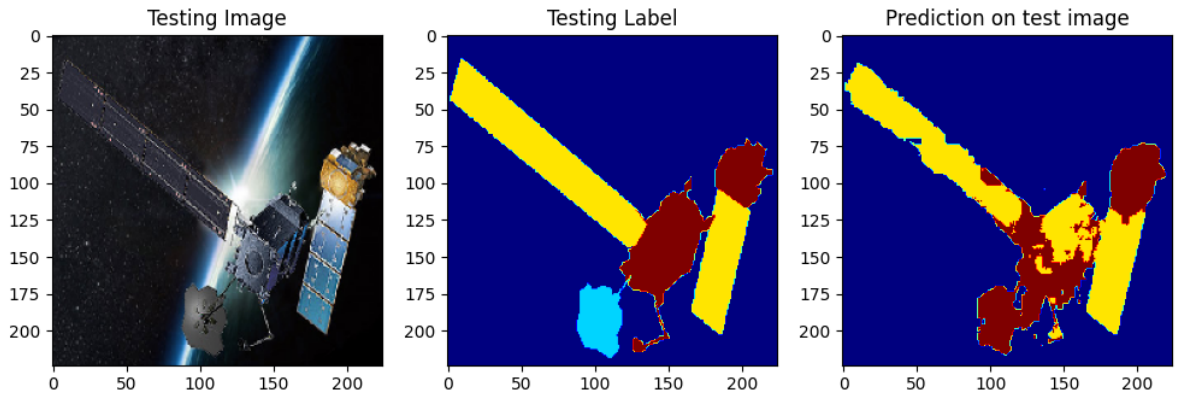

Figure 18: Prediction on test image

5.4 Experiment 4

An increment of the training set was performed in order to increase the dataset by adding to the set the first and second augmentation's blocks. 19 and 20 show less high peacks in curves than the previous experiment but, as it's easy to see, validation and training IoU went too far from each other due to the overfitting that kept increasing over the epochs with the validation stabiled around 0.54 and the training going up over 0.60.

	mean IoU score	**Loss**
Training	60%	0.1705
Validation	54%	0.1982

Table 4: Experiment 4 report

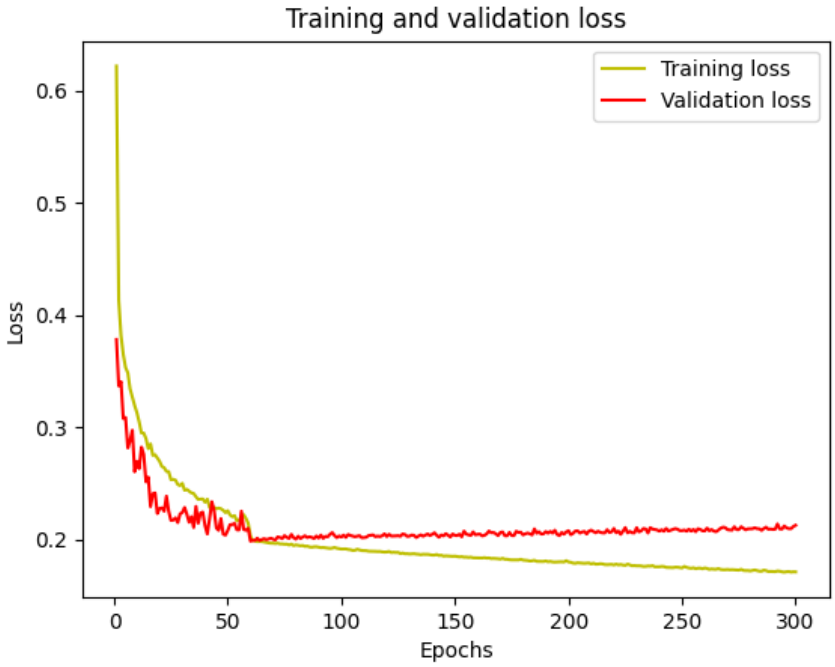

Figure 19: Training and Validation losses

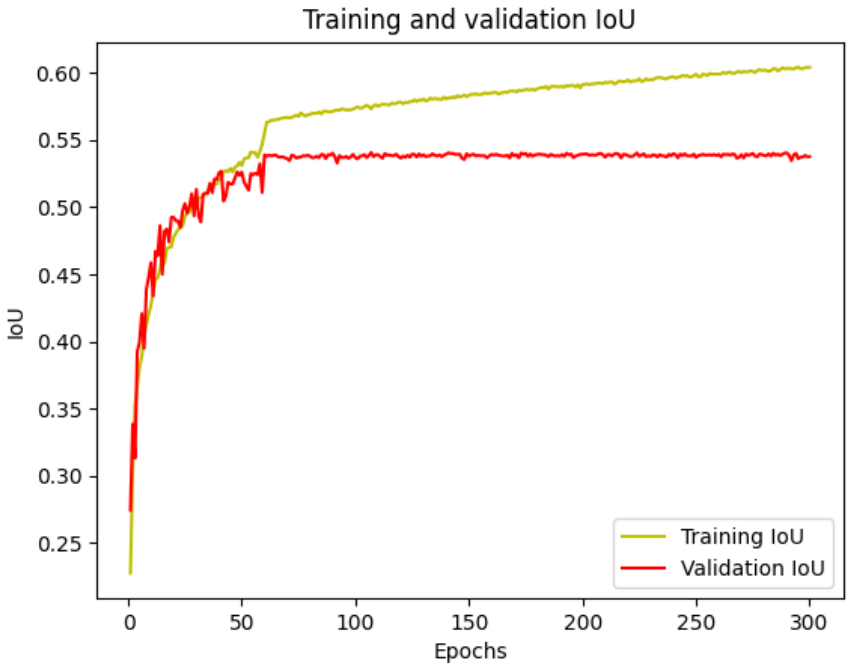

Figure 20: Training and Validation IoU metrics

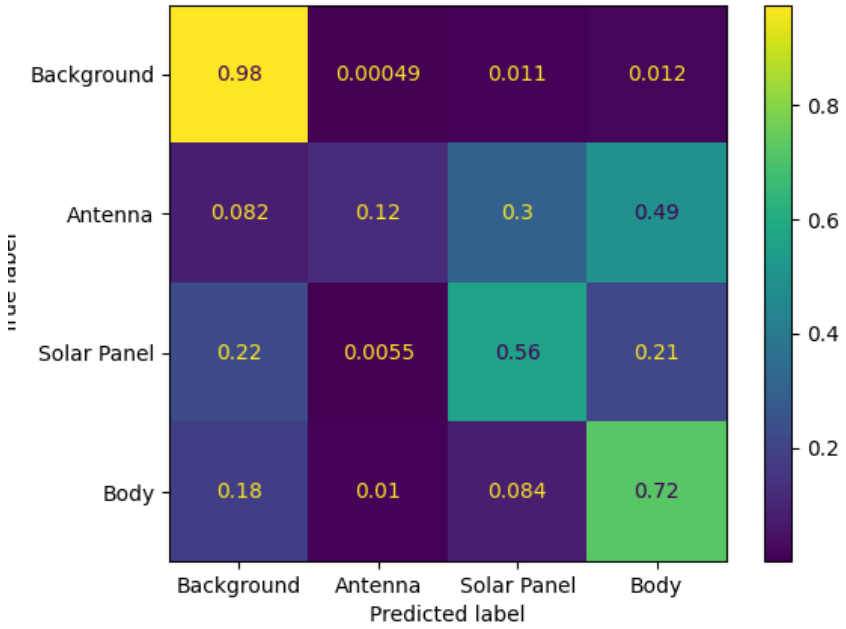

Figure 21: Confusion matrix

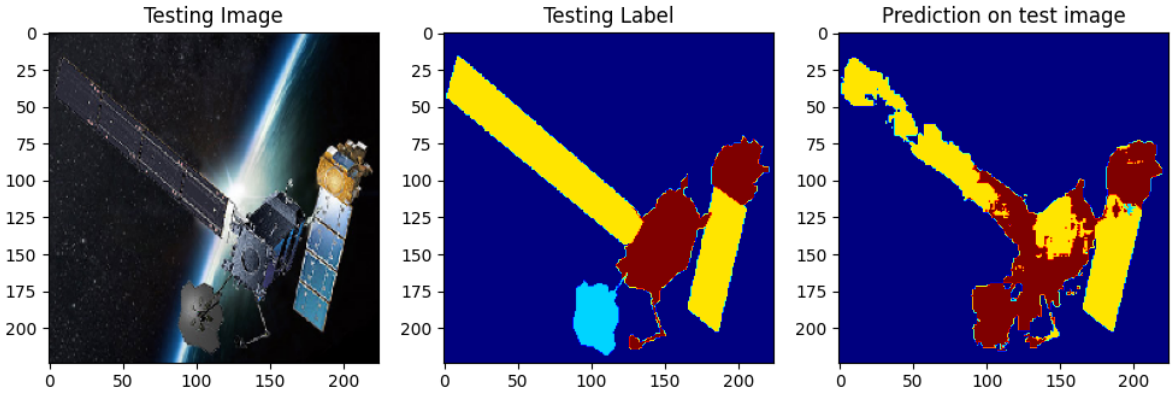

Figure 22: Prediction on test image

5.5 Experiment 5

A the third augmentation block was added to the training set. Having a bigger training set gave better performances and helped to understand that the initial dataset wasn't big enough. From 23 and 24 show that the results were not so different from the experiment 4 and the network had to be improved.

	mean IoU score	Loss
Training	59%	0.1895
Validation	54%	0.2031

Table 5: Experiment 5 report

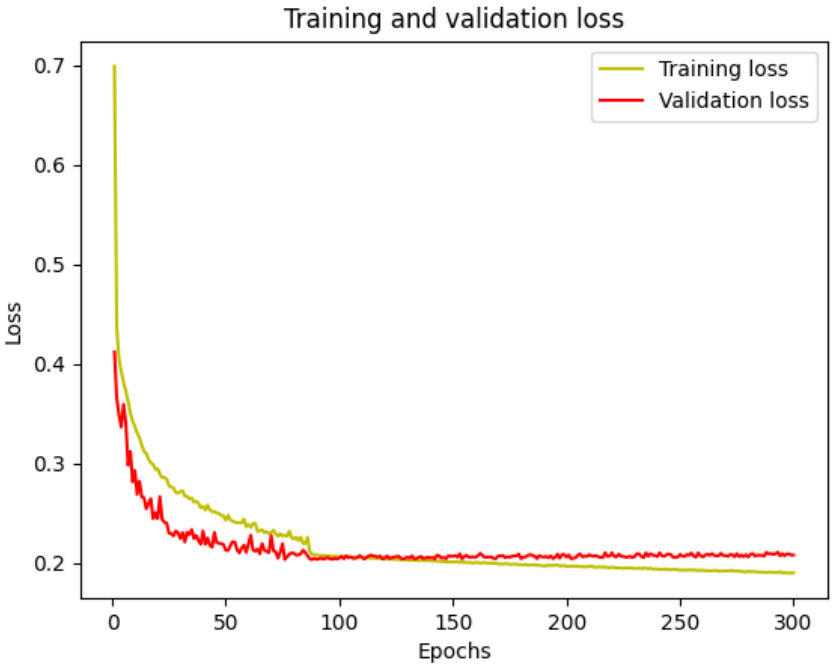

Figure 23: Training and Validation losses

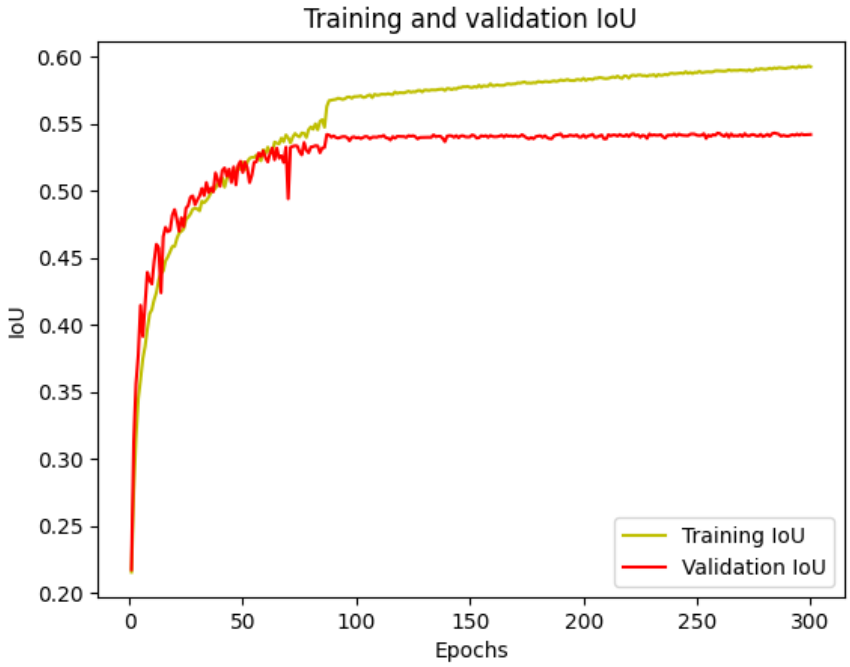

Figure 24: Training and Validation IoU metrics

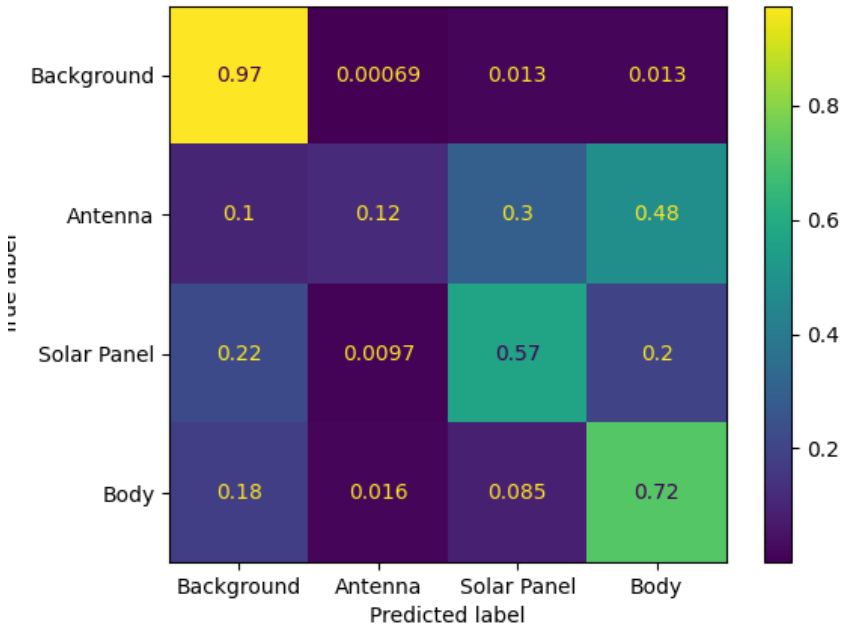

Figure 25: Confusion matrix

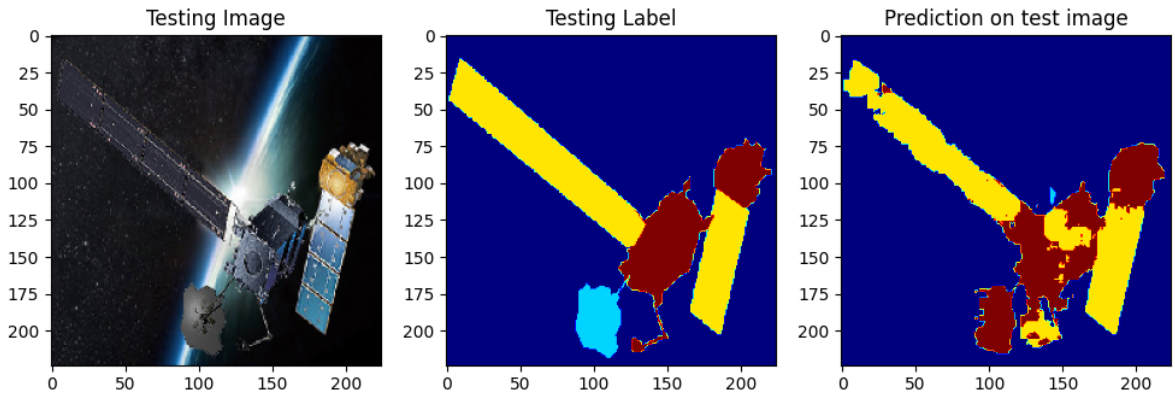

Figure 26: Prediction on test image

5.6 Experiment 6

Mantaining the training set of the previous experiment, a dropout layer was added into each block after every convolutional layer. Initially the values used for the dropout layers were 0.1 inside the encoding/decoding blocks and 0.2 inside the bridge [6]. 27 and 28 show the effects of the dropout layers on the newtork as the overfitting started lowing and training and validation IoU and loss values became more stable and similar to each other.

	mean IoU score	**Loss**
Training	55%	0.2141
Validation	54%	0.1954

Table 6: Experiment 6 report

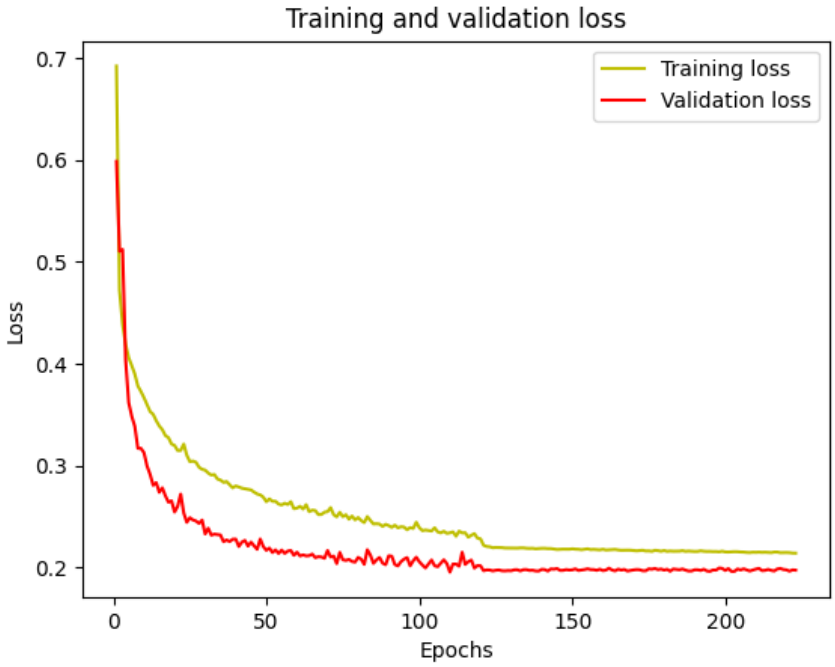

Figure 27: Training and Validation losses

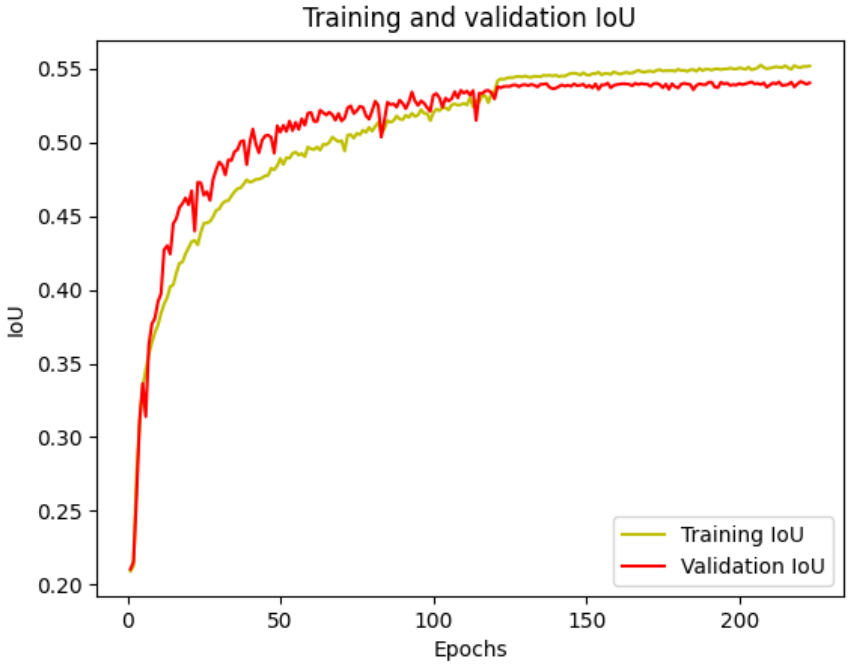

Figure 28: Training and Validation IoU metrics

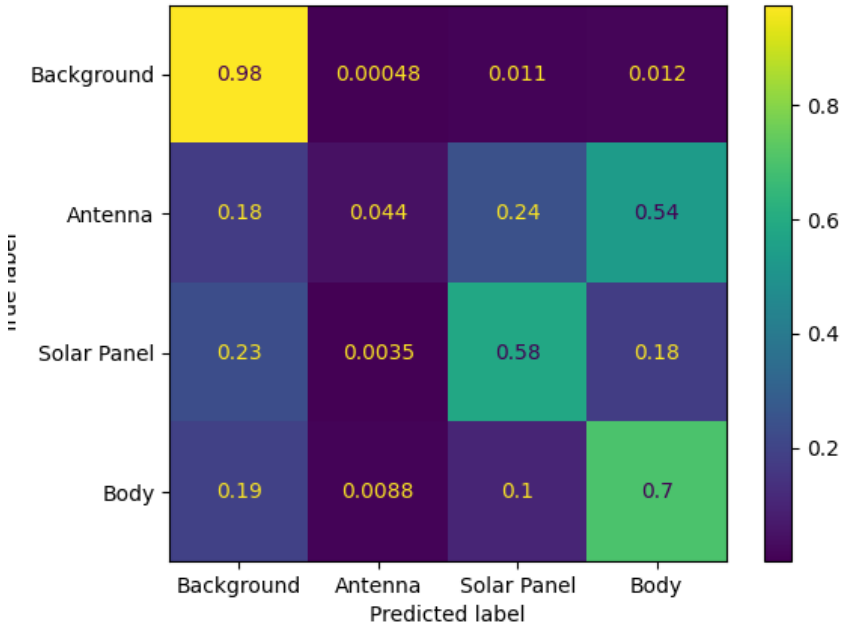

Figure 29: Confusion matrix

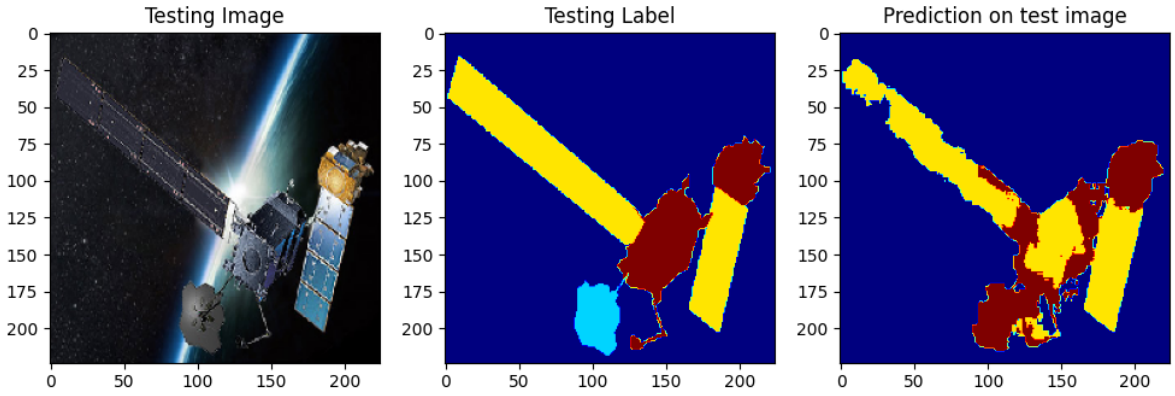

Figure 30: Prediction on test image

5.7 Experiment 7

Due to the good stabilization obtained from the experiment 6 between training and validation results and loss function, the network has been increased adding a third block in the encoding/decoding section in order to try to increase the obtained results. Along the new block, also the number of epochs used in the test increased to 300. As for the training set and dropout values, the same set and parameters from the previous experiment were kept. From 31 and 32

	mean IoU score	Loss
Training	76%	0.0924
Validation	59%	0.1720

Table 7: Experiment 7 report

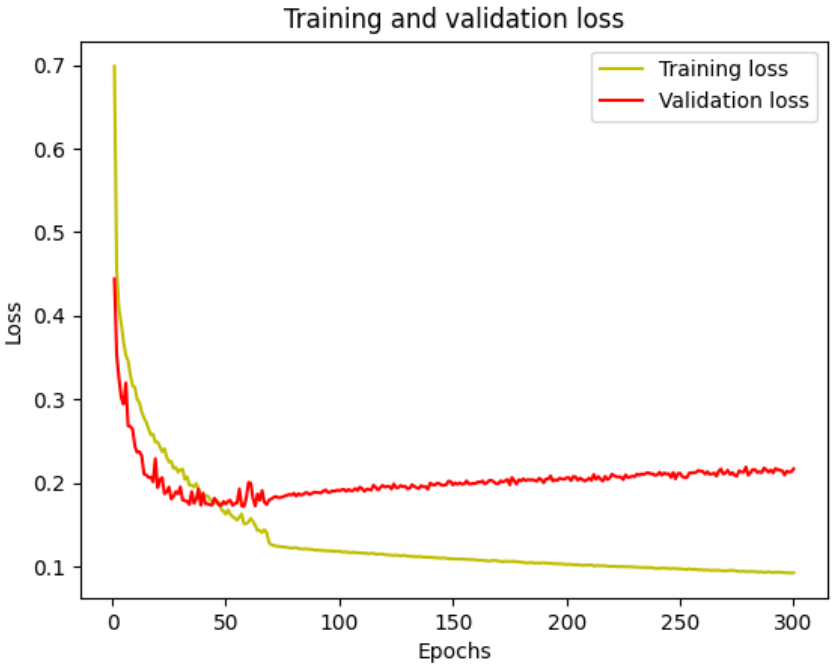

Figure 31: Training and Validation losses

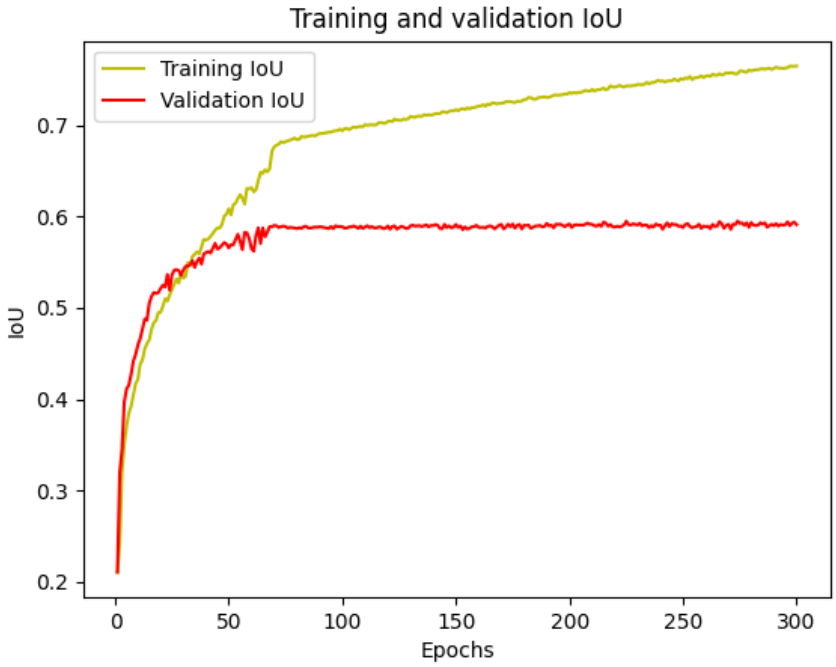

Figure 32: Training and Validation IoU metrics

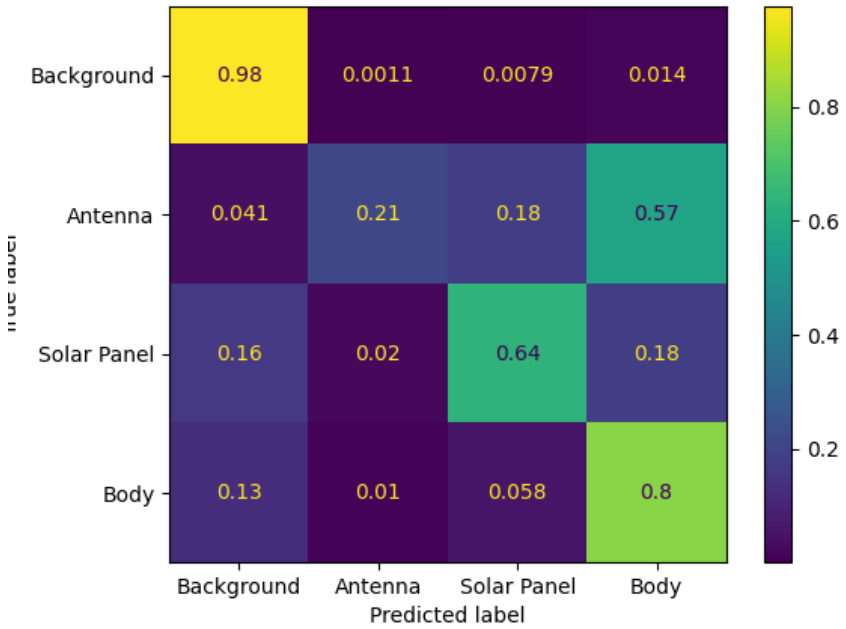

Figure 33: Confusion matrix

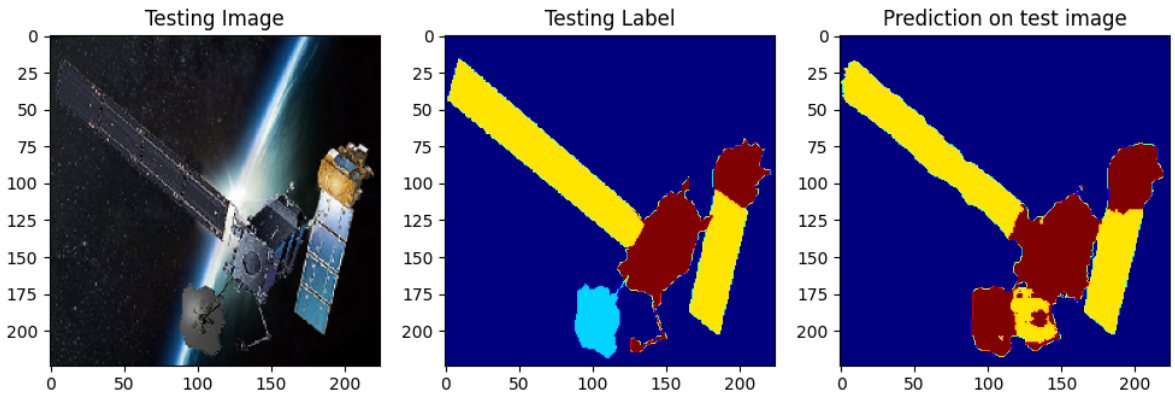

Figure 34: Prediction on test image

From 31 and 32 show a really big overfitting but, besided that, the values for the validation IoU increased from aorund 0.54 to nearly 0.60.

5.8 Experiment 8

In order to increase the validation and fight the overfitting, the training set has been espendanded by adding the second augmentation block to it and, doing so, increasing the dropout to 0.2 and 0.4 for the encoder/decoder blocks and the bridge respectively.

	mean IoU score	**Loss**
Training	65%	0.1472
Validation	59%	0.1610

Table 8: Experiment 8 report

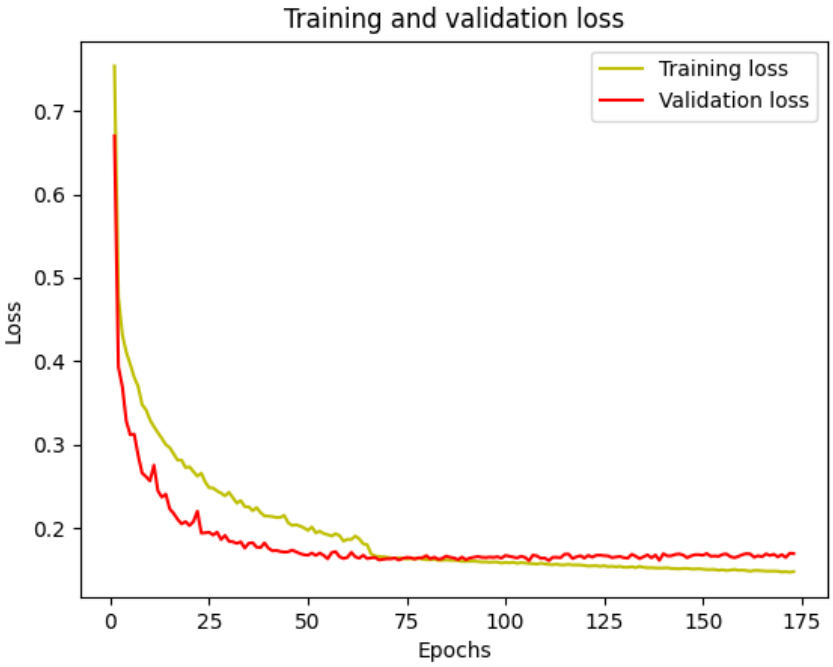

Figure 35: Training and Validation losses

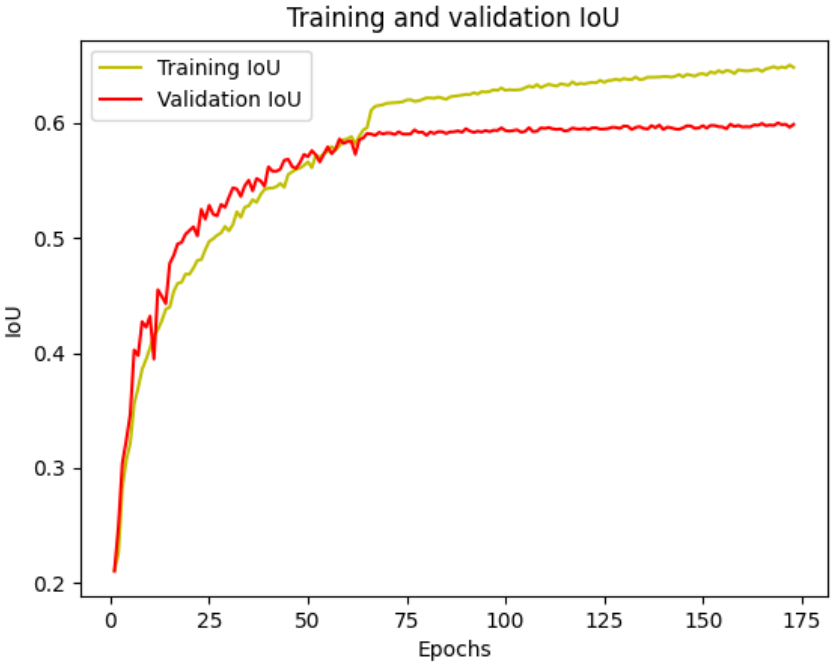

Figure 36: Training and Validation IoU metrics

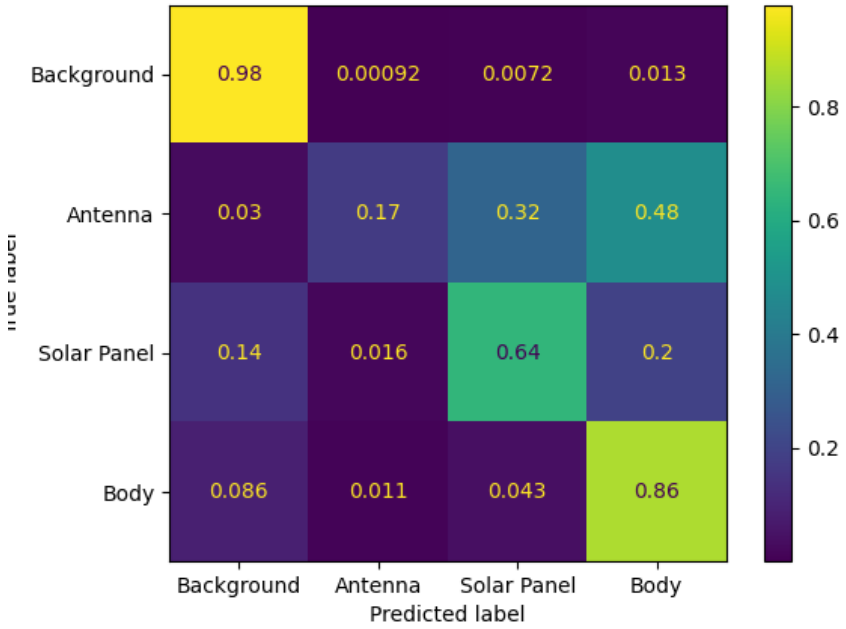

Figure 37: Confusion matrix

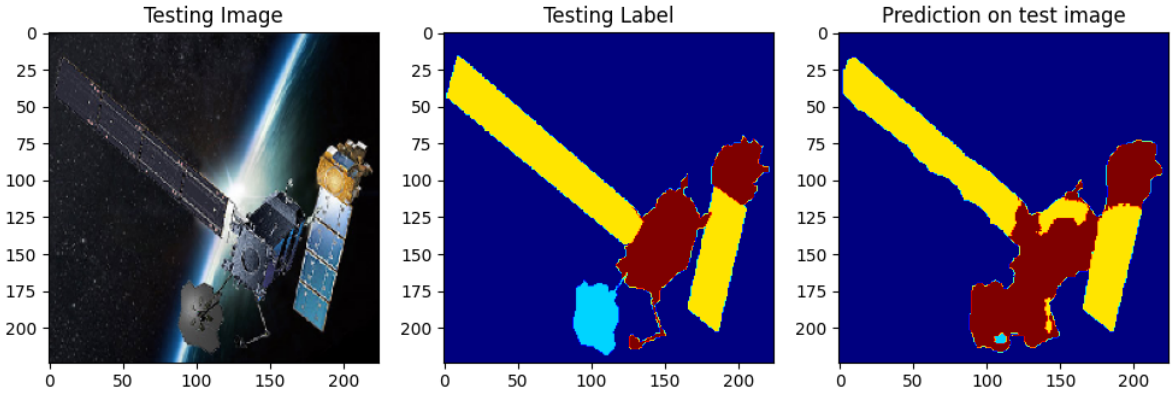

Figure 38: Prediction on test image

From 35 and 40 show a better performance with an increasing in the validation IoU but still with a considerable overfitting and a training IoU going up.

5.9 Experiment 9

Here the dropout has been increased to 0.3 and 0.5 for the encoding/deconding and bridge blocks respectively and with that, the size of the training set by adding the third augmentation block.

	mean IoU score	Loss
Training	61%	0.1639
Validation	60%	0.1630

Table 9: Experiment 9 report

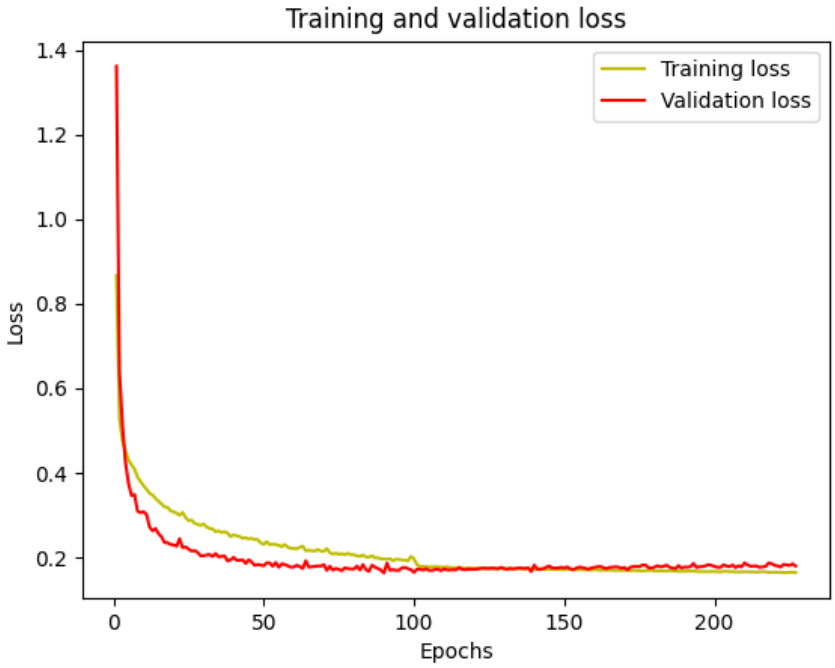

Figure 39: Training and Validation losses

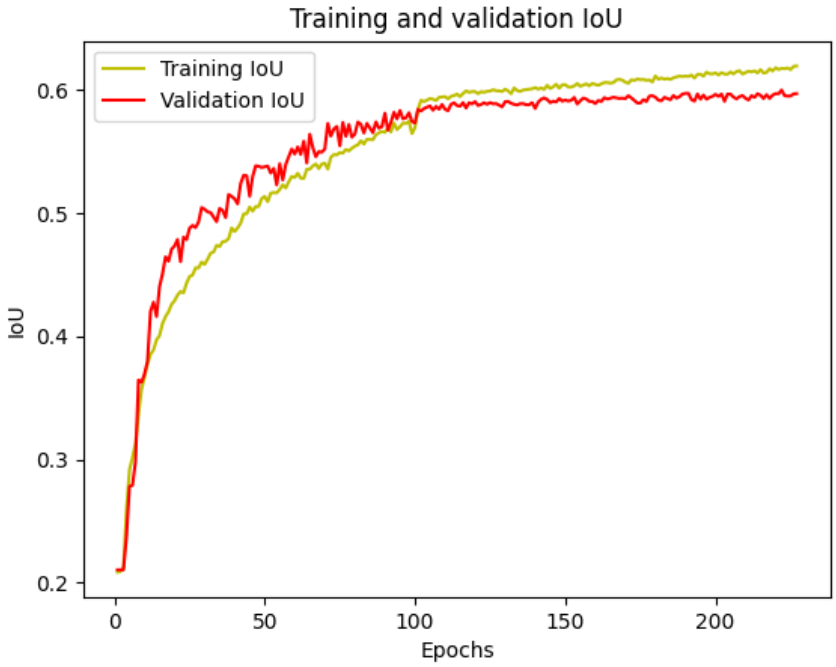

Figure 40: Training and Validation IoU metrics

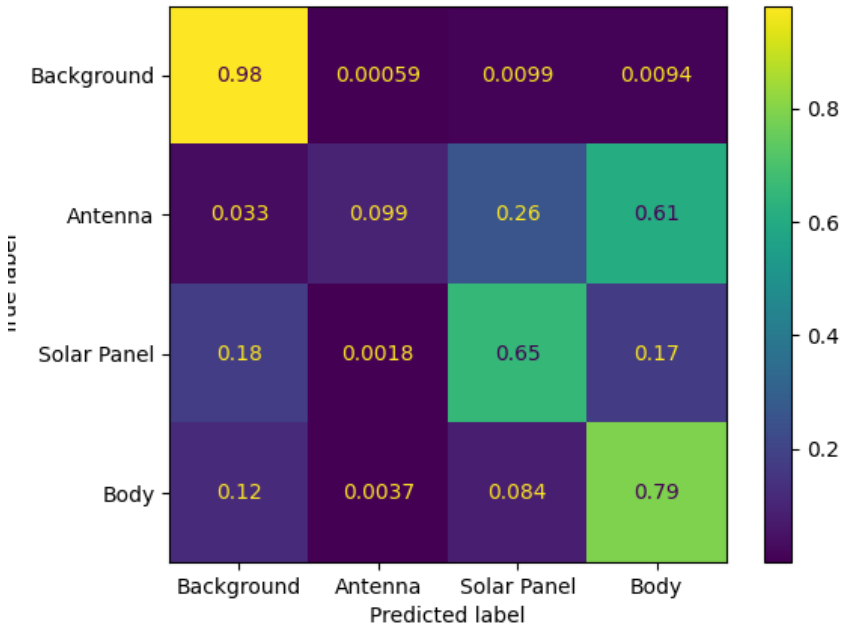

Figure 41: Confusion matrix

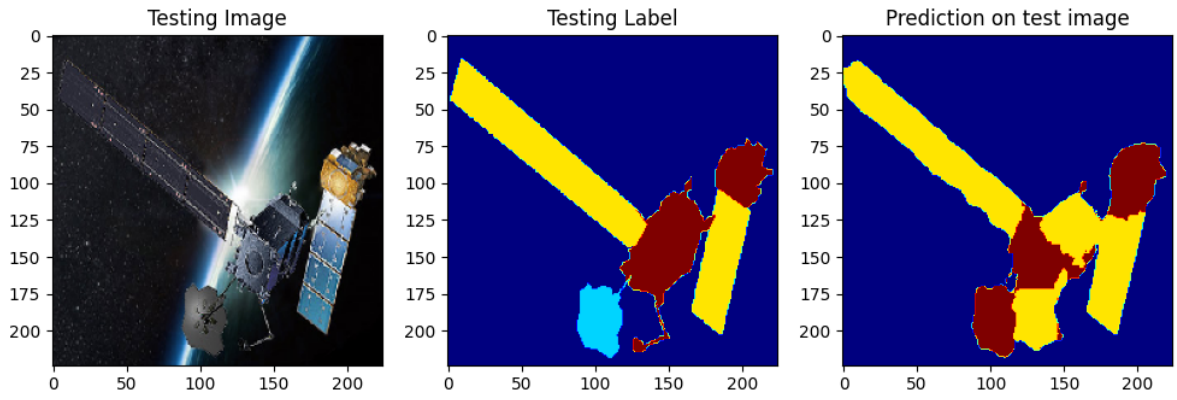

Figure 42: Prediction on test image

39 and 40 show a great improvement both in the loss and IoU function having training and validation almost with the same values for the loss function and a training and validation values higher than every past experiments. However, 40 also show a training function that keeps growing at faster speed than the IoU, which is still increasing, but slowly than the other one. This is one of the best results achiedeved and the best so far. It is also noticeable that 37 has better results in the classification then 41 but the much less overfitting and better loss reduction make this experiment a success from the previous one.

5.10 Experiment 10

Due to the good results achieved by the 9th experiment new tests have been conducted increasing the network to 4 blocks for the encoding/decoding part. Along with that a new dropout layer for additional dropout was inserted after each second convolution in the two parts of the net except for the bridge which kept only one dropout layer. The values for the dropout used were 0.3 and 0.4 for the ones in the encoder/decoder sides and 0.5 for the bridge one.

	mean IoU score	**Loss**
Training	69%	0.1202
Validation	61%	0.1503

Table 10: Experiment 10 report

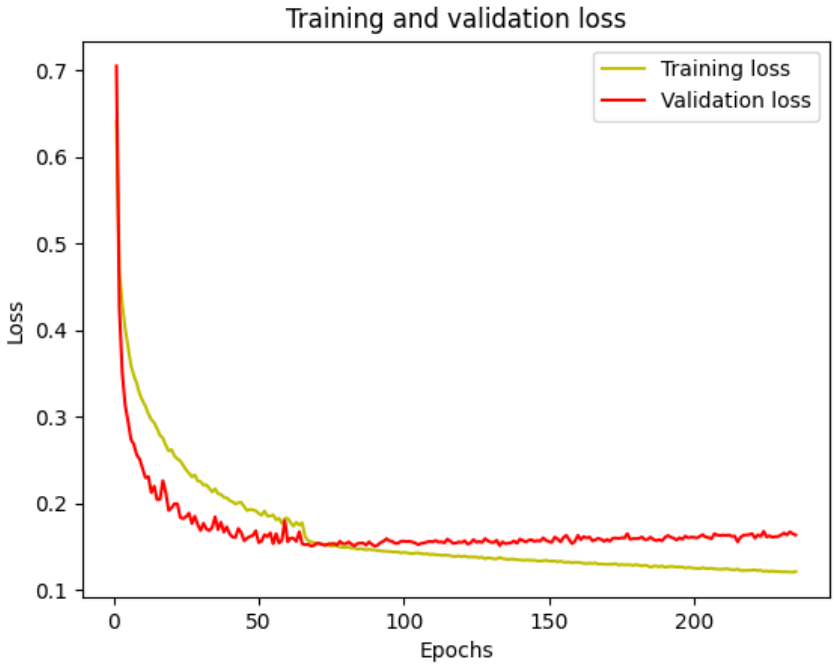

Figure 43: Training and Validation losses

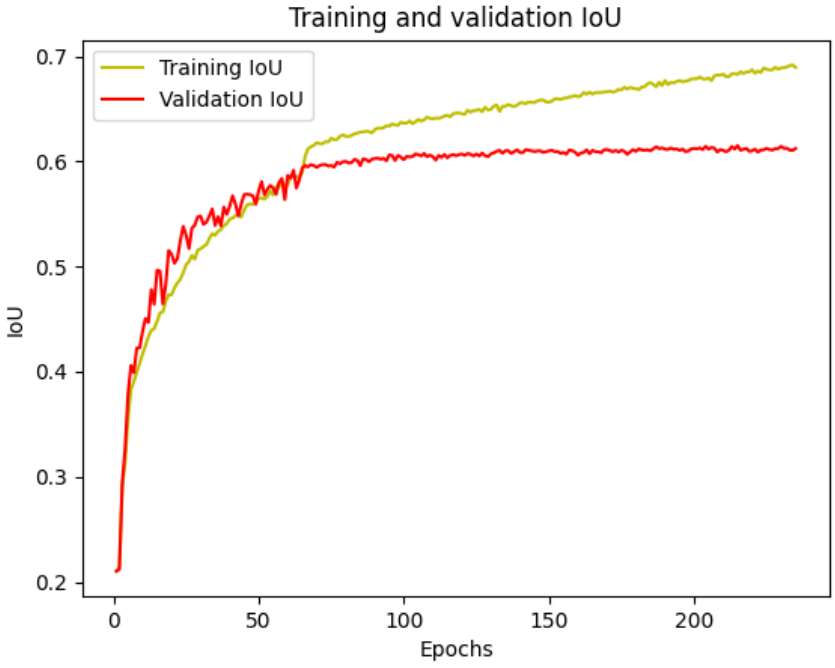

Figure 44: Training and Validation IoU metrics

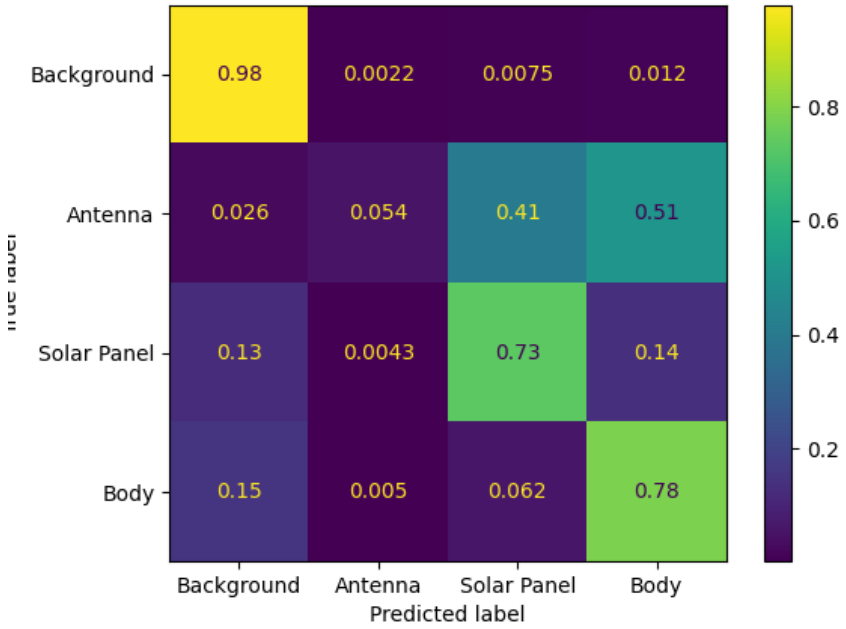

Figure 45: Confusion matrix

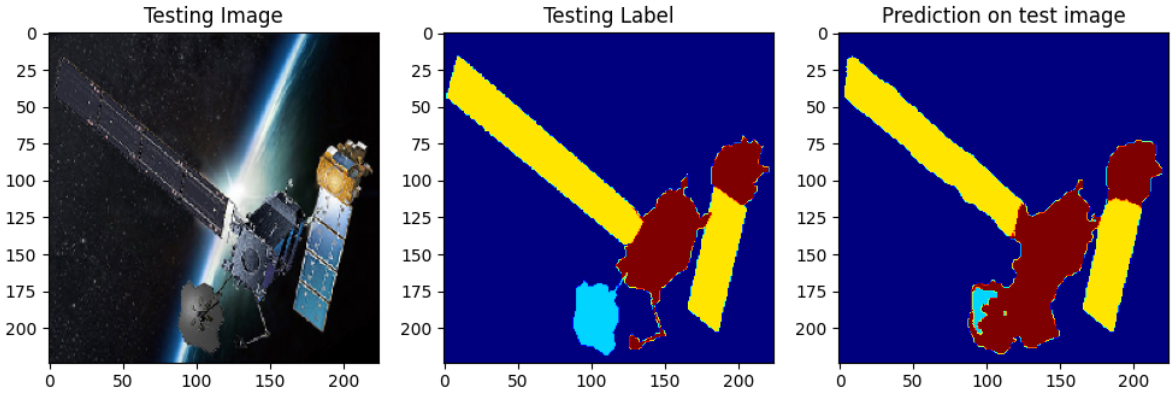

Figure 46: Prediction on test image

43 show a continuous reduction for the training loss but the validation one, after an initial decrease, started to increase again showing the presence of overfitting that can be strongly noticeable in 44. However it is also mentionable that the previous function and the confusion matrix 45 show an increasing in the classification ability of the net.

5.11 Experiment 11-12-13

In order to fight the overfitting had with the net in the experiment 10 three different experiments were conducted: 11) the dropout was increased to 0.5 in all the dropout layers. 12) batch normalization has been applied keeping the dropout of 11. 13) l2 regularization was applied inside the maxpool layers keeping the dropout layers with the

same values as 11 and 12.

	mean IoU score	Loss
Training	44%	0.2679
Validation	29%	0.3457

Table 11: Experiment 11 report

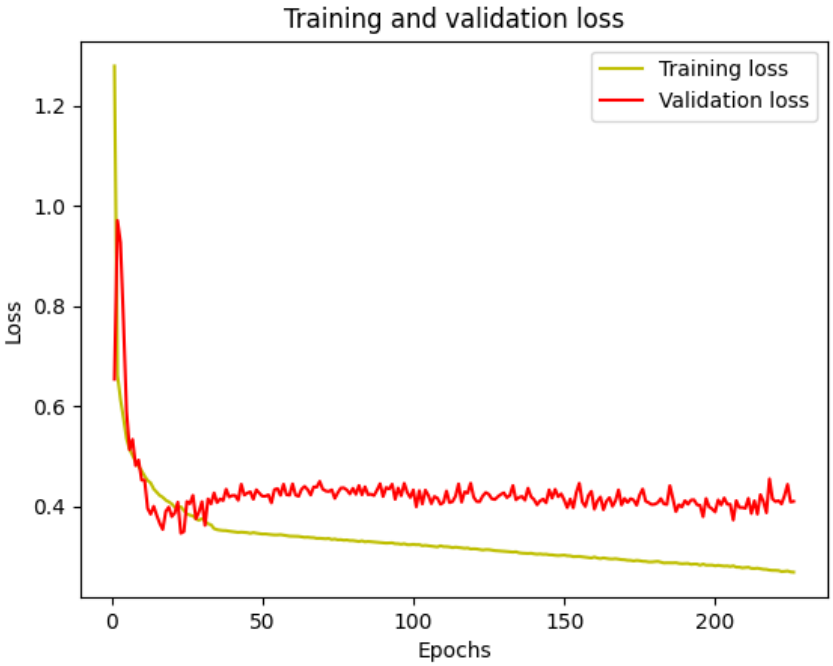

Figure 47: Training and Validation losses

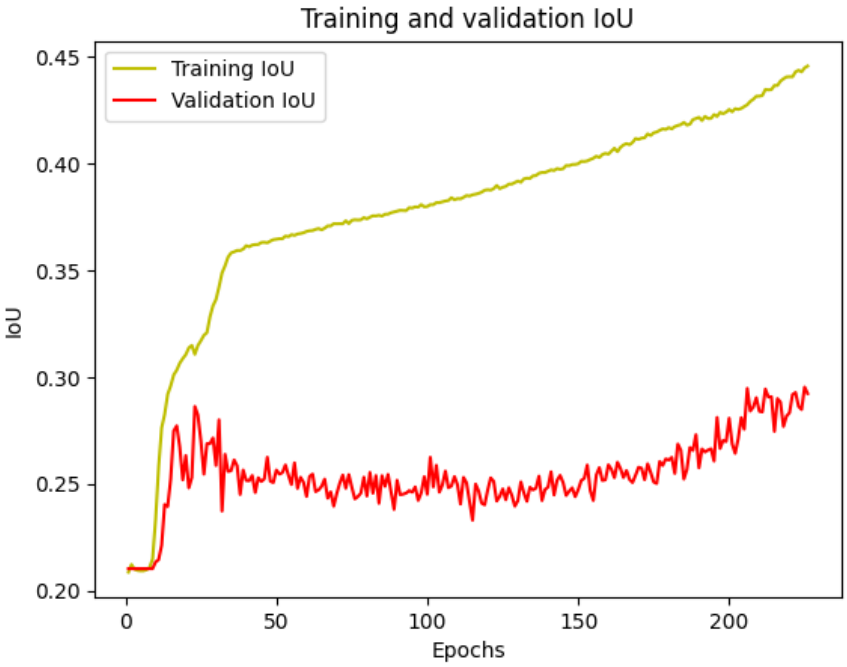

Figure 48: Training and Validation IoU metrics

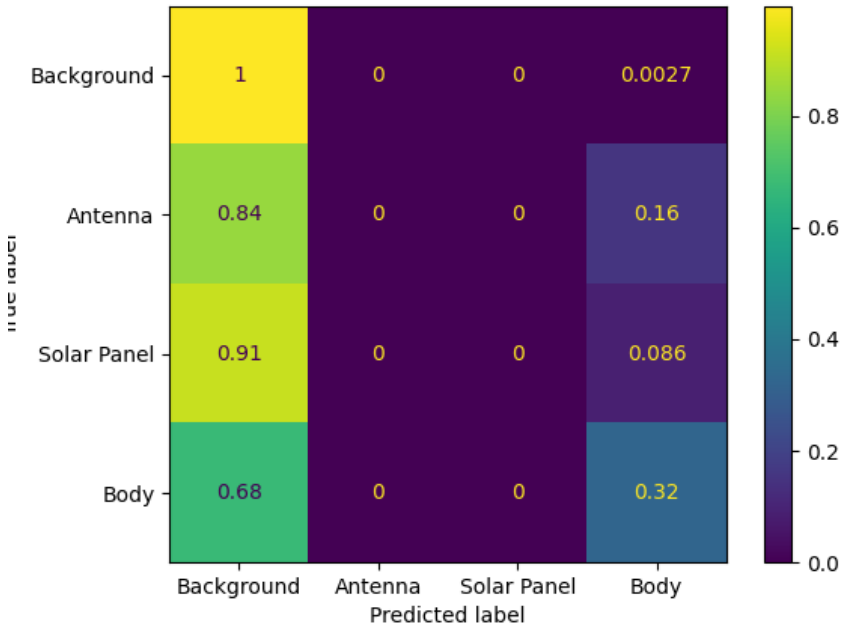

Figure 49: Confusion matrix

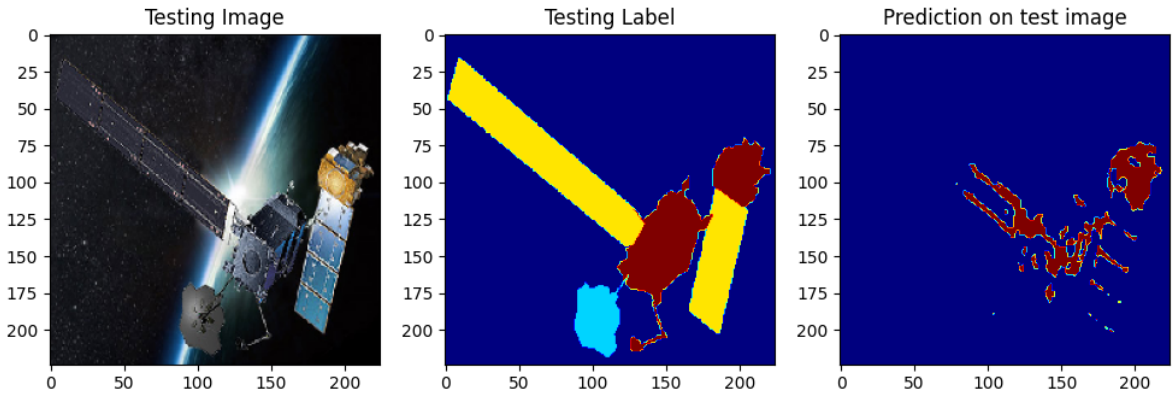

Figure 50: Prediction on test image

The results reported are very similar for each of the three experiments. As it easy to see none of them improved the network returning only results that were much worst than any other experiment.

5.12 Experiment 14

Here an increasing of the training set was tested by adding a fourth augmentation block but, this time, the augmentation was performed only on the images containing red and yellow classes respectively. The dropout values were setted back to the values used in the experiment 10.

	mean IoU score	**Loss**
Training	58%	0.1514
Validation	55%	0.1635

Table 12: Experiment 14 report

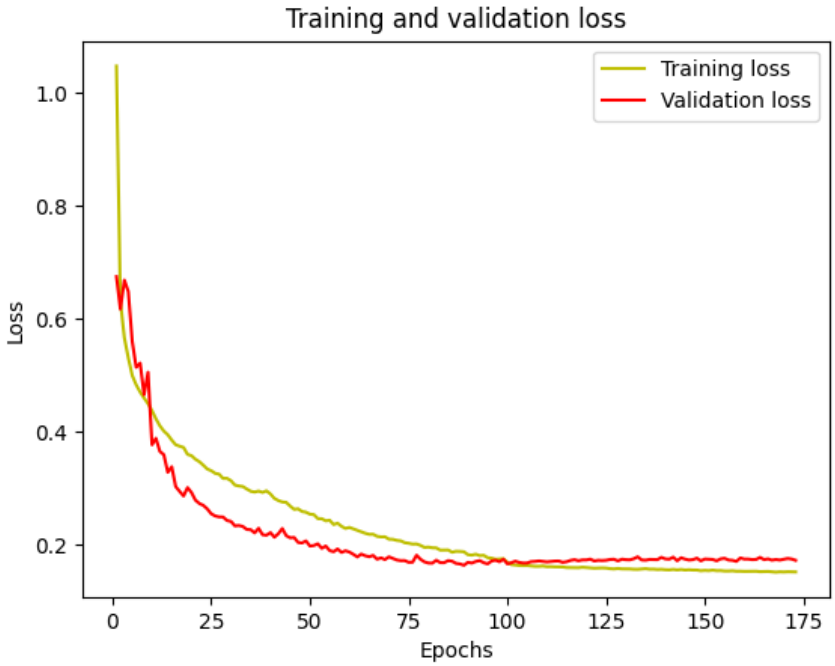

Figure 51: Training and Validation losses

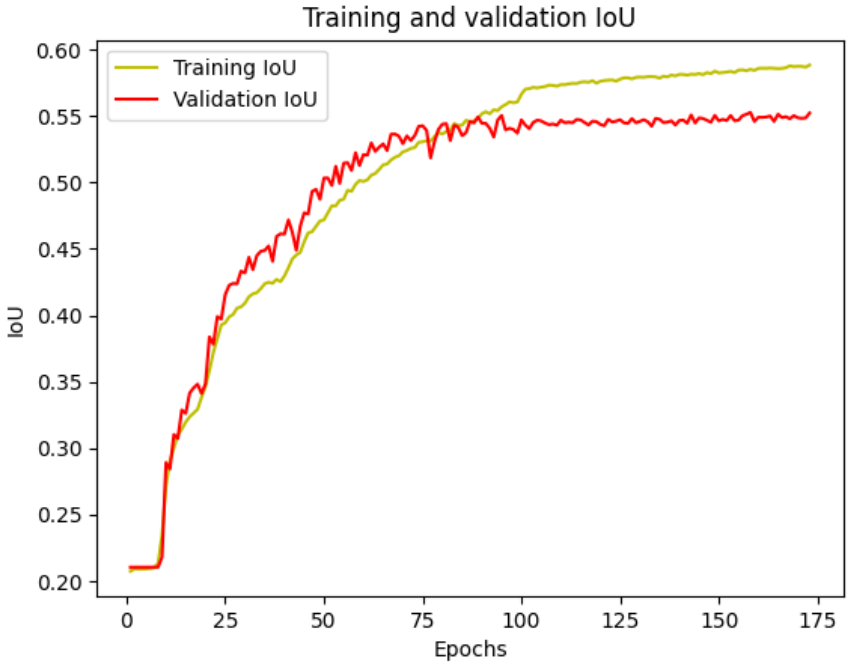

Figure 52: Training and Validation IoU metrics

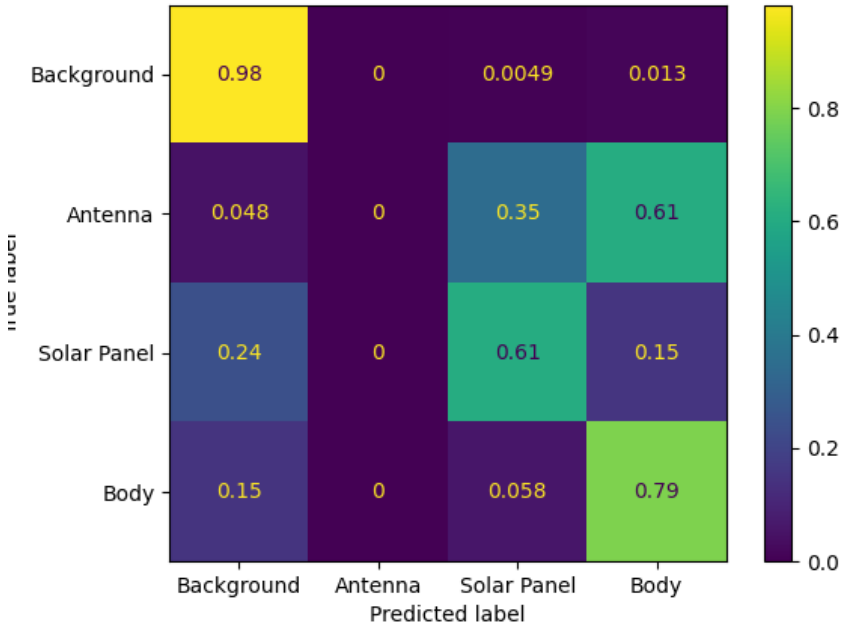

Figure 53: Confusion matrix

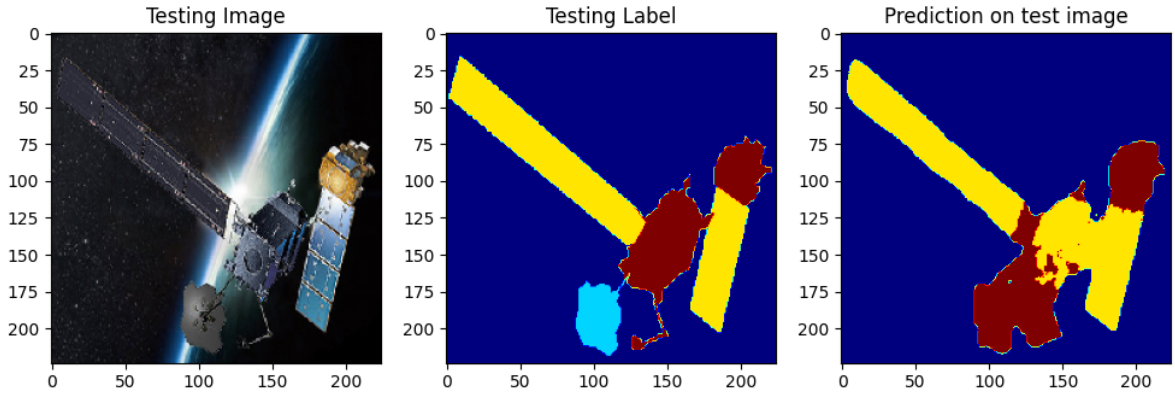

Figure 54: Prediction on test image

The results show a good loss in 51 but overfitting in 52 and decreased classification permormances as shown in 53 with a really low Antenna classification.

5.13 Experiment 15

Due to the too high training set and the too large difference from elements with red and yellow classes and the blue one, the augmentation blocks used were brought back to 3. Along with that, Nadam optimizer was added in order to improve the results of the network.

	mean IoU score	Loss
Training	70%	0.1173
Validation	60%	0.1625

Table 13: Experiment 15 report

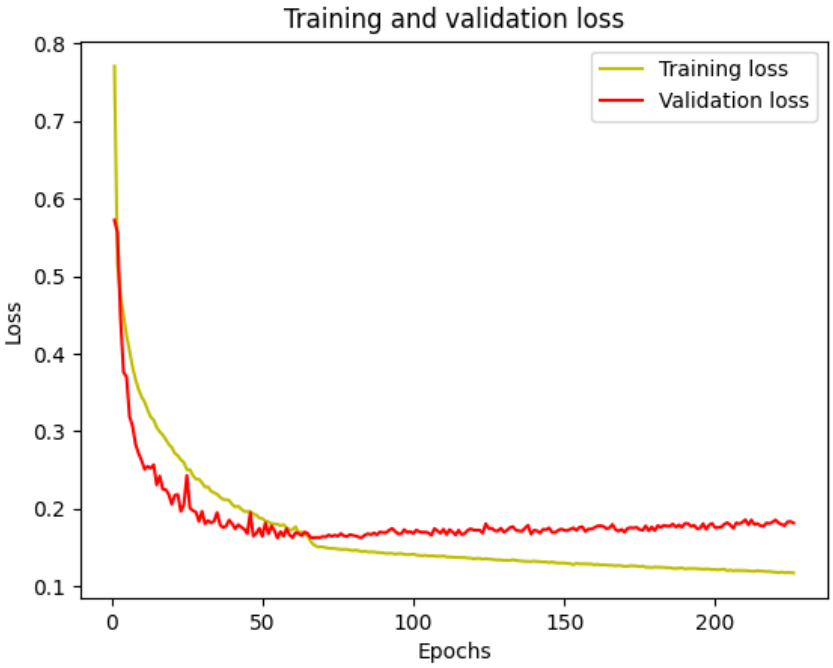

Figure 55: Training and Validation losses

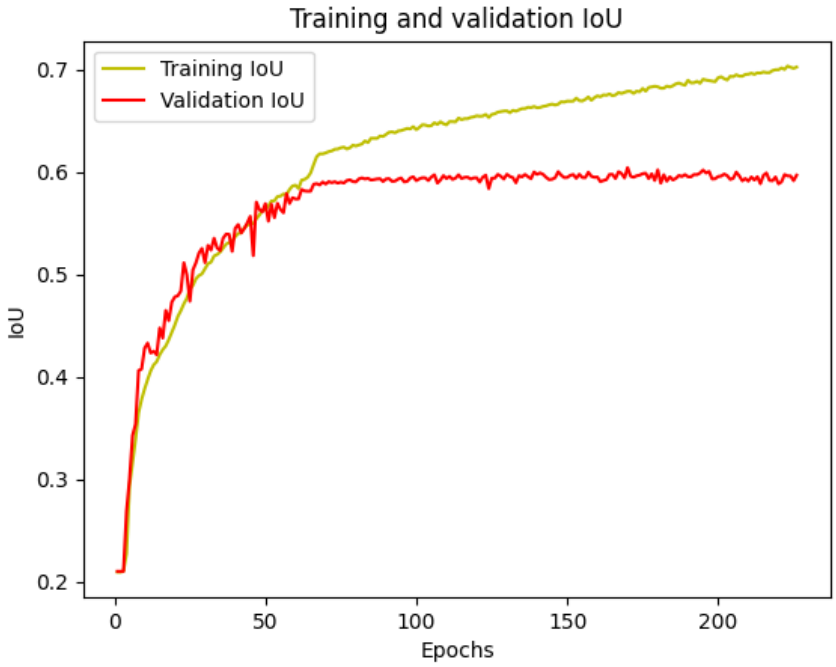

Figure 56: Training and Validation IoU metrics

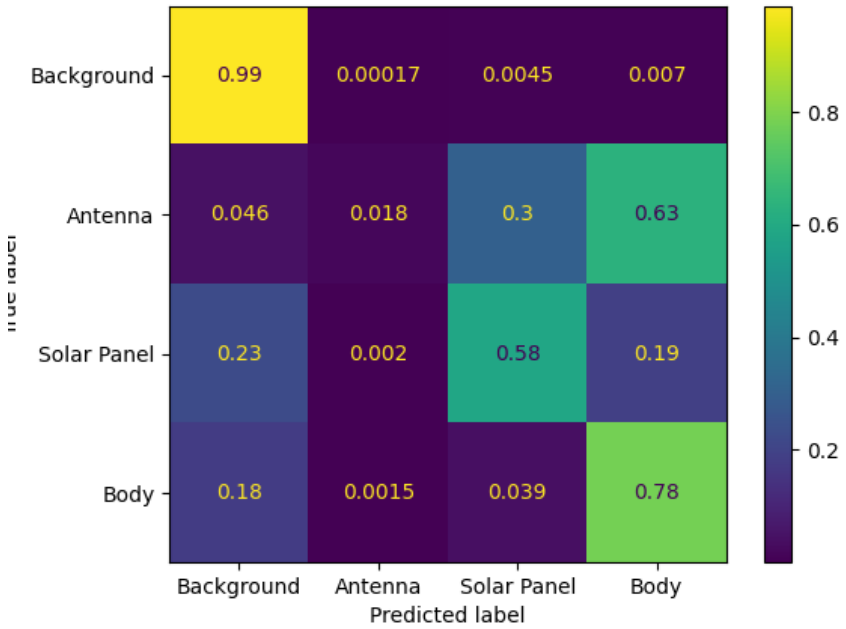

Figure 57: Confusion matrix

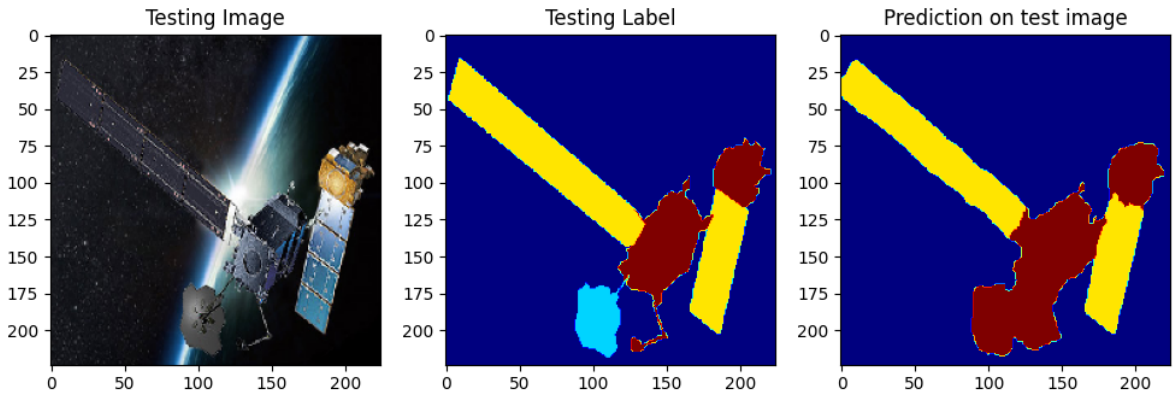

Figure 58: Prediction on test image

By comparing the results obtained in 55 and 56 with the ones of the experiment 10 can be seen that not a great improvement has been achieved even with Nadam.

5.14 Experiment 16

As next experiment a Focal Loss Function was used instead of the cross-entropy in order to improve the misclassified classes and, along with that, Nadam optimizer was replaced by Adam.

	mean IoU score	**Loss**
Training	62%	0.0152
Validation	58%	0.0182

Table 14: Experiment 16 report

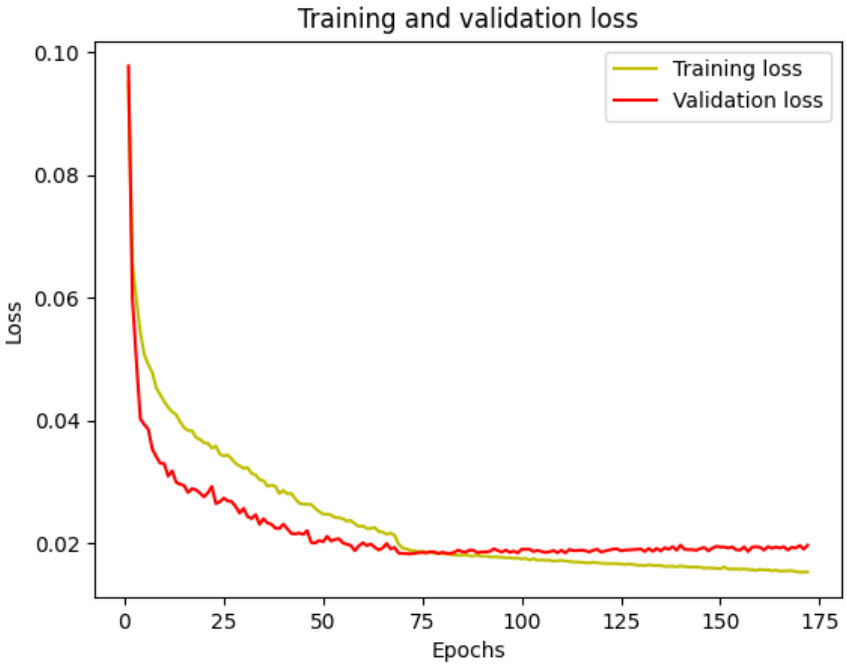

Figure 59: Training and Validation losses

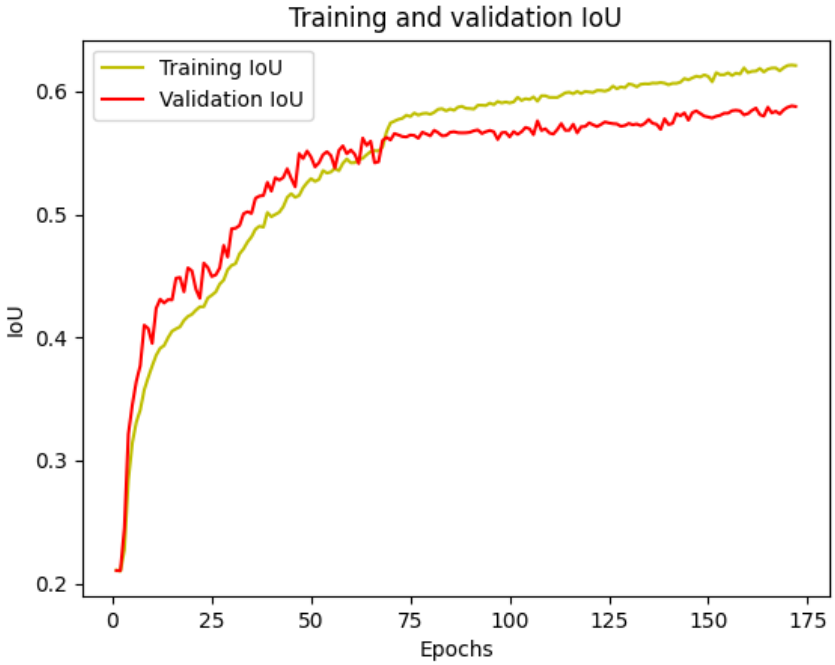

Figure 60: Training and Validation IoU metrics

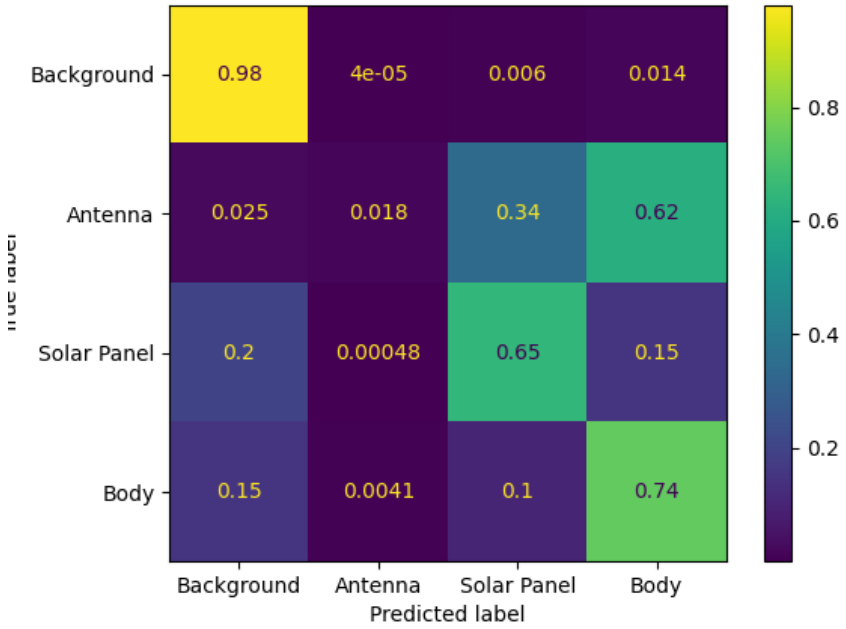

Figure 61: Confusion matrix

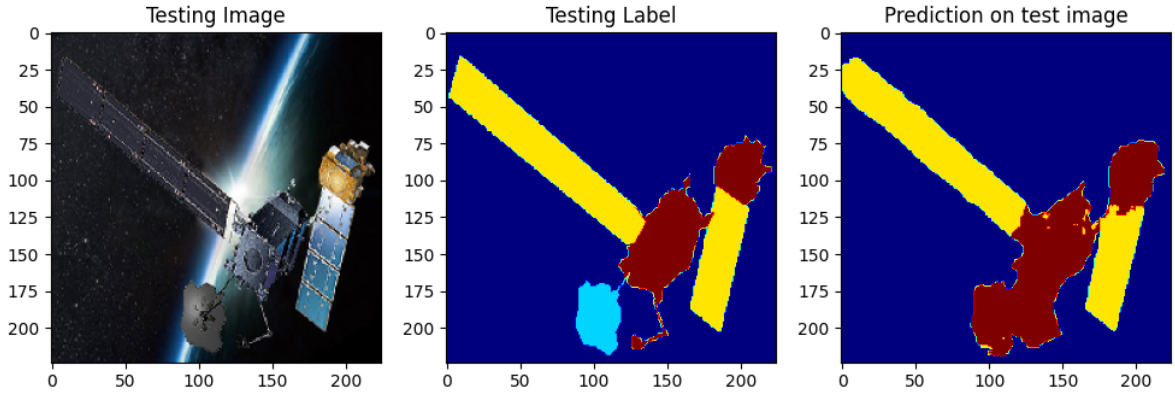

Figure 62: Prediction on test image

59 and 60 show a big improvement compared to the results of the previous experiment. Along with that the confusion matrix 61 reports an higher classification for the solar panel and the body but a decreased value for the antenna.

5.15 Experiment 17

In this experiment the class weights were calculated and inserted in the fit in order to classify the antenna class.

	mean IoU score	Loss
Training	58%	0.0184
Validation	48%	0.3299

Table 15: Experiment 17 report

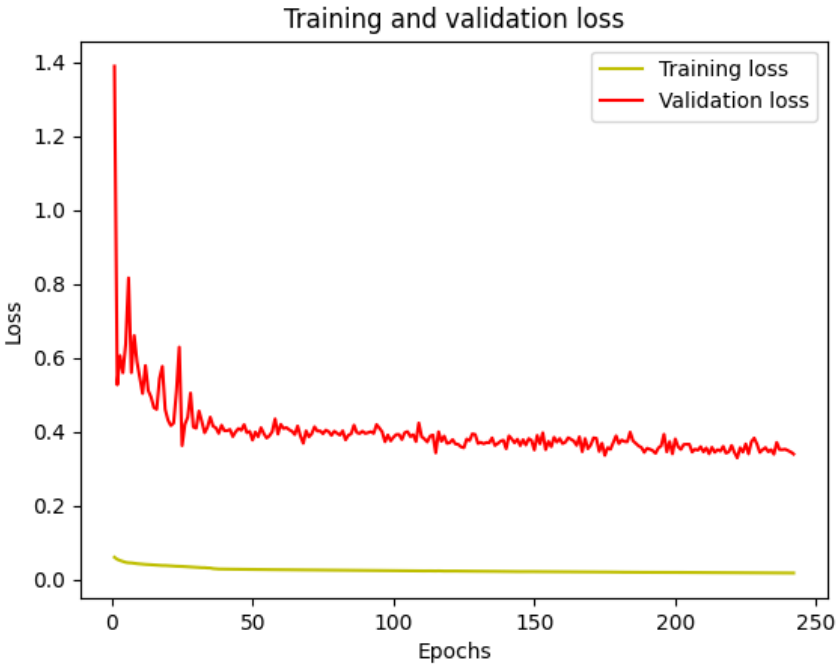

Figure 63: Training and Validation losses

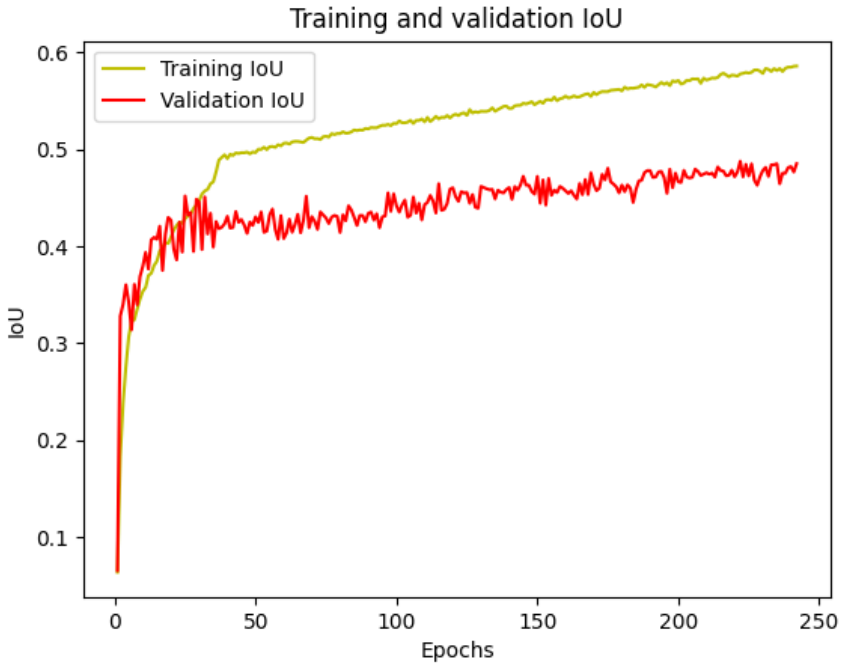

Figure 64: Training and Validation IoU metrics

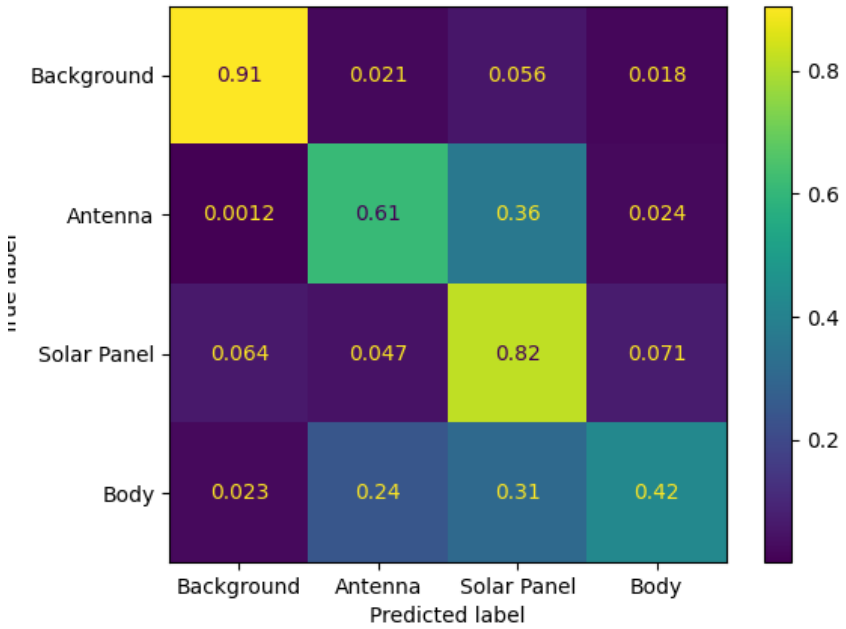

Figure 65: Confusion matrix

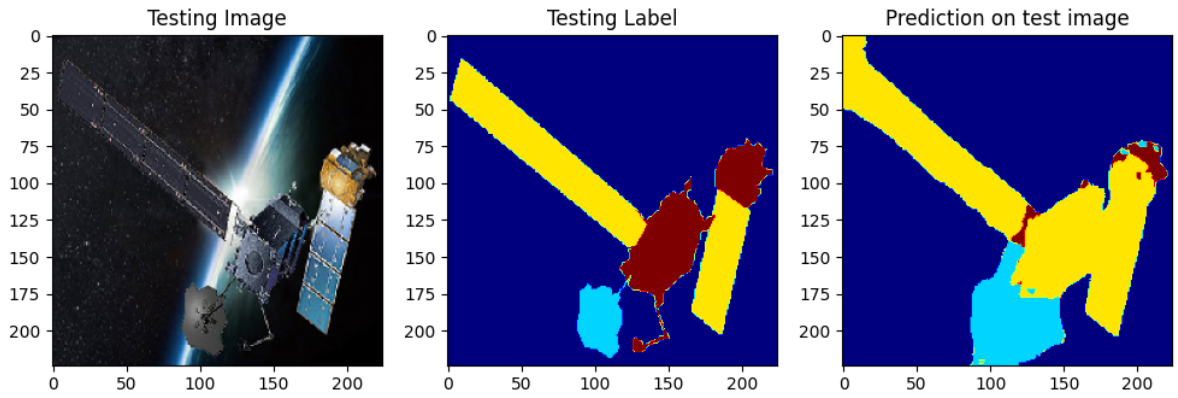

Figure 66: Prediction on test image

From 63 64 can been seen that despite the way better classification of the blue class, as reported in the confusion matrix 65, loss functions and the values of validation and training had not good results.

6 ResNet34 - U-Net

As first pre-trained network used it has been decided to use ResNet34 due to fact that, not only is one of the most used net for this kind of problems but also because, it has good performance on small datasets and, with it, residual connections could also be used a performance experiment. In the following experiments common parameters were used such as, filters for convolutional layers setted to 16 up to 256, Adam as optimizer and categorical cross-entropy as loss function. The backbone was imported (as for the next section) from the segmentation_models library [7].

6.1 Experiment 1

With encoder freezed the first experiment took the 3 augmentation blocks and trained the model.

	mean IoU score	**Loss**
Training	87%	0.0505
Validation	70%	0.1131

Table 16: Experiment 1 report

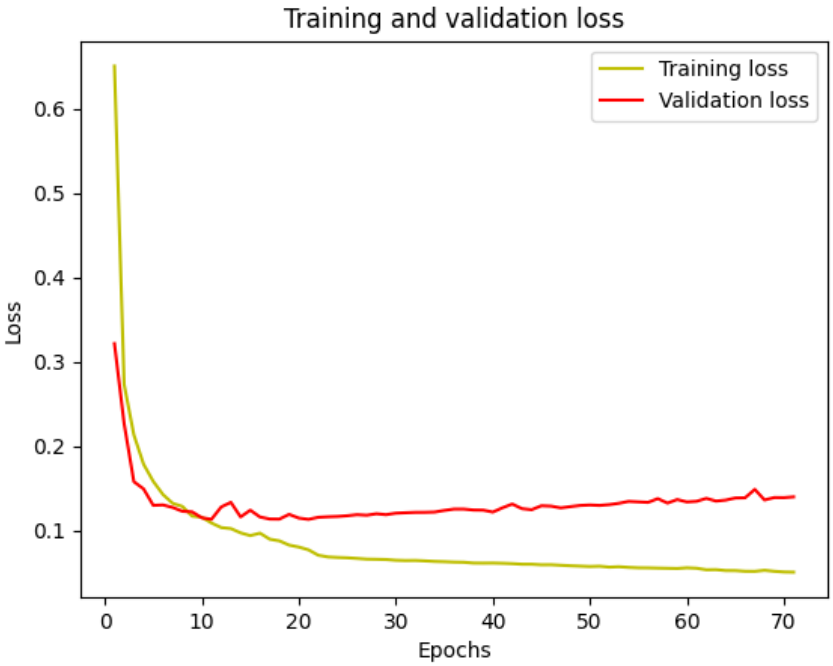

Figure 67: Training and Validation losses

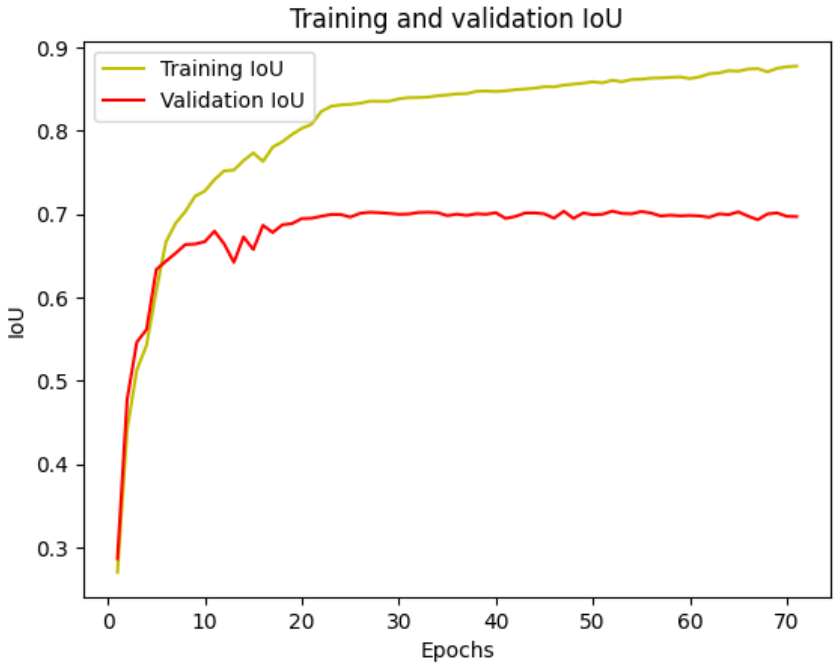

Figure 68: Training and Validation IoU metrics

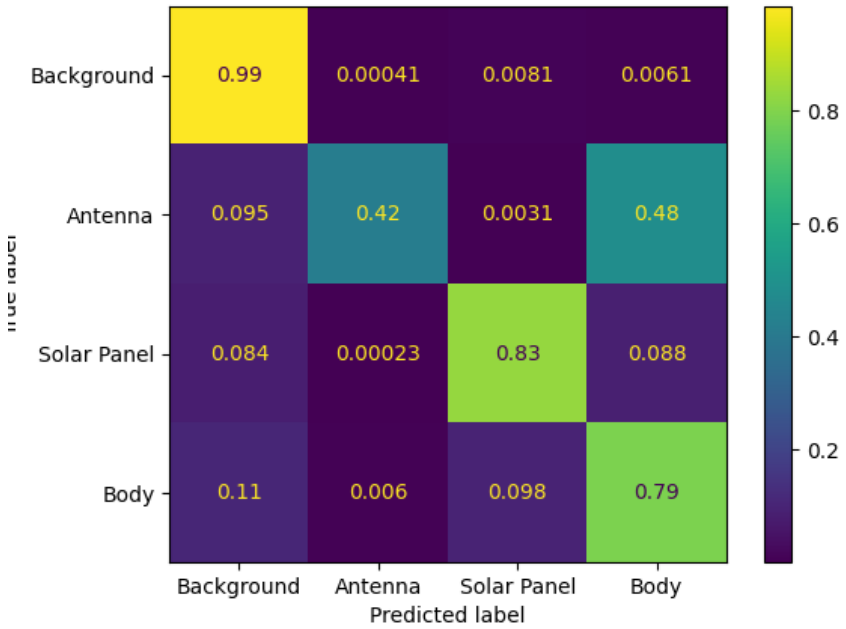

Figure 69: Confusion matrix

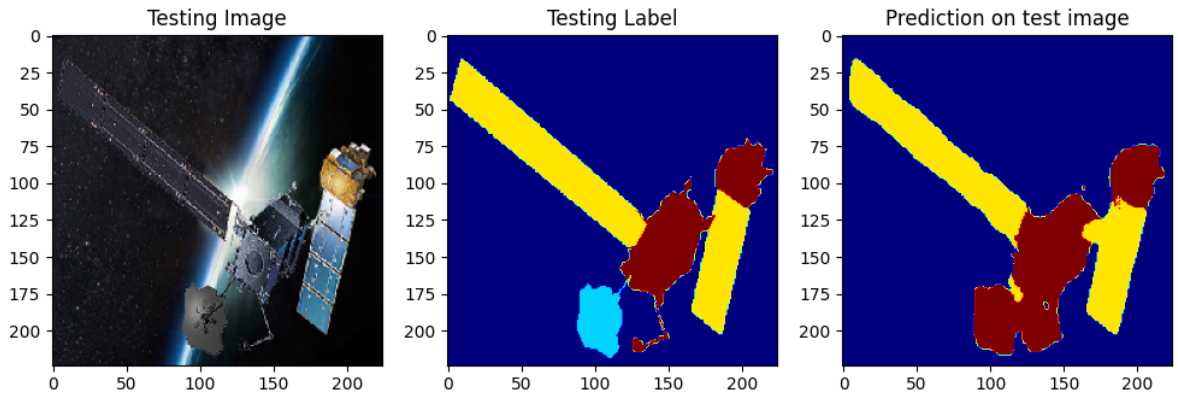

Figure 70: Prediction on test image

It can be seen from 67 and 72 there is a big overfitting problem and the validation loss started to grow again after a first decrease in the early epochs, this is due to the fact that here no dropoutlayer had been inserted. Despite that the values obtained from the confusion matrix are much higher than before thanks to the pretrained net.

6.2 Experiment 2

In order to decrease the overfitting a dropout layer was added after each convolution layer in the decoder part of the network,with value setted to 0.3.

	mean IoU score	**Loss**
Training	83%	0.0693
Validation	69%	0.1117

Table 17: Experiment 2 report

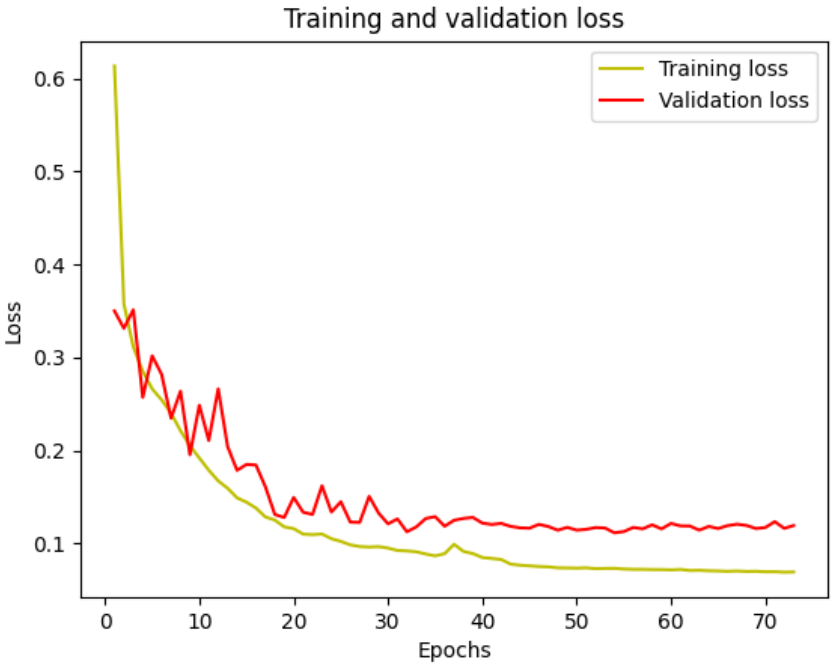

Figure 71: Training and Validation losses

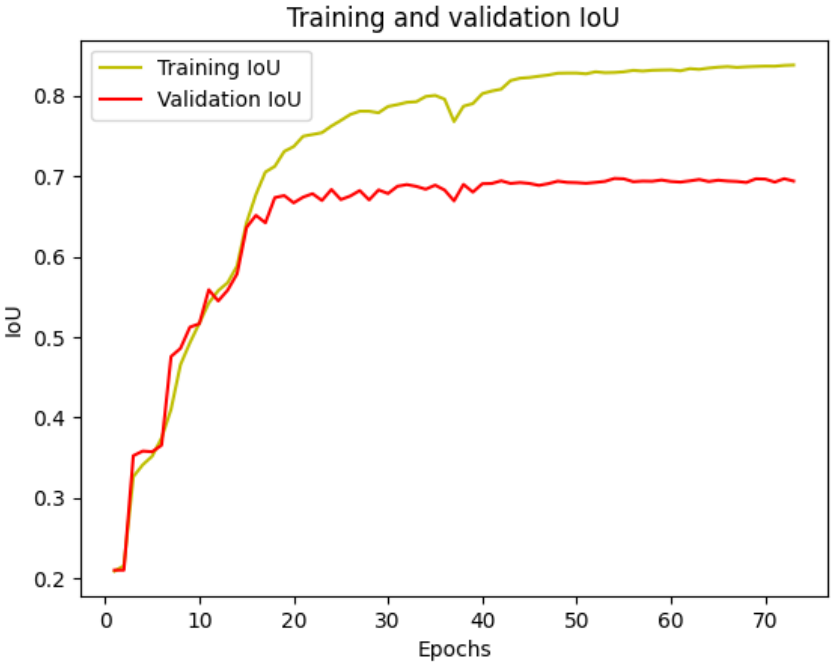

Figure 72: Training and Validation IoU metrics

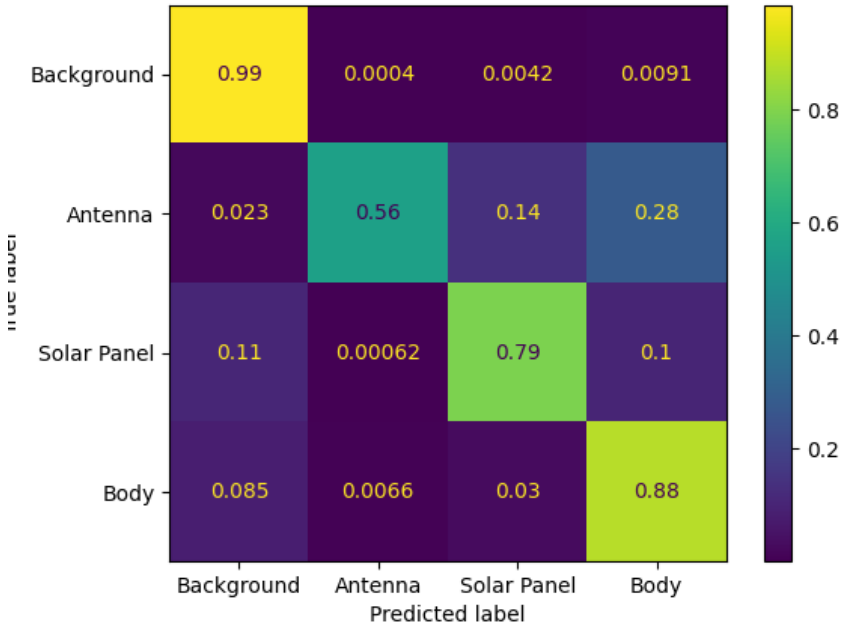

Figure 73: Confusion matrix

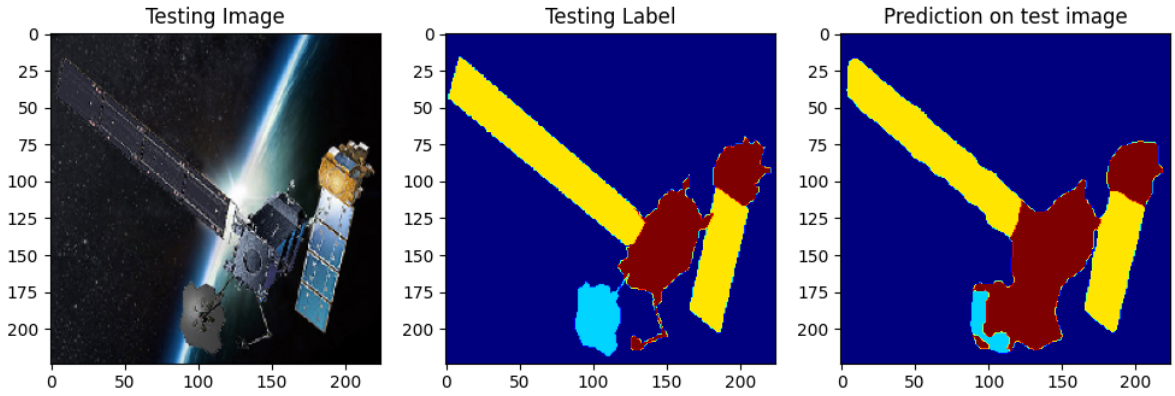

Figure 74: Prediction on test image

From 71 and 72 can be seen that the dropout helped to overcome the previous problems, having now a validation and training loss curves following a similar trend but, as for the IoU score, having still an evident overfitting. Considering the dropout values, another experiment was conducted increasing the value of the extra dropout layer from 0.3 to 0.5 but the results were worst than the current one so it was not taken in consideration for the report.

6.3 Experiment 3

Keeping the same net from the previous experiment, a batch normalization layer was added in between every convolution layer and it's following activation function.

	mean IoU score	Loss
Training	94%	0.0231
Validation	72%	0.1196

Table 18: Experiment 3 report

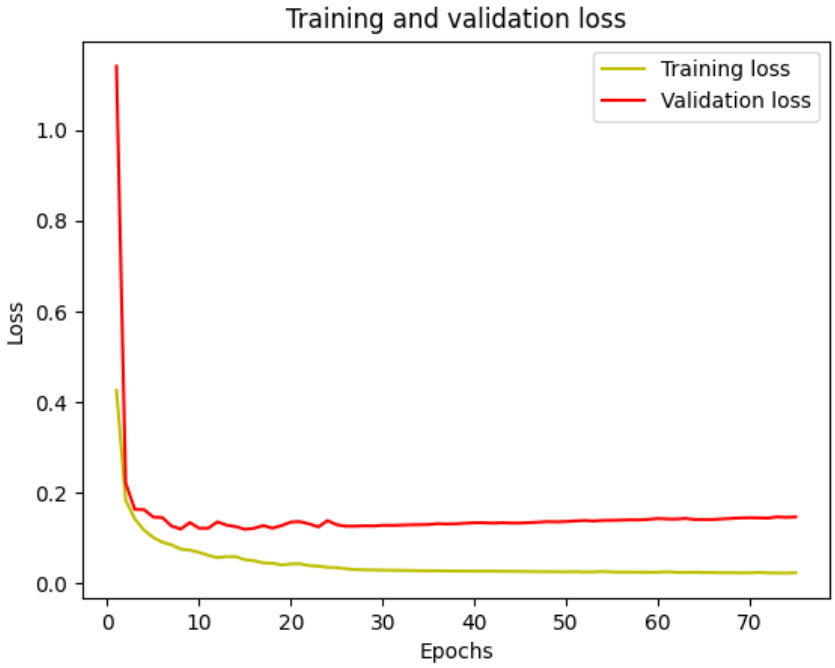

Figure 75: Training and Validation losses

Figure 76: Training and Validation IoU metrics

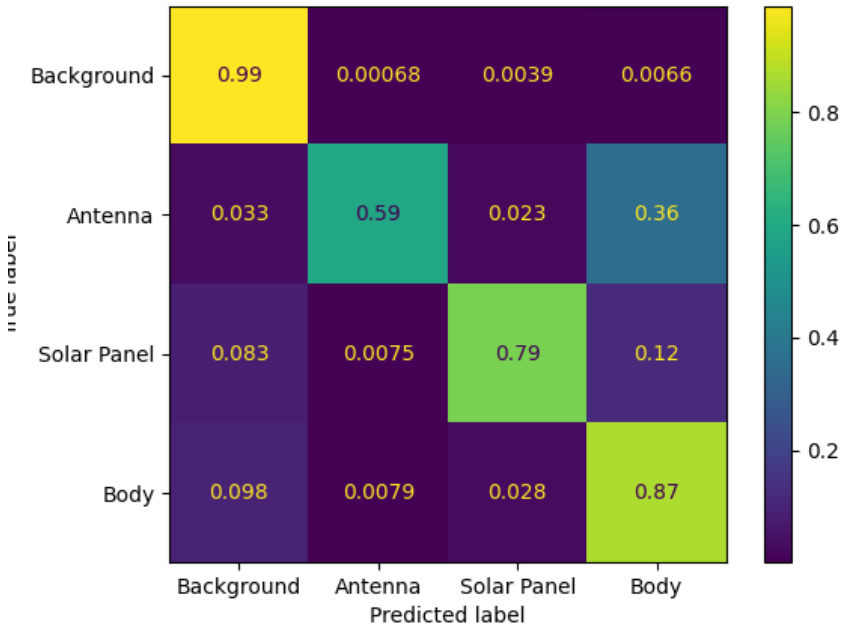

Figure 77: Confusion matrix

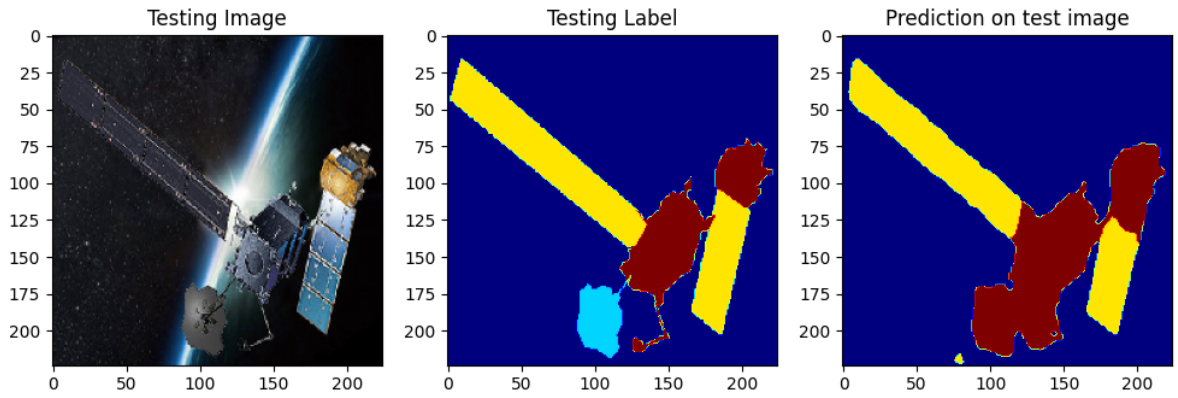

Figure 78: Prediction on test image

The results obtained were not so good if compared to the previous experiment. The batch normalization helped to have smoother loss curves as reported in 75 but the variance between training and validation IoU scores in 76 shows a huge increase of the overfitting. This is also because in this experiment no dropout layer was considered, so they had to be added.

6.4 Experiment 4

From the net of experiment 3 the dropout layers were added in the decoder part as for experiment 2.

	mean IoU score	**Loss**
Training	90%	0.0407
Validation	72%	0.1176

Table 19: Experiment 4 report

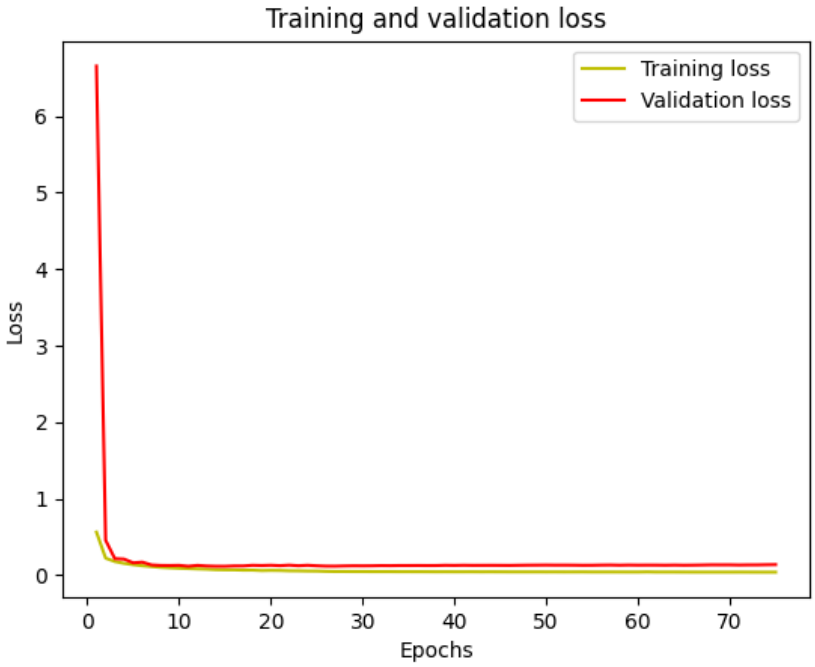

Figure 79: Training and Validation losses

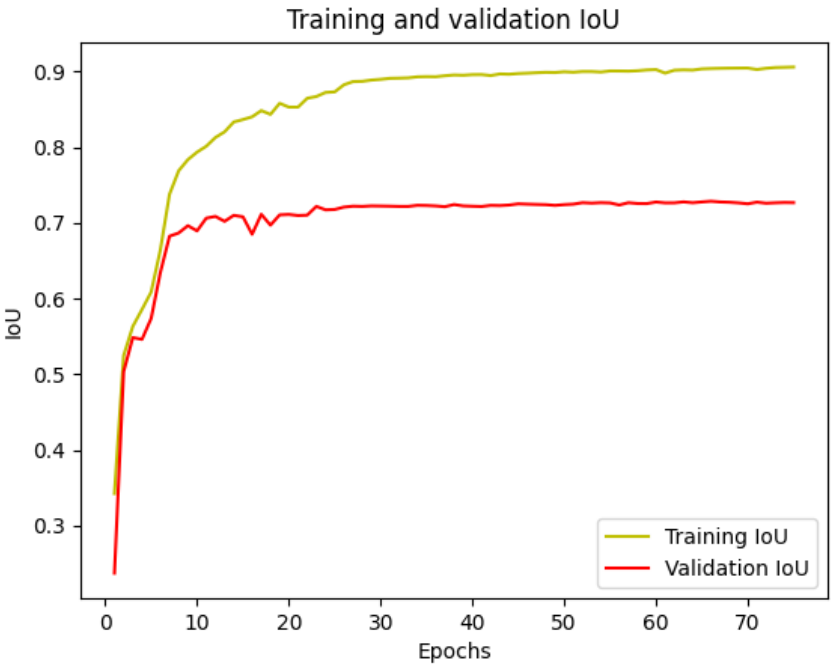

Figure 80: Training and Validation IoU metrics

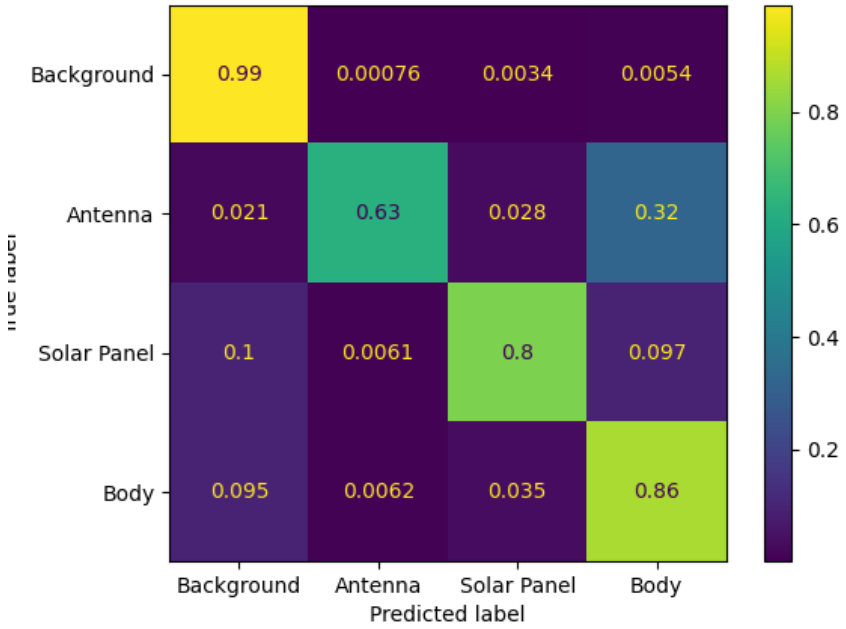

Figure 81: Confusion matrix

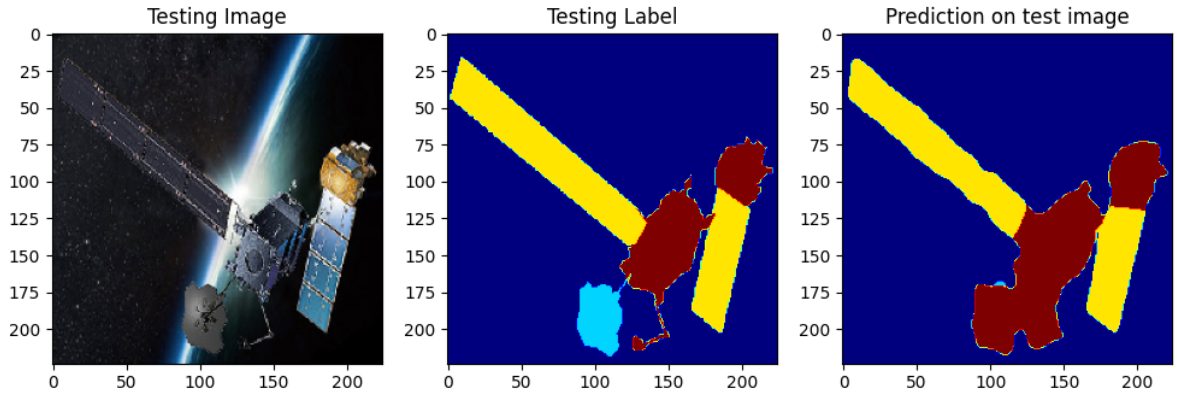

Figure 82: Prediction on test image

As can been seen from 79 and 80 the dropout layers helped significatively to reduce overfitting. The loss curves are almost identical following the same trend as for the IoU scores in which the difference betweeb training and validation reduced even if no so significantly.

6.5 Experiment 5

Starting from the previous net a regularization technique was applied in order to penalize the heavy weights by adding a penalty to the loss function. This method had the objective to overcome to the overfitting problem.

	mean IoU score	Loss
Training	90%	0.7748
Validation	73%	0.8569

Table 20: Experiment 5 report

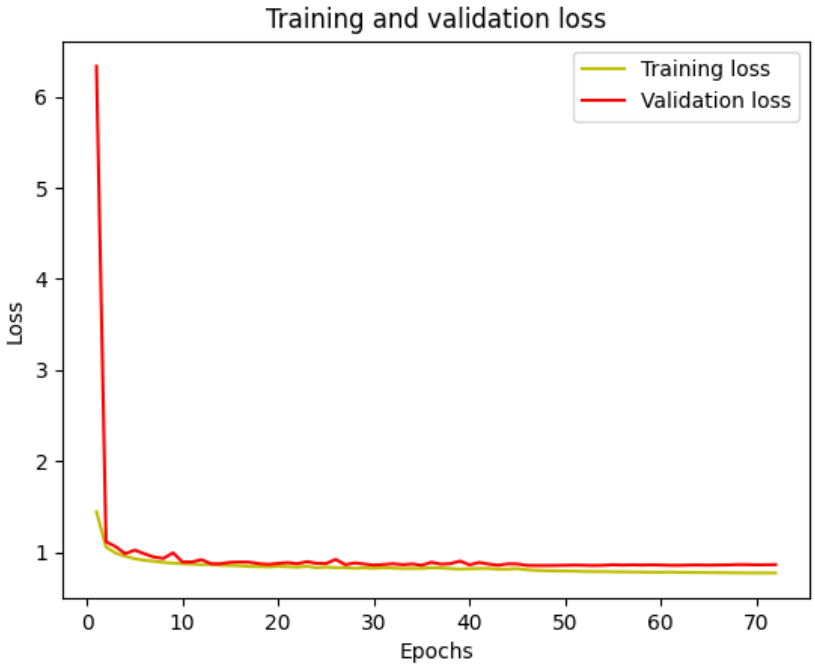

Figure 83: Training and Validation losses

Figure 84: Training and Validation IoU metrics

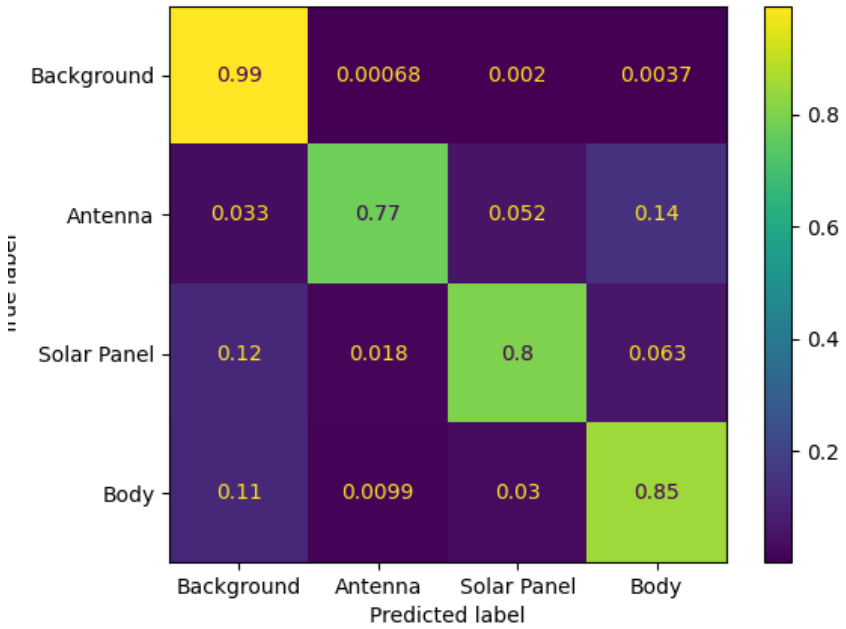

Figure 85: Confusion matrix

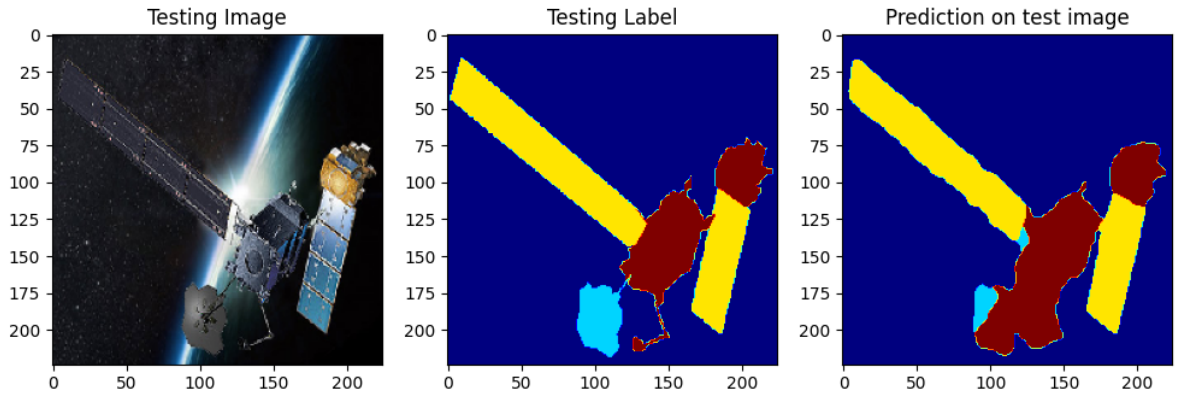

Figure 86: Prediction on test image

From 83 and 84 can be seen that the score values retrieved were higher than the previous experiment and as for the loss function trends that were almost the same. The overfitting decreased a little bit from the previous experiment even if not considerably but the higher values of the validation were still a good improvement.

6.6 Experiment 6

Here has been tried optimizer Nadam instead of the usual Adam, to test if it could gave better results.

	mean IoU score	**Loss**
Training	90%	0.7716
Validation	73%	0.8540

Table 21: Experiment 6 report

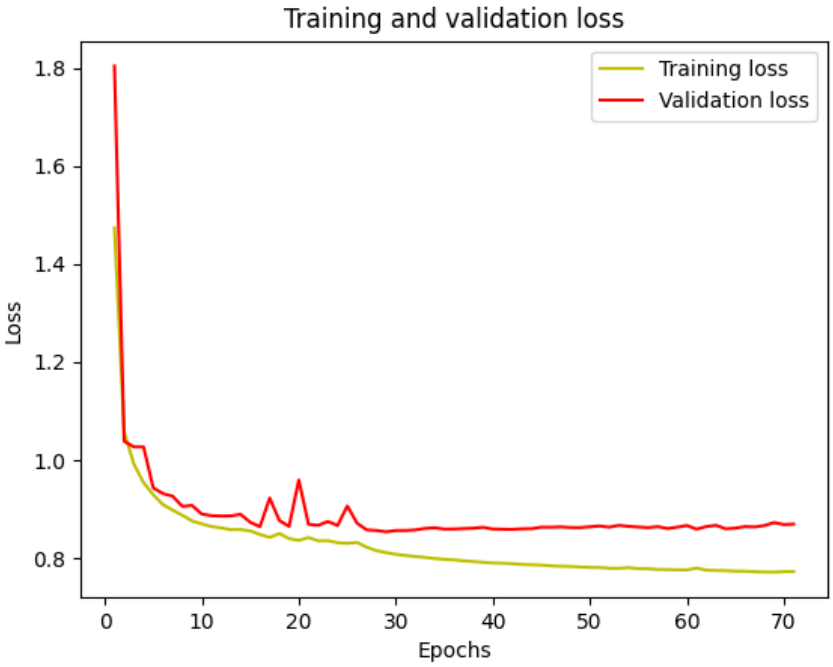

Figure 87: Training and Validation losses

Figure 88: Training and Validation IoU metrics

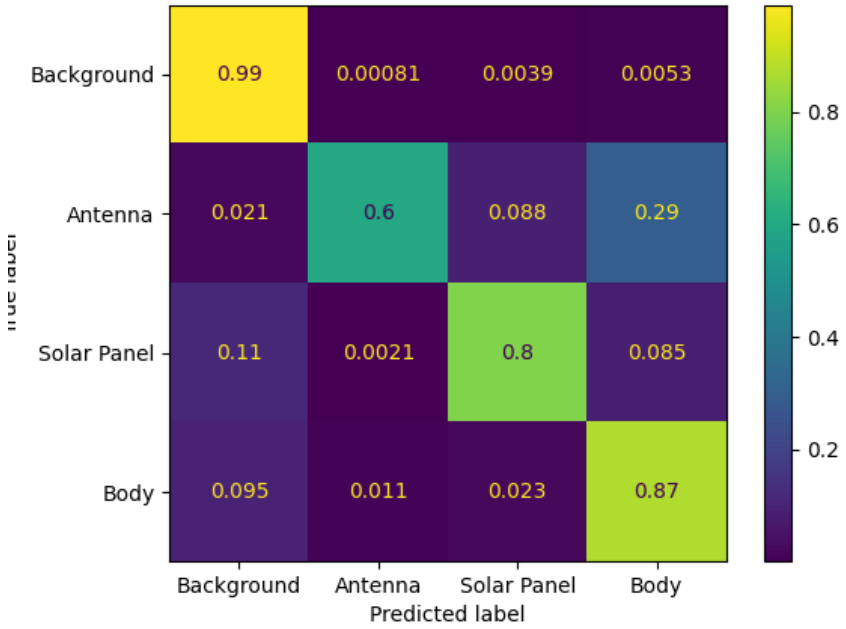

Figure 89: Confusion matrix

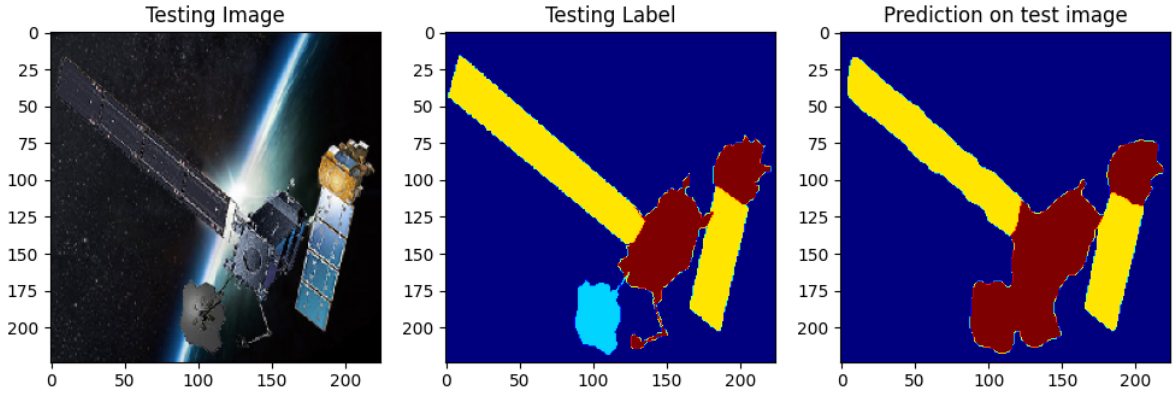

Figure 90: Prediction on test image

The results from 87 and 88 shows that Nadam didn't improve our results showing lower values than the previous experiment in which Adam optimizer was used.

6.7 Experiment 7

From the previous net another loss function has been taken in consideration: the focal loss.

	mean IoU score	Loss
Training	86%	0.7220
Validation	72%	0.7285

Table 22: Experiment 7 report

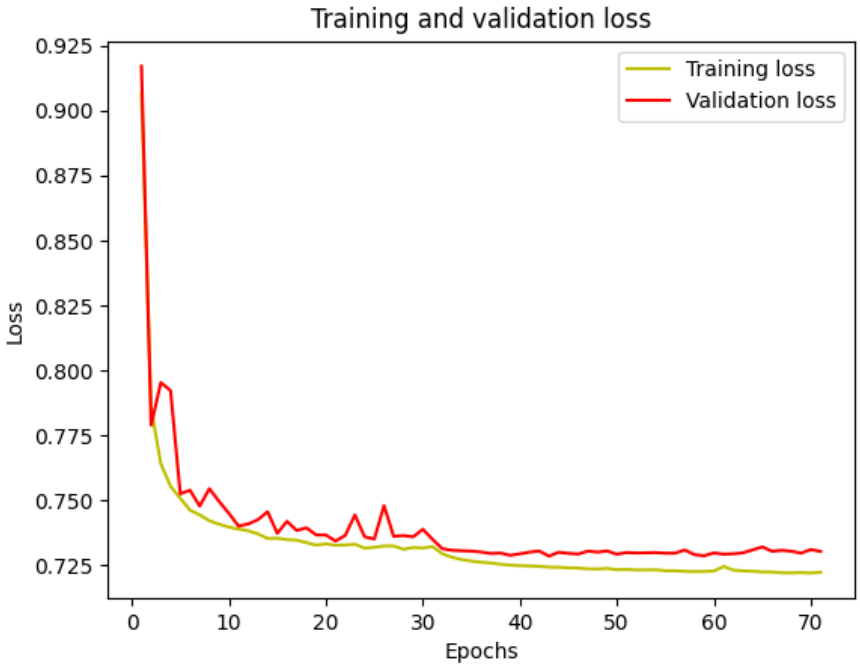

Figure 91: Training and Validation losses

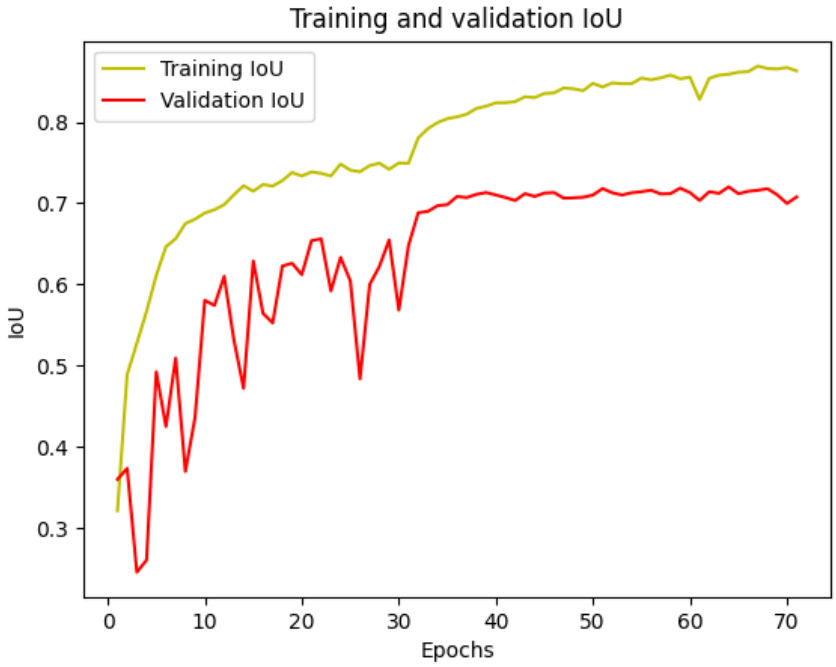

Figure 92: Training and Validation IoU metrics

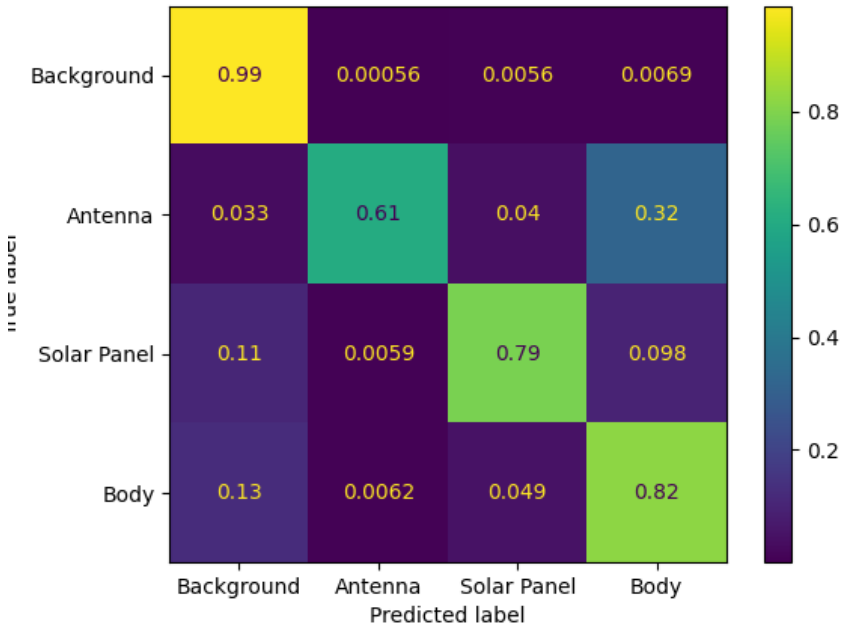

Figure 93: Confusion matrix

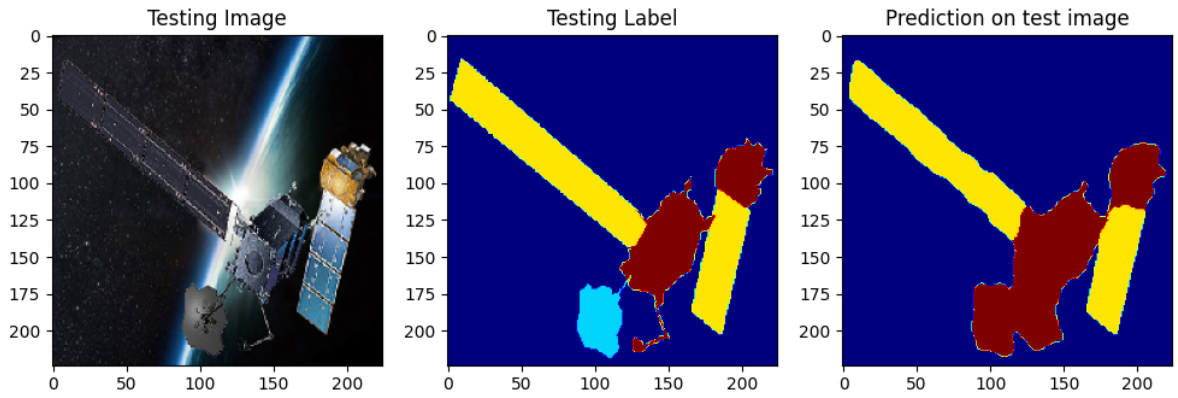

Figure 94: Prediction on test image

The performance showed in 91 and 92 the focal loss helped improving the results of experiment 6 but it was still a bad result compared to the experiment 5 which still remains the best one.

6.8 Experiment 8

Given the previous net here the sample weights were added to the fit.

	mean IoU score	**Loss**
Training	80%	0.7226
Validation	68%	0.8567

Table 23: Experiment 8 report

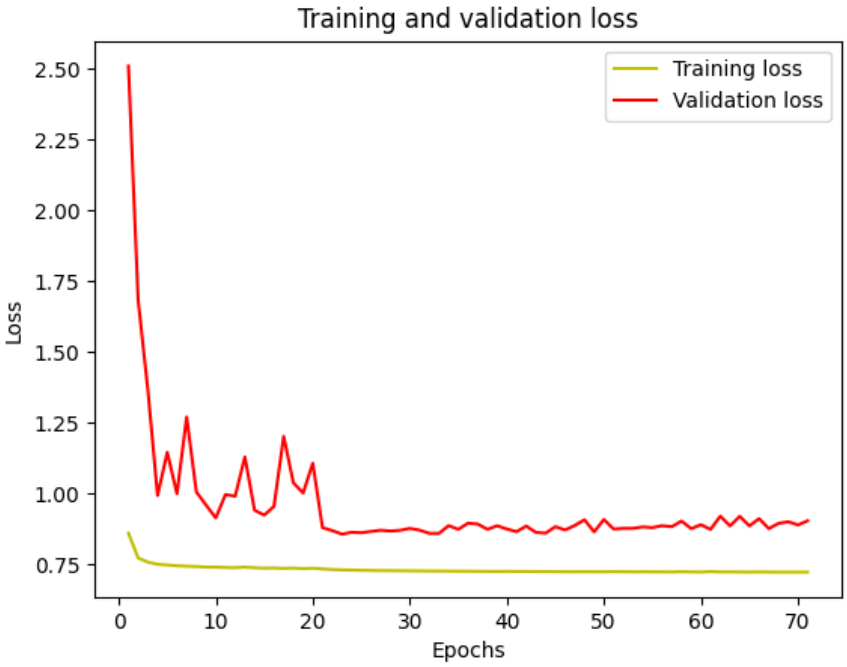

Figure 95: Training and Validation losses

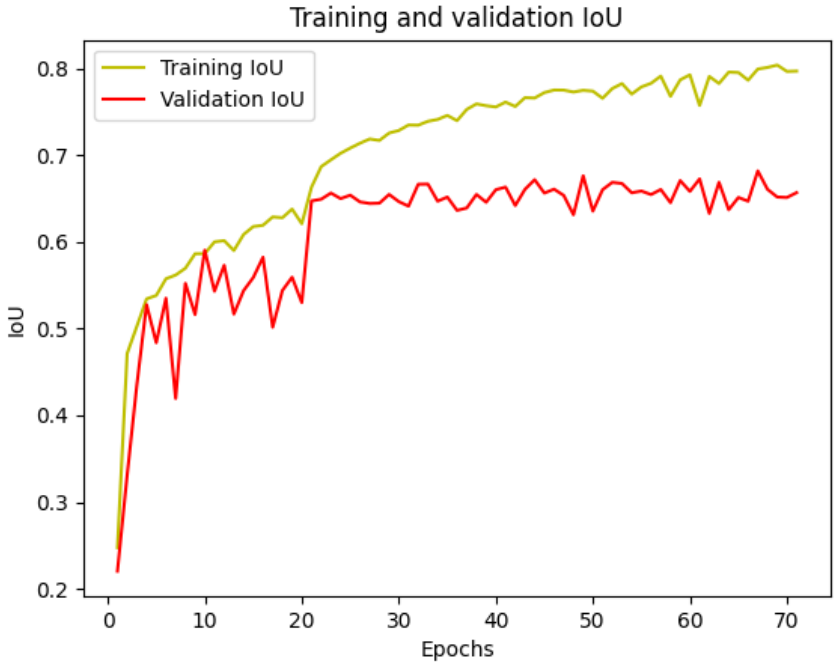

Figure 96: Training and Validation IoU metrics

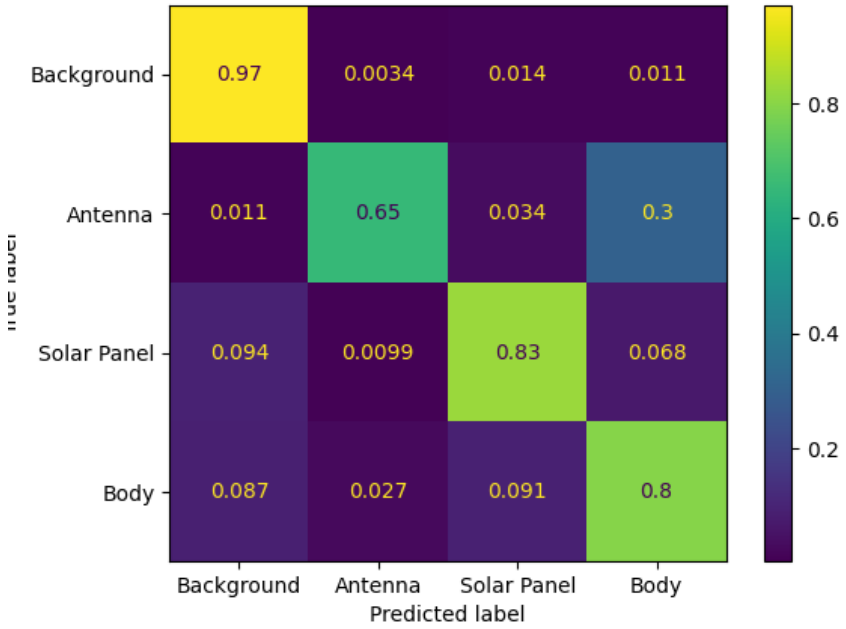

Figure 97: Confusion matrix

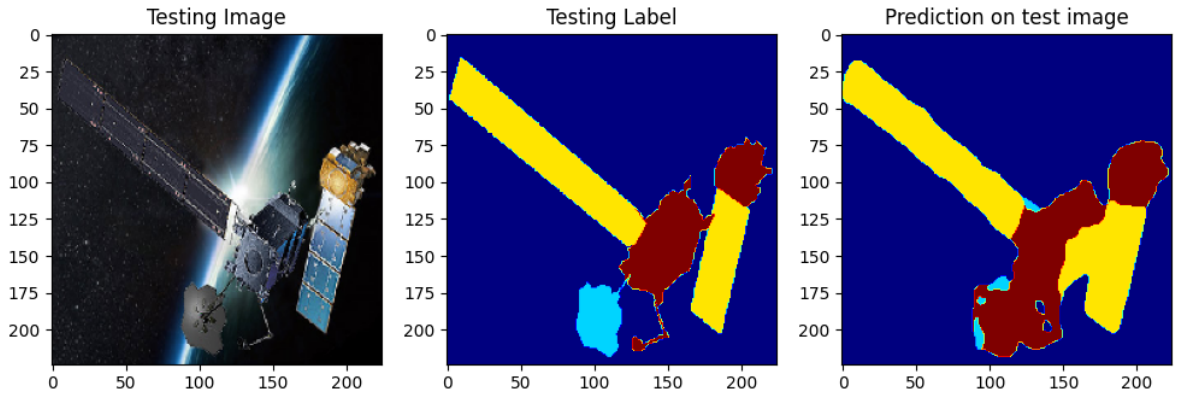

Figure 98: Prediction on test image

The results obtained, as seen from 96 and 96 were not so good even compared to the previous experiment.

6.9 Experiment 9

As last experiment for ResNet34 a fine tuning approach has been tried. It consisted in unblocking the last layer of the encoding in order to train it along the decoder part and insert a dropout layer between the output of the encoder and the beginning of the decoder with value setted to 0.5.

	mean IoU score	Loss
Training	88%	0.7781
Validation	73%	0.8375

Table 24: Experiment 9 report

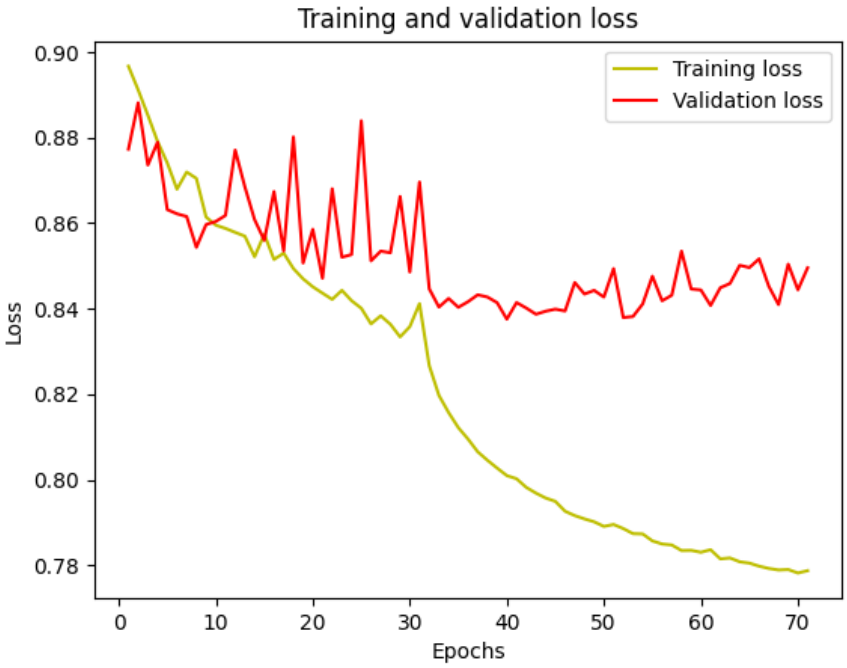

Figure 99: Training and Validation losses

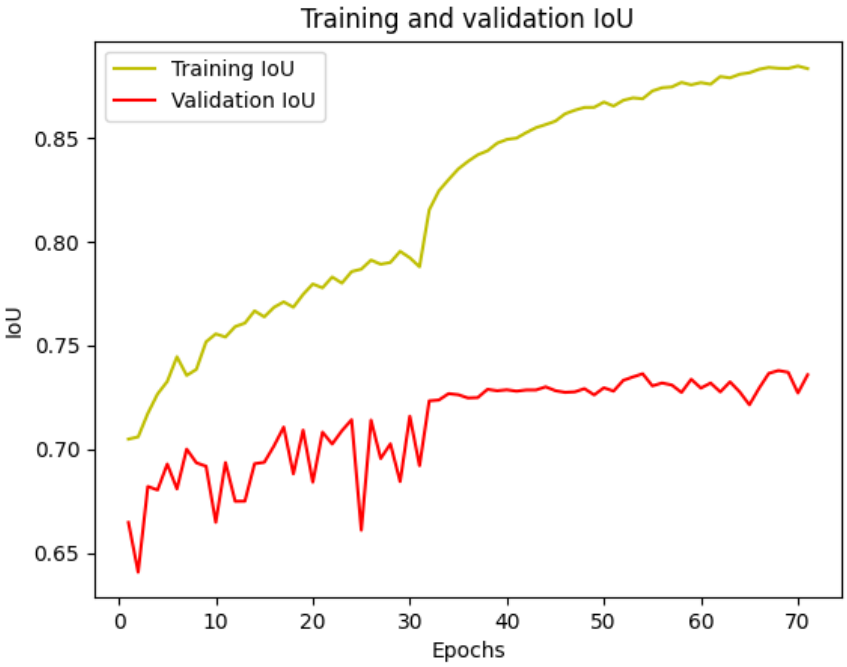

Figure 100: Training and Validation IoU metrics

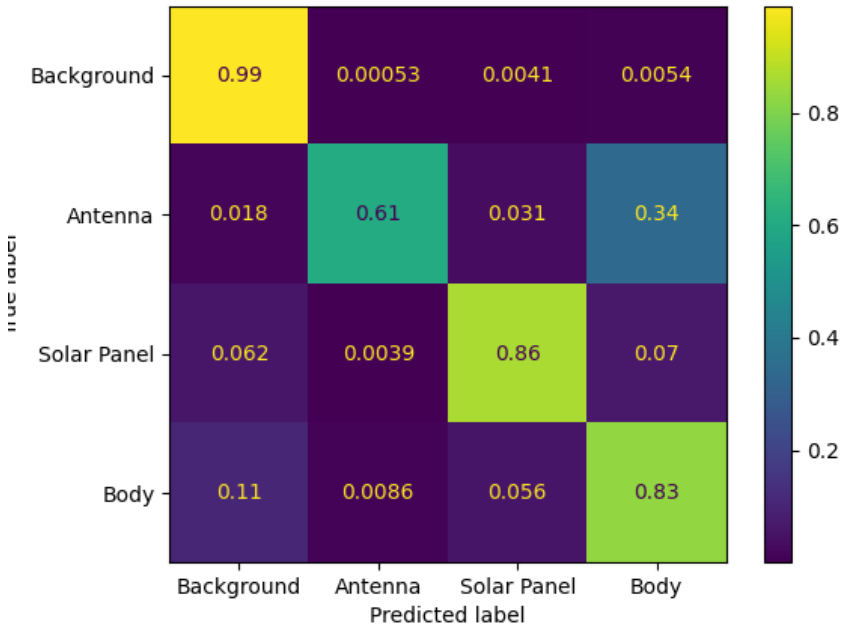

Figure 101: Confusion matrix

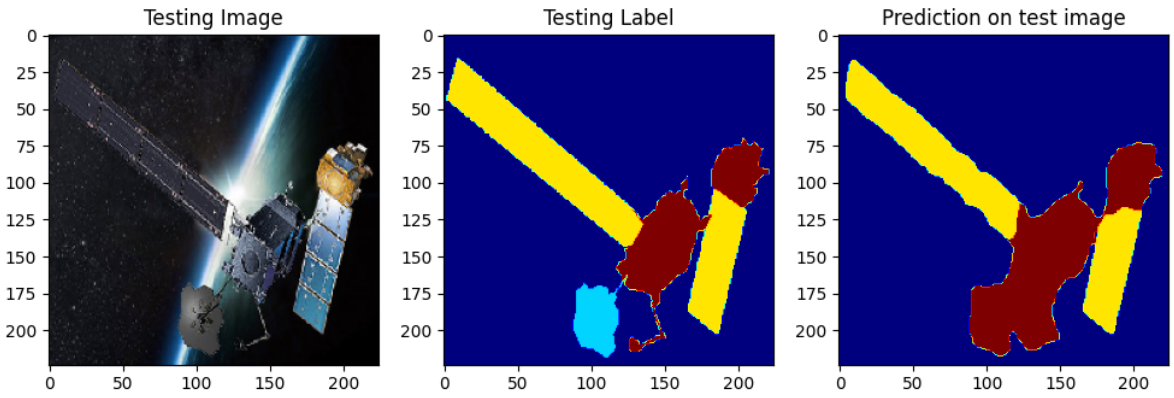

Figure 102: Prediction on test image

The 99 and 100 shows a good increase in performance. The validation score is the highest achieved so far as for the overfitting and difference with the training score and validation function having closer values. Even if the loss values may not be the best ones achieved, the results achieved for the validation score make this experiment the best one so far, even if with a little of compromise between overfitting and iou score.

7 VGG16 - U-Net

In this section are reported some experiments done by using another pretrained network, VGG16. The choice was made due to the fact that this net is one of the most used for the semantic segmentation problems, it's able to recognize even tiny portions of a class inside the image and, also due to it's better performances on small datasets. For all the following experiments, some common parameters and characteristics for the network have been used. Five blocks for the decoder were used with the corresponding filter values starting from 8 to 126 for the first experiment and from 16 up to 256 for all the following ones. As optimizer Adam was used as it is one of the standard optimizers used for this kind of problems. Cross-entropy was used as loss function at first in order to help minimizing the loss values and functions. Also, in the decoder part, batch normalization layers were added between each convolutional layers and it's activation function. As for the metrics IoU was used as for all the previous other experiments.

7.1 Experiment 1

The first experiment started by using directly 3 augmentation blocks and 3 dropouts setted to 0.3 since having a huge number of parameters required a dropout for better results.

	mean IoU score	**Loss**
Training	82%	0.0632
Validation	63%	0.1474

Table 25: Experiment 1 report

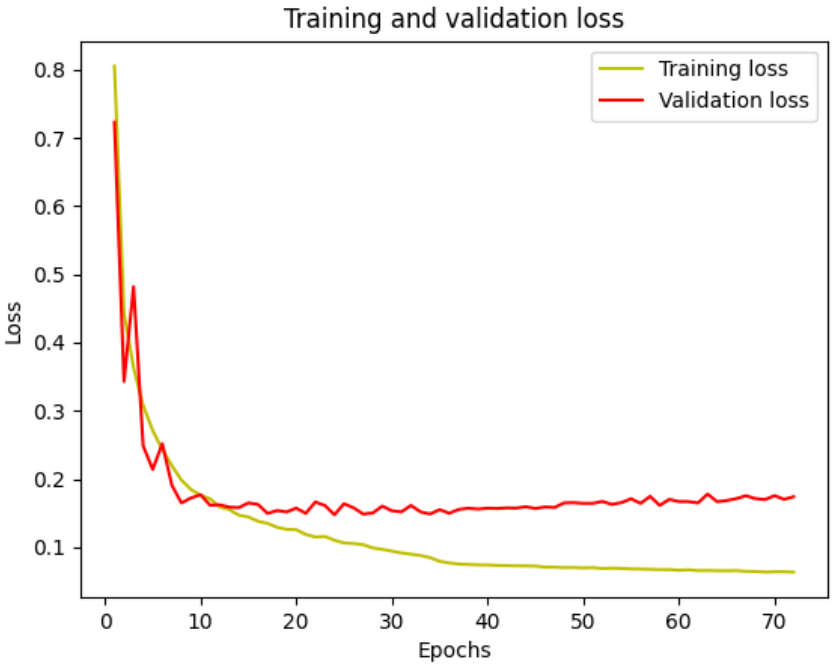

Figure 103: Training and Validation losses

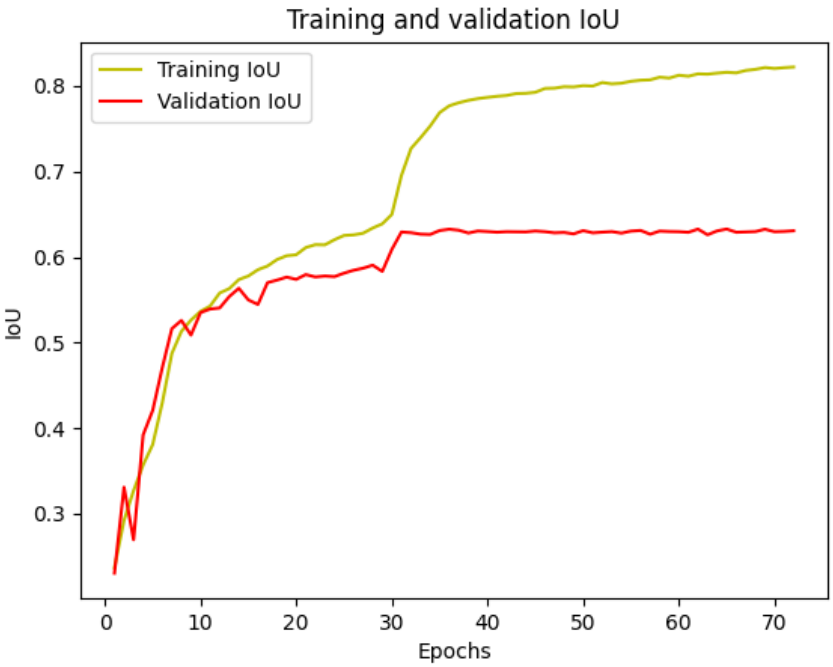

Figure 104: Training and Validation IoU metrics

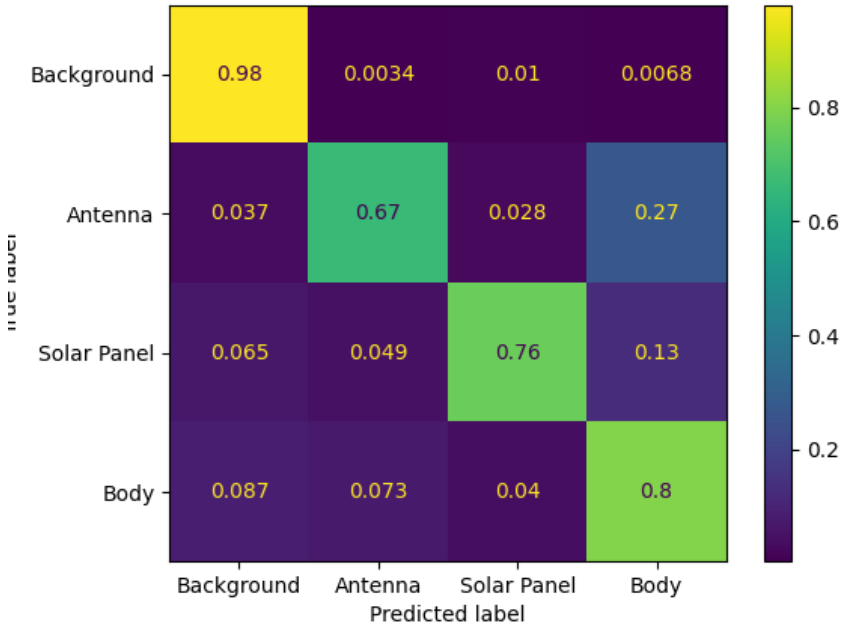

Figure 105: Confusion matrix

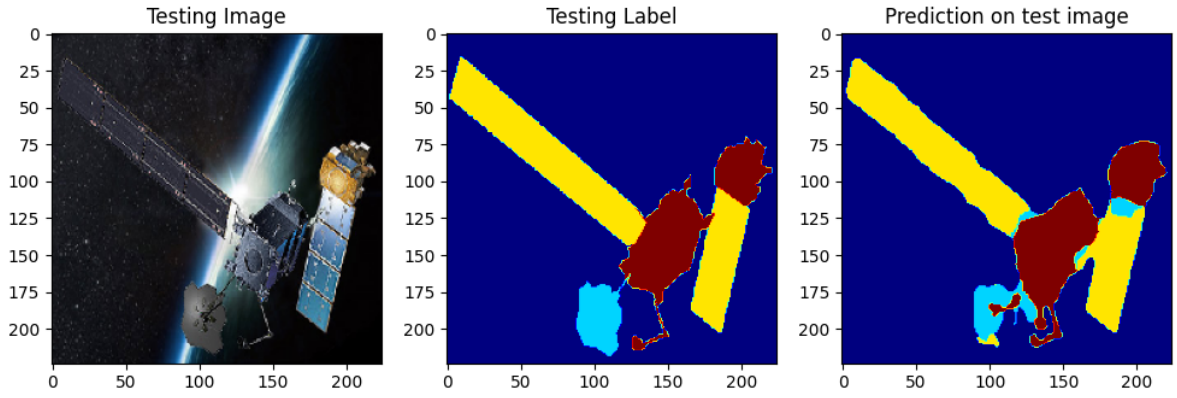

Figure 106: Prediction on test image

The results reported in 103 and 104 are not good and show a huge overfitting, as for the IoU validation score which is also lower than the RESNET34 results.

7.2 Experiment 2

In order to try to increase the score of the validation the convolution filters here, and in the next experiments, were increased from 8-128 to 16-256.

	mean IoU score	Loss
Training	90%	0.0393
Validation	69%	0.1293

Table 26: Experiment 2 report

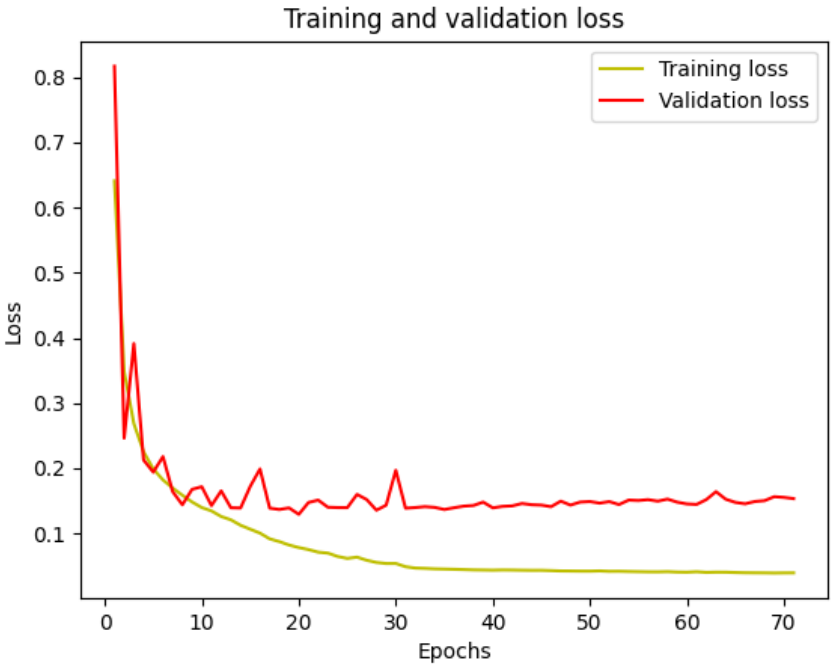

Figure 107: Training and Validation losses

Figure 108: Training and Validation IoU metrics

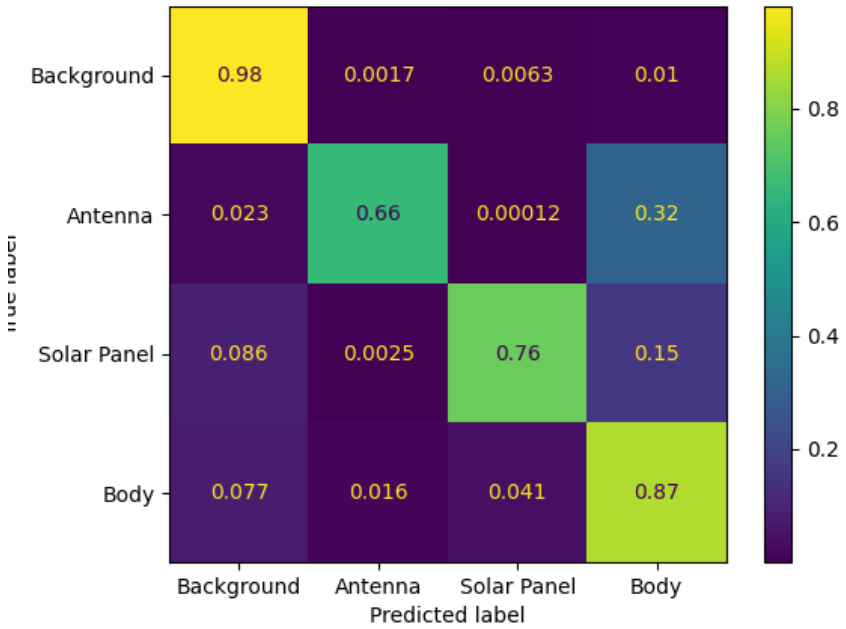

Figure 109: Confusion matrix

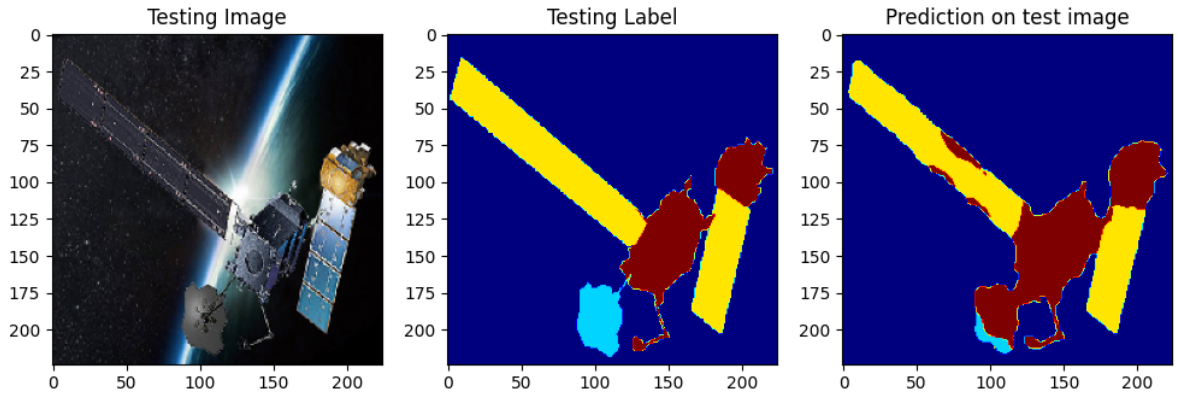

Figure 110: Prediction on test image

By looking the confusion matrix at 109 and the functions 107 and 108 the scores improved a little bit but with them also the overfitting increased.

7.3 Experiment 3

In order to lower the overfitting, in this experiment the dropout values were increased from 0.3 to 0.5.

	mean IoU score	**Loss**
Training	89%	0.0460
Validation	69%	0.1314

Table 27: Experiment 3 report

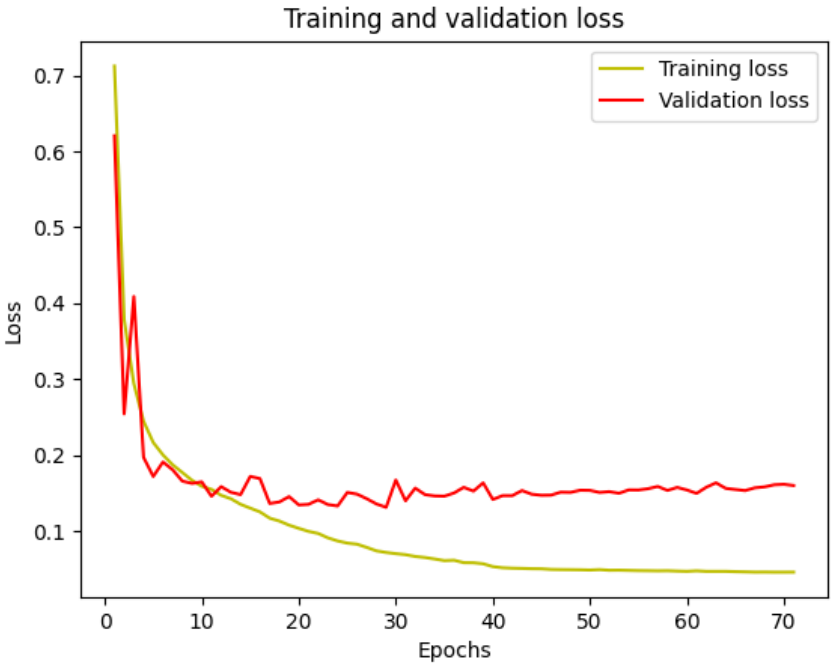

Figure 111: Training and Validation losses

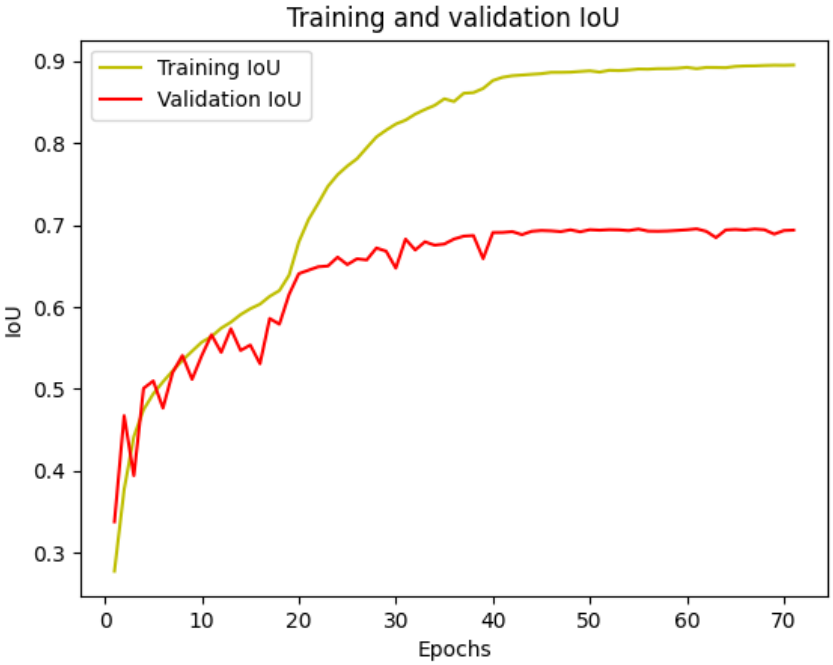

Figure 112: Training and Validation IoU metrics

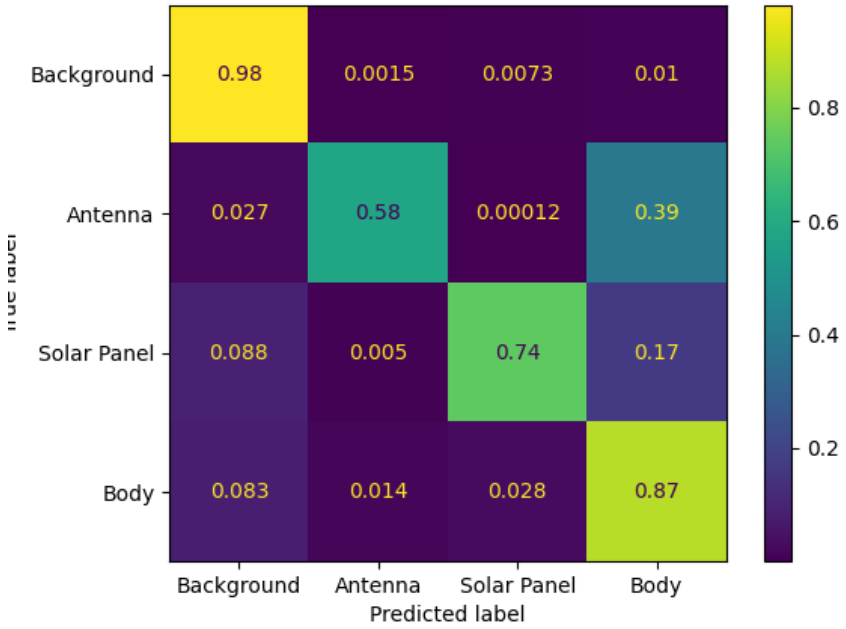

Figure 113: Confusion matrix

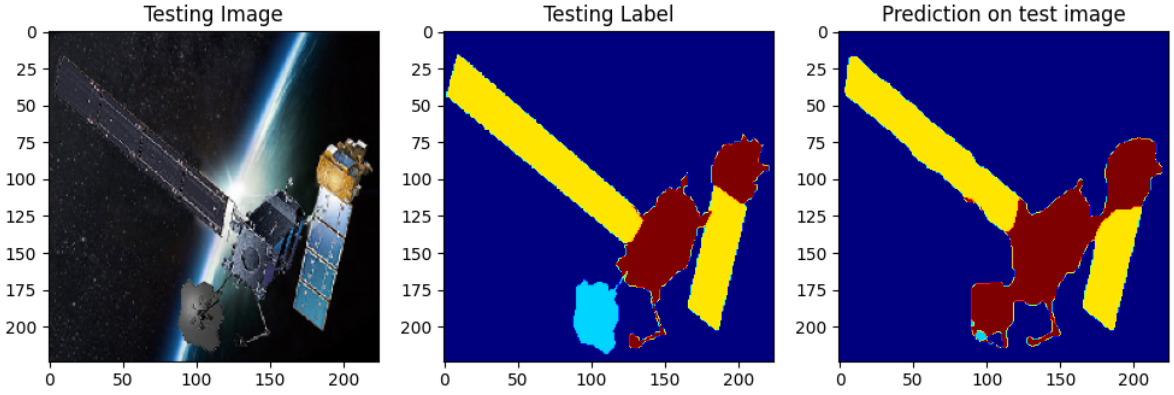

Figure 114: Prediction on test image

It can be seen from 111 and 112 that the increase of the dropout helped to reduce the overfitting but not considerably in this case. Also 113 shows a decrease of the classification scores for antenna and solar panel.

7.4 Experiment 4

In order to try to improve the last network, a L2-regulatization has been applied considering all the setting used for experiment 3. The model here has been trained for 100 epochs with early stopping.

	mean IoU score	Loss
Training	73%	0.2978
Validation	64%	0.3307

Table 28: Experiment 4 report

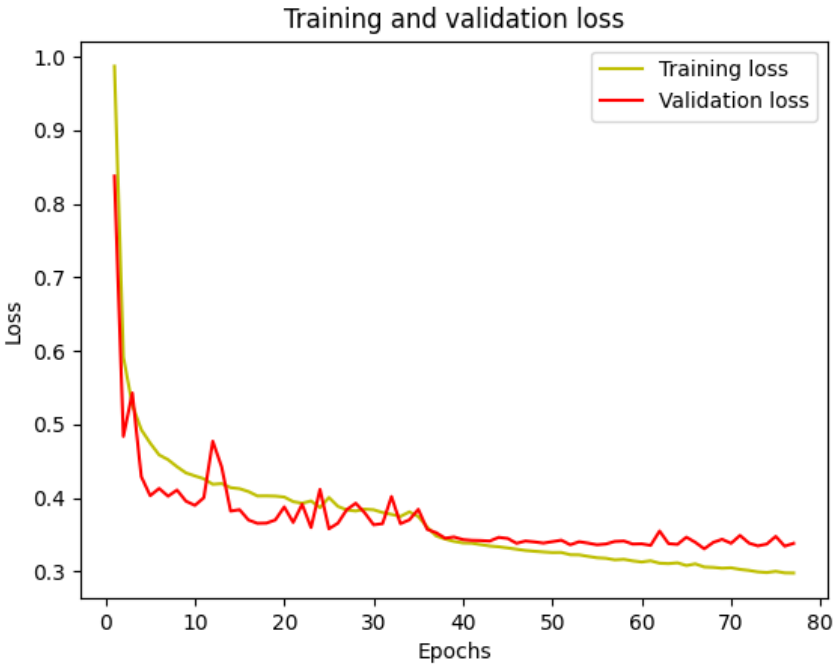

Figure 115: Training and Validation losses

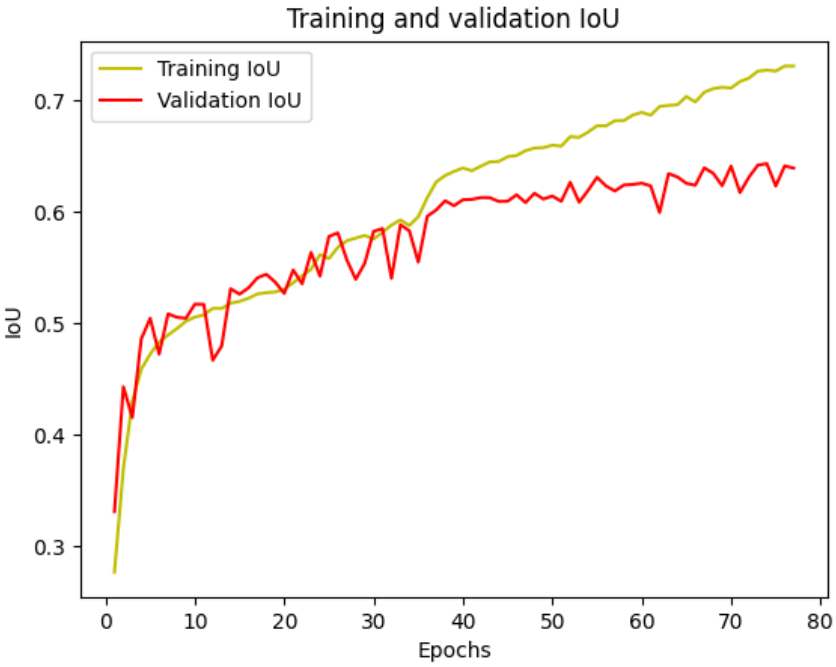

Figure 116: Training and Validation IoU metrics

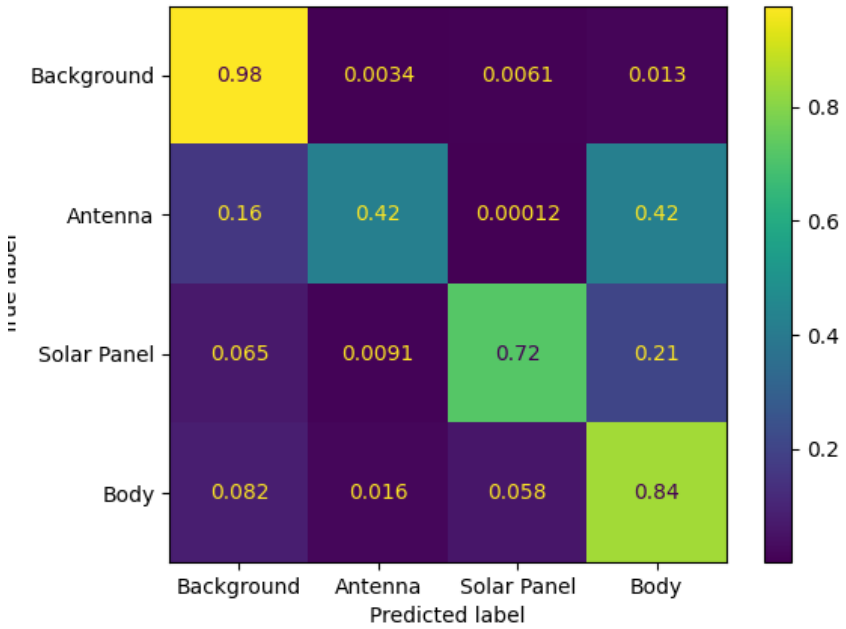

Figure 117: Confusion matrix

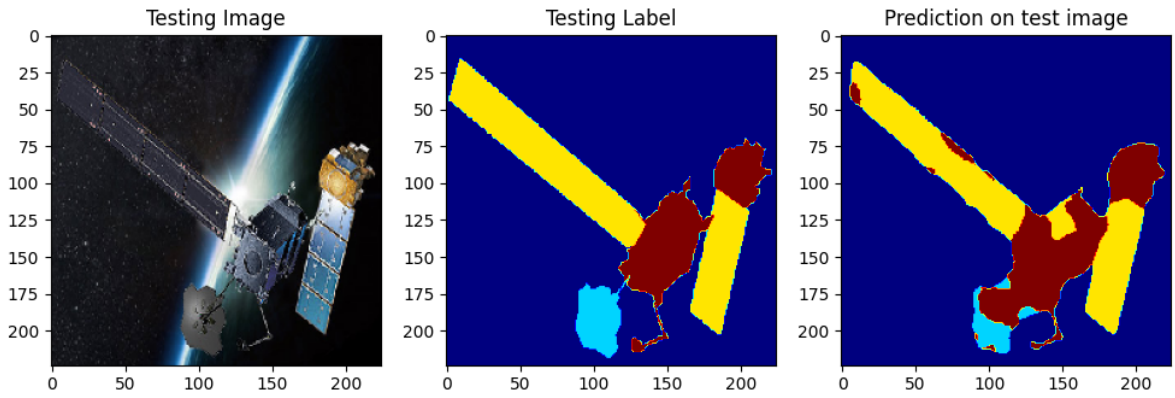

Figure 118: Prediction on test image

It can be seen from 115 and 116 that the values obtained for the loss function are higher than the previous ones and the validation score decreased, but the overfitting decreased considerably to a value smaller than before. Considering that the model has been trained for less than 100 epochs and, by looking the graphs, there was a possibility of improvement by increasing the number of epochs.

7.5 Experiment 5

From the experiment 5 the optimizer used was changed to Nadam to check the differences.

	mean IoU score	**Loss**
Training	87%	0.0537
Validation	68%	0.1309

Table 29: Experiment 5 report

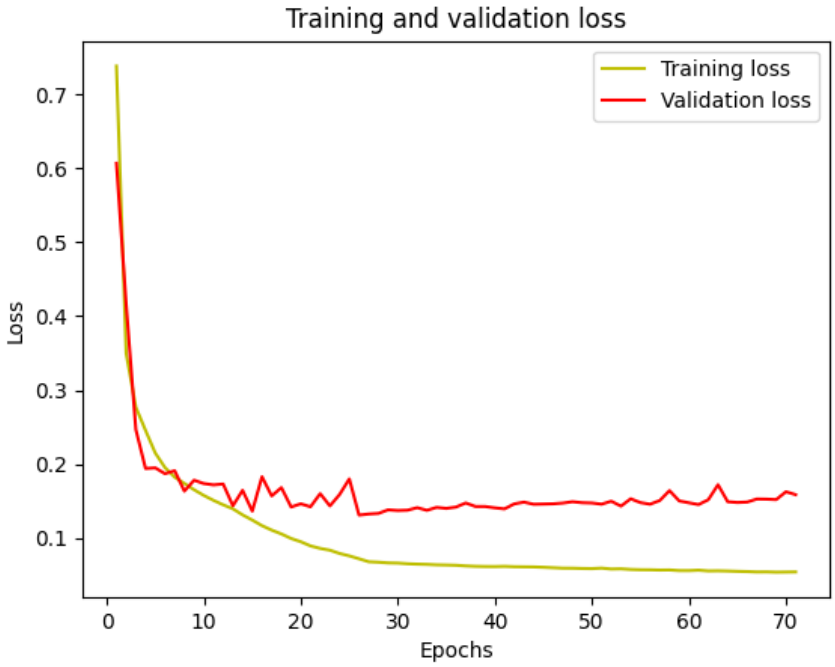

Figure 119: Training and Validation losses

Figure 120: Training and Validation IoU metrics

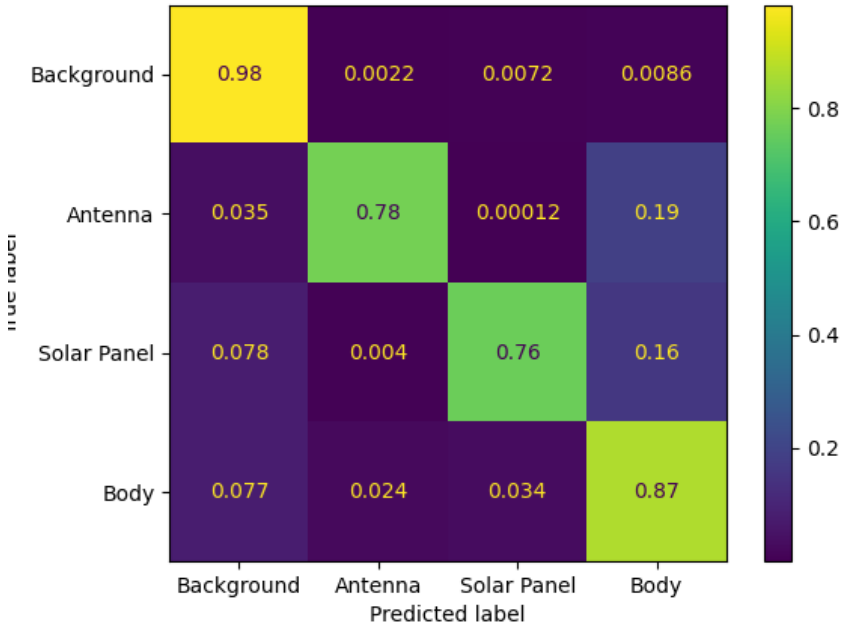

Figure 121: Confusion matrix

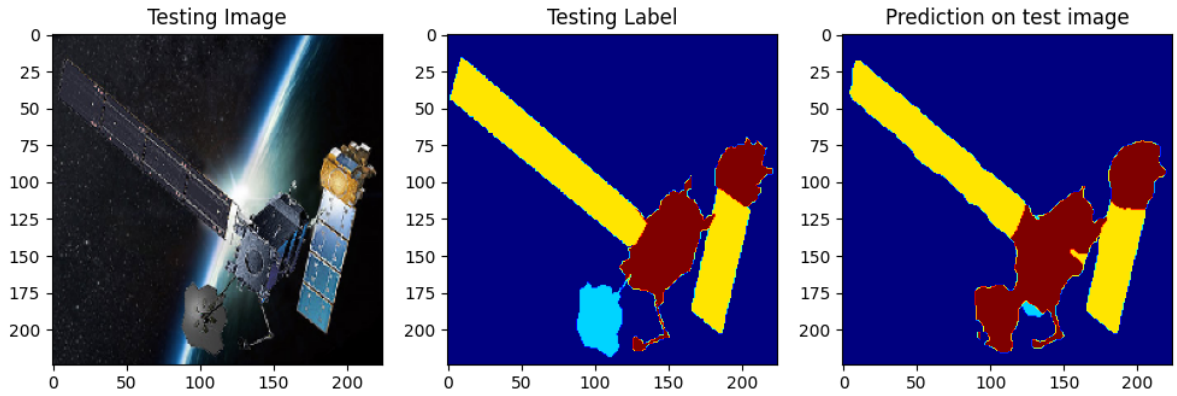

Figure 122: Prediction on test image

The overall values of loss 119 and validation score 120 increased from the previous experiment but with them also the overfitting was not as good as before.

7.6 Experiment 6

In this experiment cross-entropy was changed with the focal loss function.

	mean IoU score	Loss
Training	87%	0.0050
Validation	68%	0.0137

Table 30: Experiment 6 report

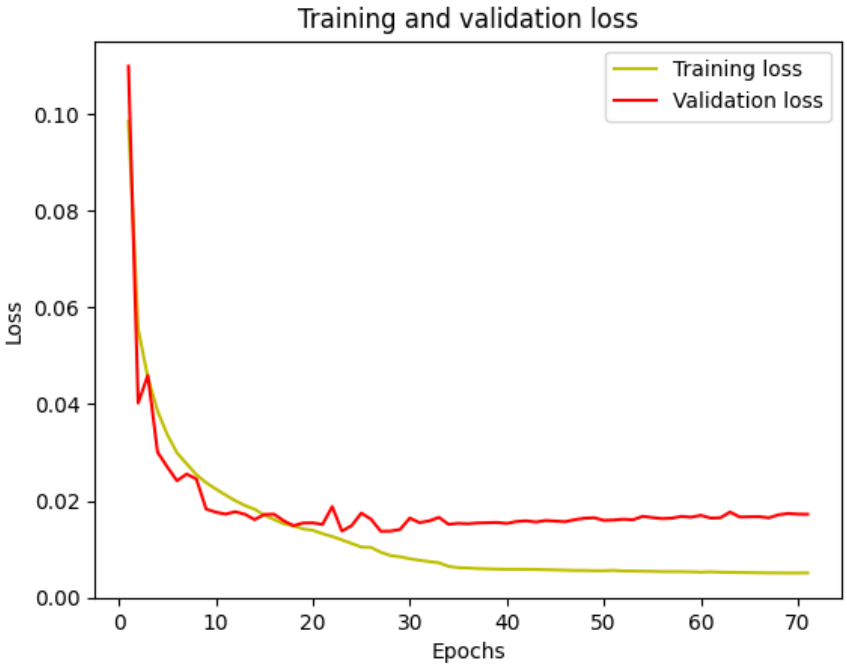

Figure 123: Training and Validation losses

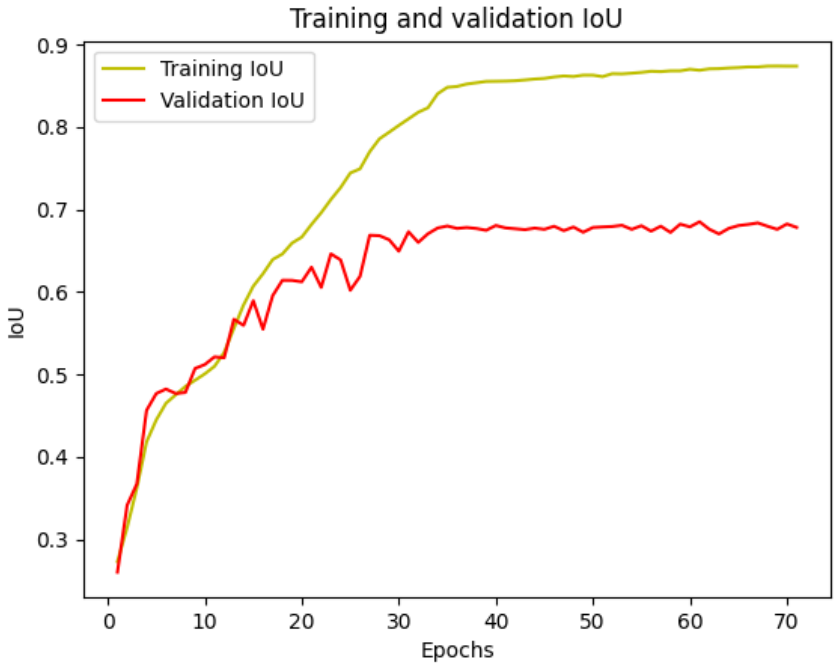

Figure 124: Training and Validation IoU metrics

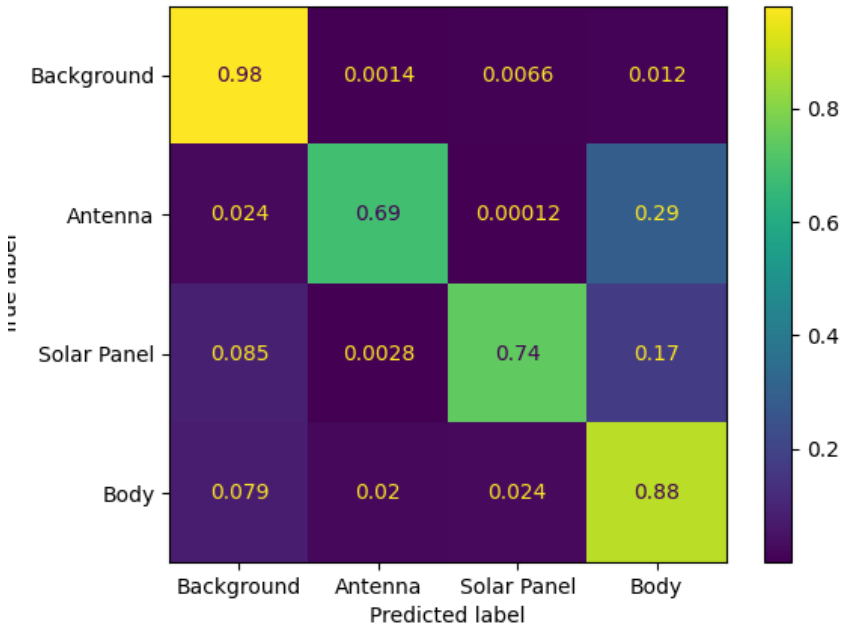

Figure 125: Confusion matrix

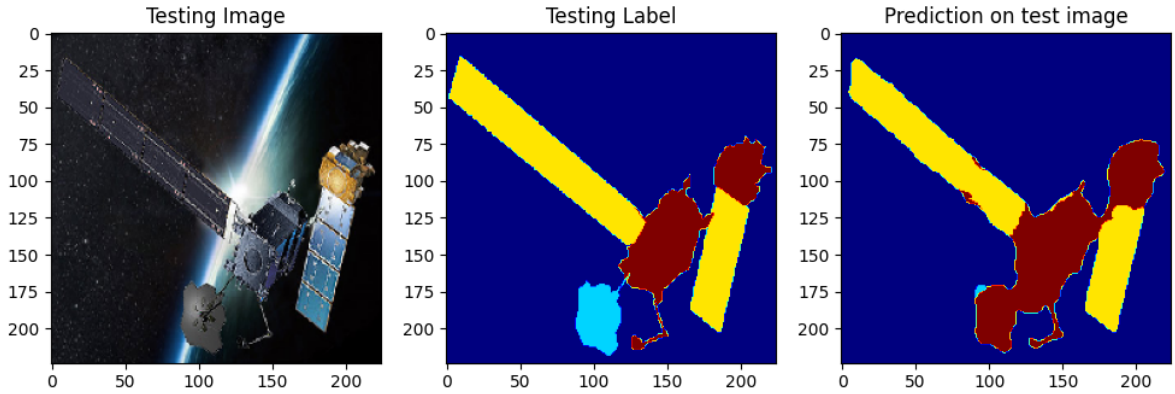

Figure 126: Prediction on test image

As shown in 123 and 124 the results obtained are almost similar to the previous experiment.

7.7 Experiment 7

With the loss function turned back to cross-entropy, here, the sample weights were applied to the fit function in order to try to mitigate the class imbalance problem giving more weight and importance to the Antennas.

	mean IoU score	**Loss**
Training	83%	0.0038
Validation	68%	0.1673

Table 31: Experiment 7 report

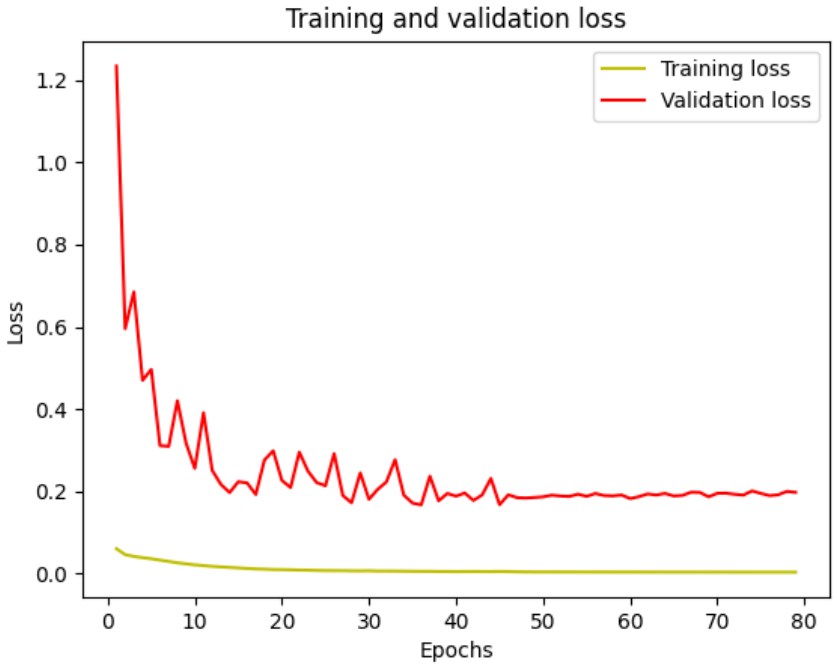

Figure 127: Training and Validation losses

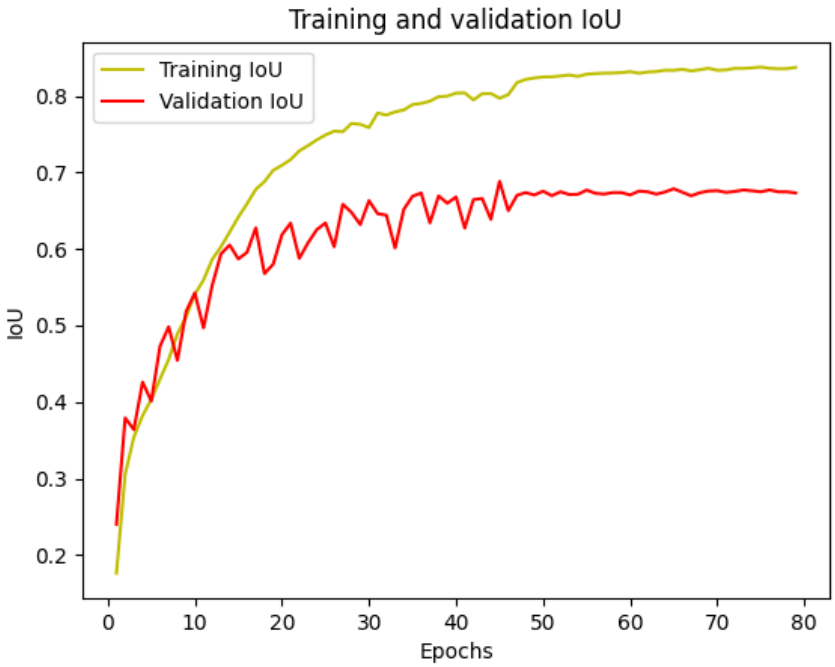

Figure 128: Training and Validation IoU metrics

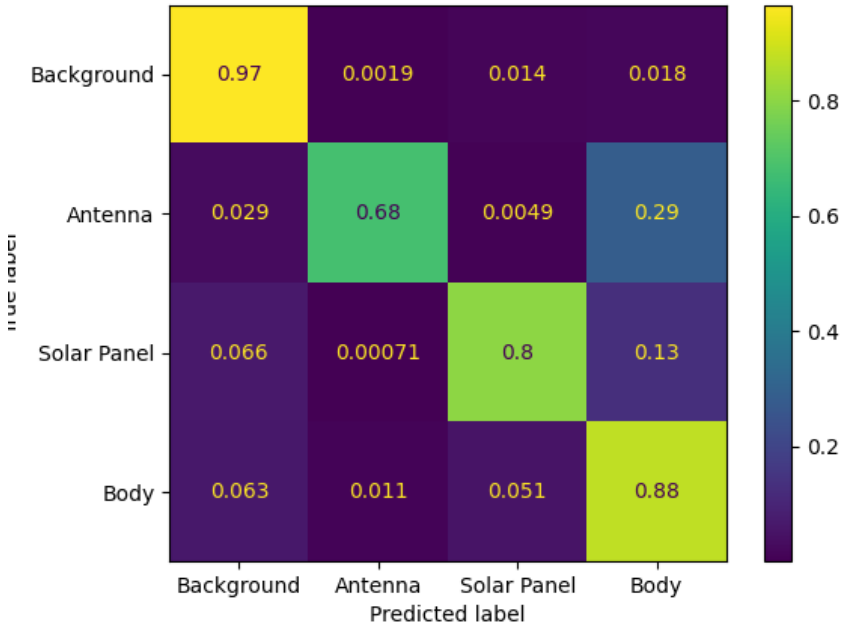

Figure 129: Confusion matrix

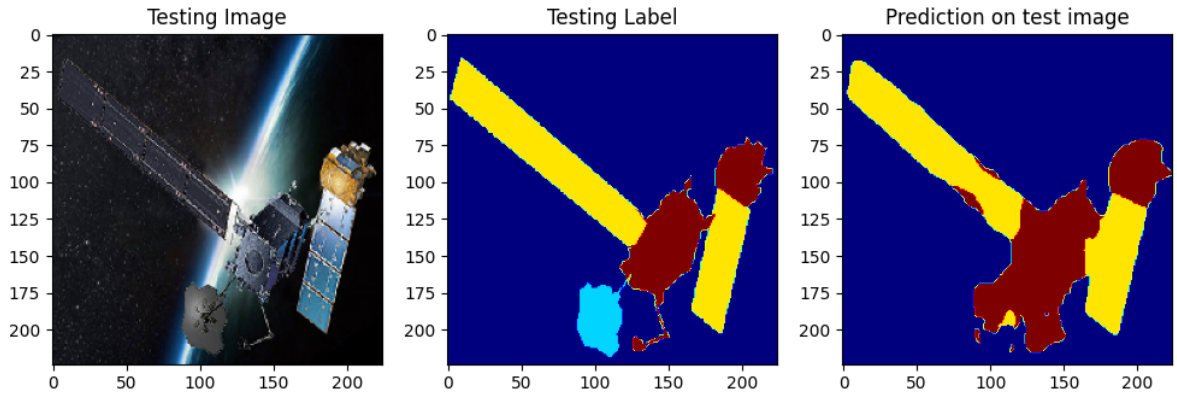

Figure 130: Prediction on test image

From 127 and 128 can be seen that the use of the sample weights didn't helped at all. The overfitting remained, approximately, at the same level of the previous experiment but the loss functions were completely different and with different trend.

7.8 Experiment 8

Since the good results the experiment 4 reported by the usage of L2 regularization, here an experiment was conducted using the same regularization technique and unblocking the last section of the pre-trained encoder.

	mean IoU score	Loss
Training	83%	0.2612
Validation	67%	0.3207

Table 32: Experiment 8 report

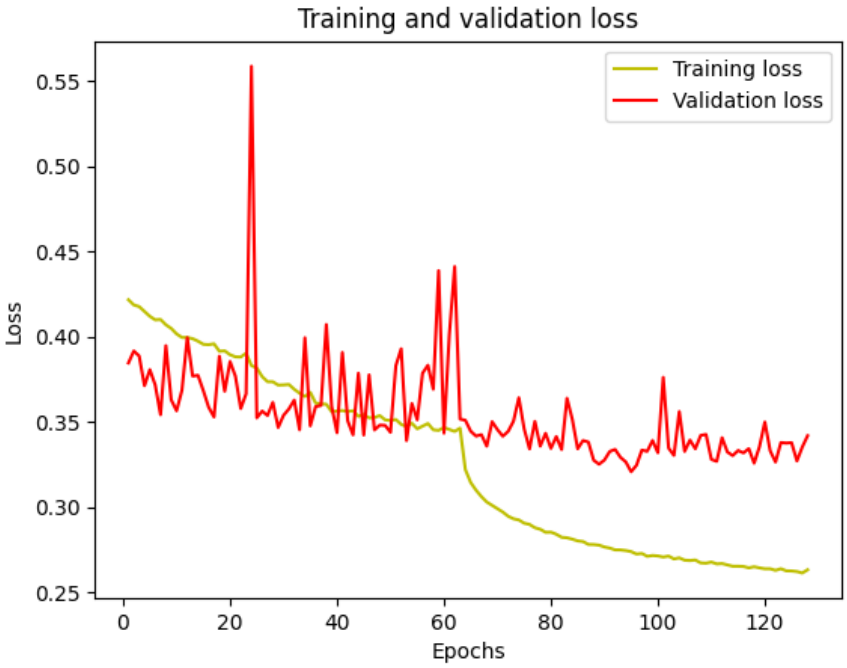

Figure 131: Training and Validation losses

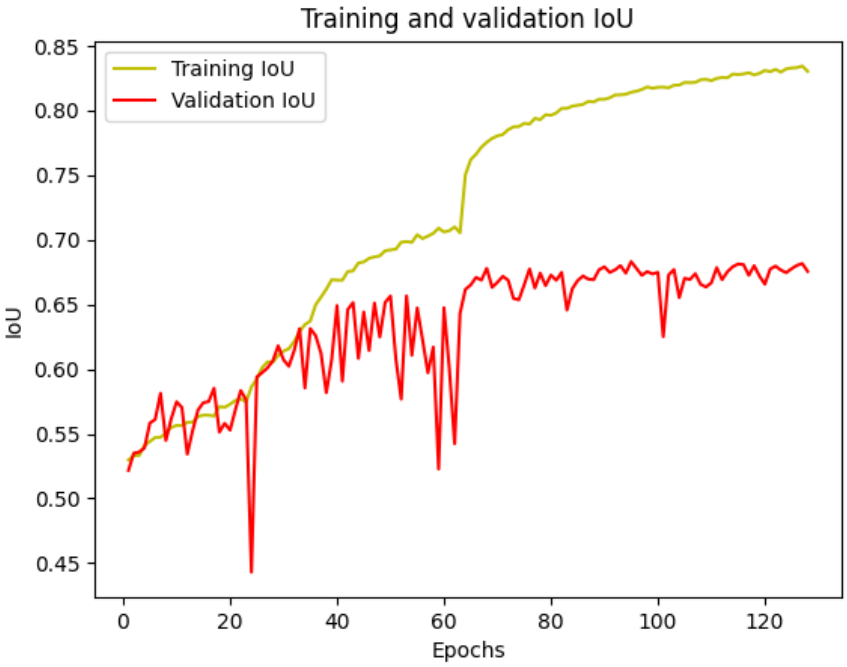

Figure 132: Training and Validation IoU metrics

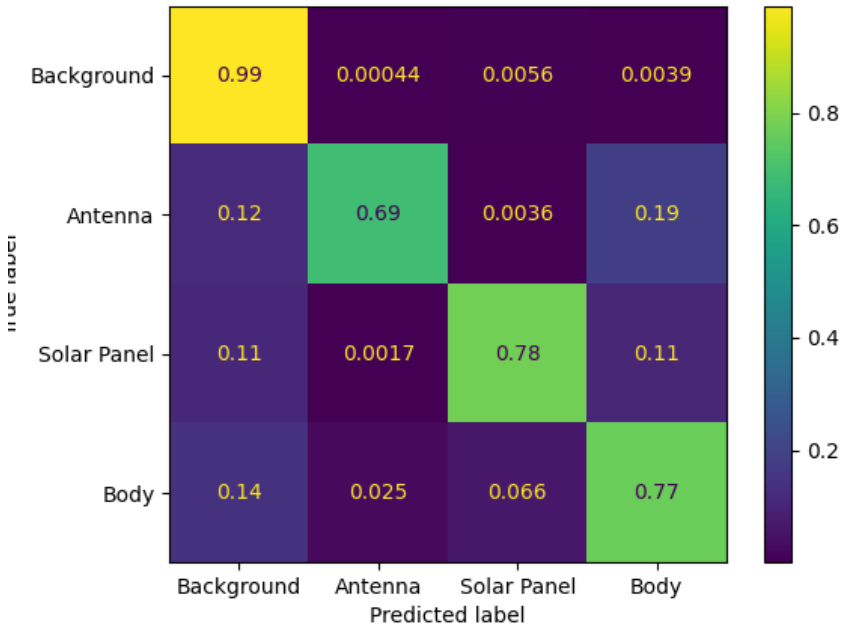

Figure 133: Confusion matrix

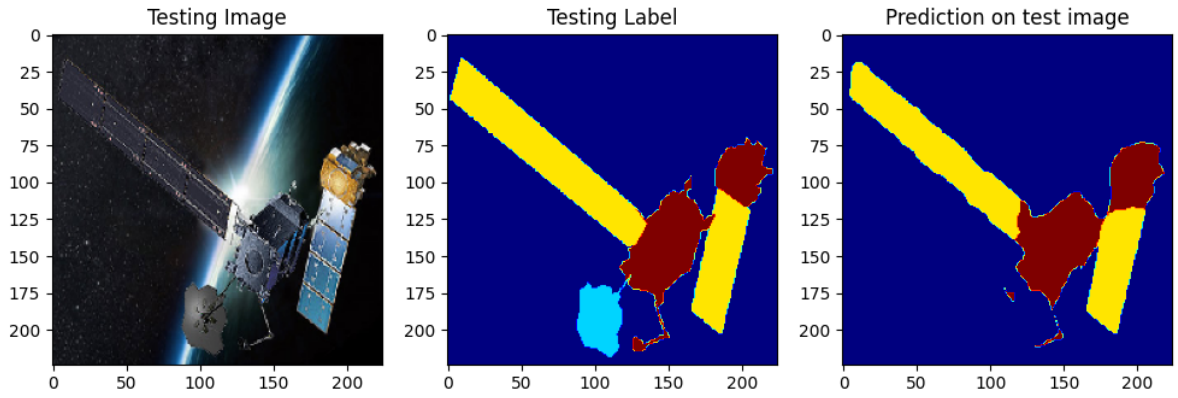

Figure 134: Prediction on test image

From 131 can be seen there's too much difference between validation and training loss, with the validation loss swinging between 0.35 and 0.40 and the training loss decreasing with a consistent collapse around the 60th epoch. Also the overfitting showed by looking 132 is very high.

8 Extra experiments

For the sake of testing the results and briefly exploring other architectures, two other experiments were conducted: one with a ResNet50 backbone and a U-Net decoder containing Attention gates and an ensemble of the best classifiers found in this project.

8.1 ResNet50 - Attention U-Net

The Attention U-Net [8] is a variant of the U-Net architecture that incorporates attention gates to enhance its performance in semantic segmentation tasks. Attention gates 135 are inserted between the contracting (downsampling) and expanding (upsampling) paths of the U-Net. The purpose of these attention gates is to selectively highlight relevant spatial features from the contracting path and propagate them to the expanding path, allowing the model to focus on the most informative regions. Their gate module consists of three main components: the query branch, the key-value branch, and the attention mechanism. The query branch takes as input a feature map from the contracting path, while the key-value branch receives a feature map from the expanding path. These branches are responsible for extracting relevant information and preparing it for the attention mechanism. The attention mechanism computes attention weights by comparing the query features with the key features. Various methods can be used to calculate these weights, such as dot product, concatenation followed by a learned mapping, or a combination of both. The resulting attention weights indicate the importance of each spatial position in the key-value feature map.The attention weights are then applied to the value features, which are obtained by convolving the key features. The weighted values are combined to create the attended representation, which is then concatenated with the feature map from the expanding path. This concatenation allows the model to incorporate the attended information into the upsampling process.

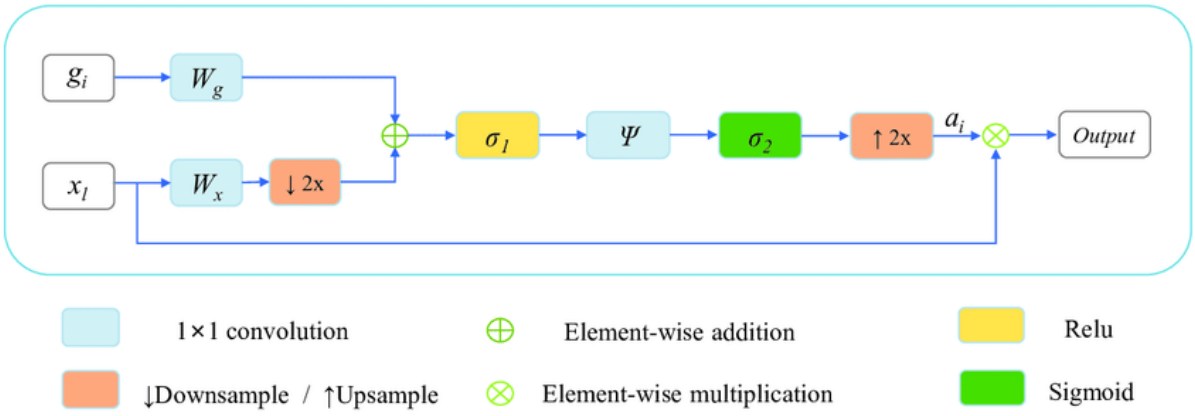

Figure 135: Attention gate structure

Following the results of the experiment on a ResNet50 backbone with an Attention U-Net decoder are reported. The model was imported from the keras-unet-collection [9]

library. Unfortunately the library couldn't be forked, meaning modification (as adding dropout or any other regularization to the model) could not be implemented so the only kind of regularization used in this case was the batch normalization on the output of the convolutional layers.

	mean IoU score	**Loss**
Training	95%	0.0178
Validation	71%	0.1052

Table 33: Experiment extra report

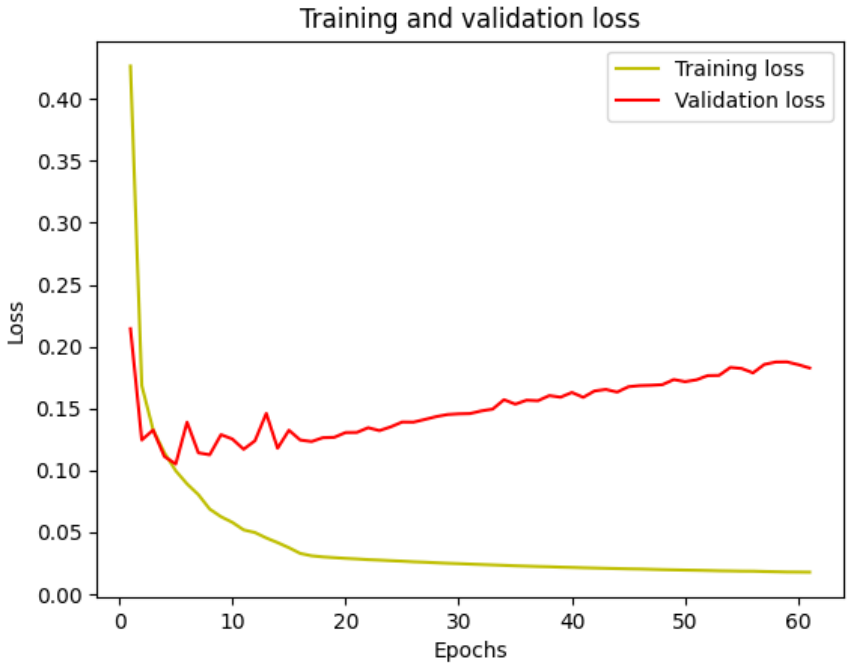

Figure 136: Training and Validation losses

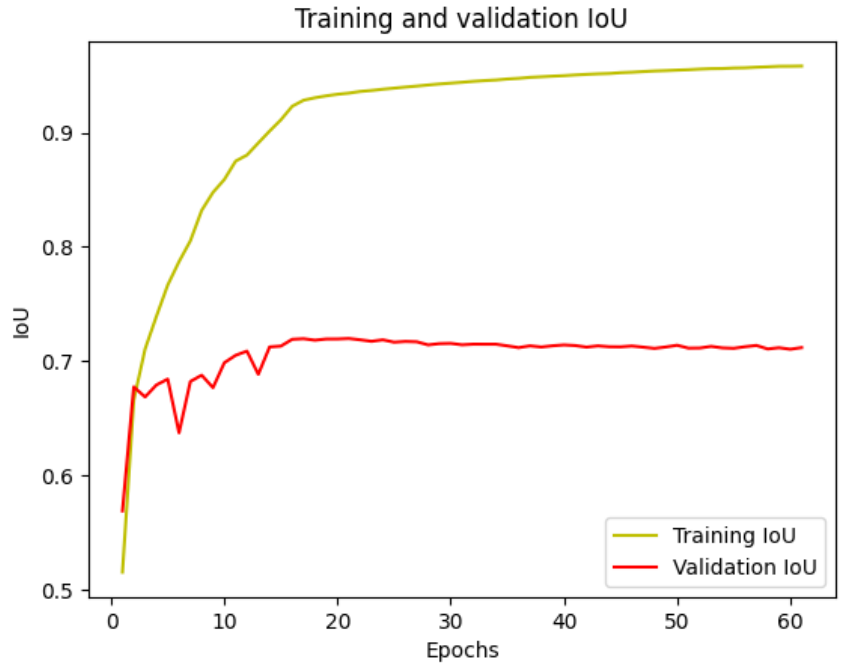

Figure 137: Training and Validation IoU metrics

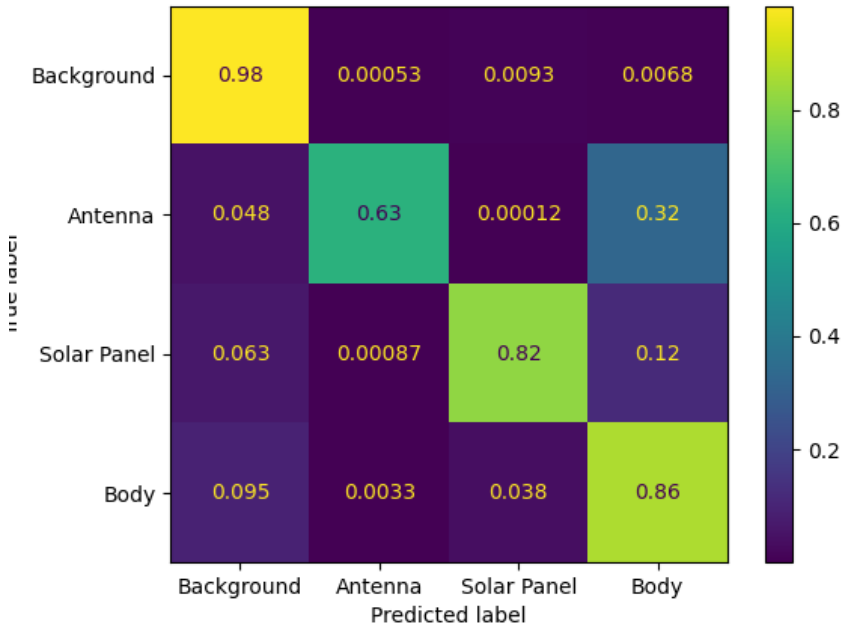

Figure 138: Confusion matrix

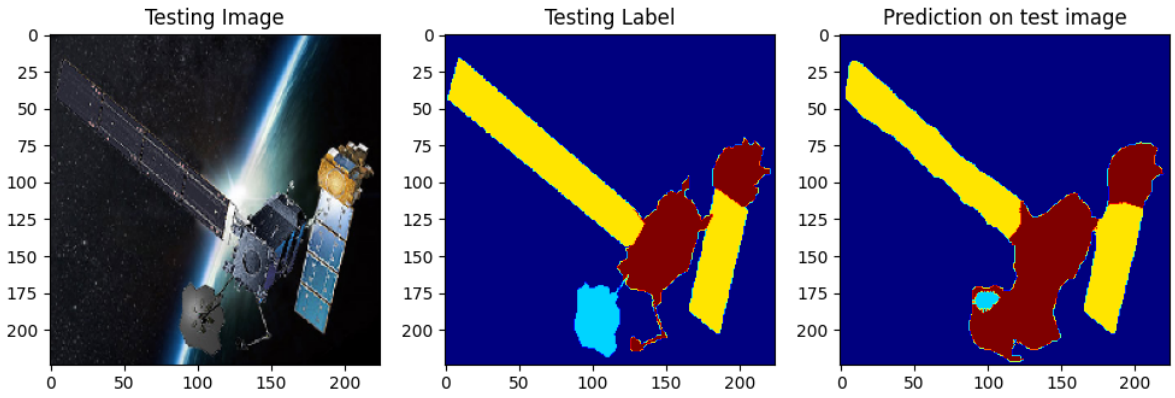

Figure 139: Prediction on test image

From 136 and 137 it can be seen that the overfitting is one of the largest found till now, but at the same time the IoU score on the validation is one of the highest reached in this report. This implies that with the possibility of adding regularization and further research for this architecture, it could be one of the best performance-wise.

8.2 Ensemble

As a last experiment, a weighted ensemble segmentation was conducted by taking for each architecture the best performing networks, specifically:

- Experiment 9 from the U-Net from scratch
- Experiment 5 from ResNet34 - U-Net
- Experiment 3 from VGG16 - U-Net

	IoU
U-Net	60%
ResNet34 - U-Net	73%
VGG16 - U-Net	69%
Esnsemble	73%

Table 34: mean IoU scores on validation set

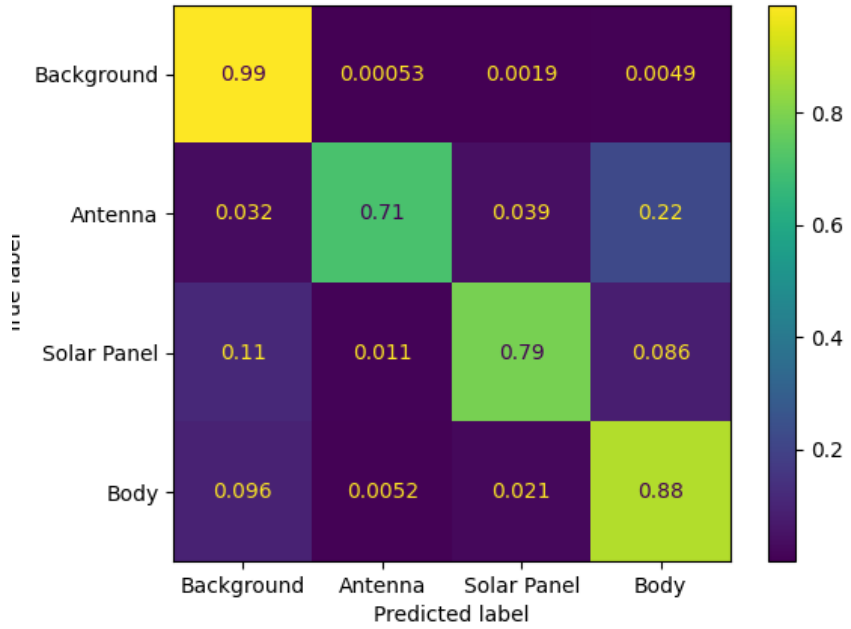

Figure 140: Confusion matrix

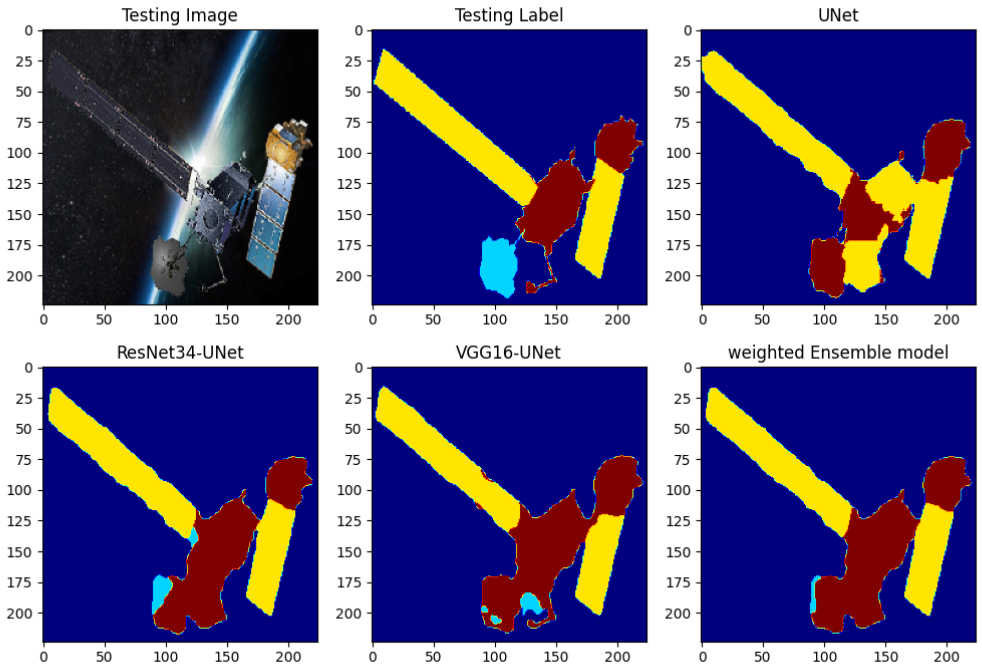

Figure 141: Prediction on test image

The weights were computed through a grid-search: each classifier was assigned a weight in range [0,3] and for each combination of weights the weighted prediction were added

and the mean IoU on the validation set 34 set was monitored. In the end the optimal weights were:

- 0 for the U-Net network from scratch
- 3 for the ResNet34 - U-Net
- 2 for the VGG16 - U-Net

The grid-search basically assigned most of the weight to the ResNet34-UNet predictions, being it the network with the highest performance in this report. Due to the limited dataset and backbone testing, these result were sadly expected, but a further study could be conducted in this scope.

9 Conclusion

The objective of this study was to try different (and limited in number) kinds of architecture for the image segmentation of artificial satellite bodies and to create a baseline for advanced research in computer vision regarding space situational awareness. As already discussed, it is an ever-growing problem with more and more public and private entities entering the space industry, this meaning also an ever-growing number of debris and space junk being released in the atmosphere.

From this report can be deducted that the main weakness was specifically the dataset. The problem was not really in its dimensions, as a properly crafted data augmentation could double or even triple the number of images of which train different kind of networks, but in the class imbalance between the images. This kind of class imbalance limited the samples on which augmentation could be executed, as its main scope was not to increase their number, but to try and rebalance, even marginally, the dataset.

All in all the results obtained with the tested networks are sufficient as a starting point and are not to be ignored. For starters, from the confusion matrices and IoU scores, all the tested network reached a reasonably high precision in segmenting the body parts and the solar panels of all the subjects, even finding the smallest ones. The problem obviously arose for the minority class, but even then most of the networks segmented at least partially well the antennas. Reaching a maximum of 73% of IoU score with ResNet34-UNet is nothing to scorn of as it basically demonstrate that it found problems only with antennas. Interestingly, the focal loss didn't bring the incresed performance expected.

Further studies could be conducted on the different networks by increasing the number of filters per block instead of building increasingly deeper networks or creating a backbone specifically crafted for the task at hand, while obviously trying to rebalance the different classes with new images.

References

[1] Holger Krag. Esa's space safety programme. In *First International Orbital Debris Conference*, volume 2109, page 6147, 2019.

[2] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation, 2015.

[3] Brett Koonce and Brett Koonce. Resnet 34. *Convolutional Neural Networks with Swift for Tensorflow: Image Recognition and Dataset Categorization*, pages 51–61, 2021.

[4] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition, 2015.

[5] Hoang Anh Dung, Bo Chen, and Tat-Jun Chin. A spacecraft dataset for detection, segmentation and parts recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2012–2019, 2021.

[6] Sungheon Park and Nojun Kwak. Analysis on the dropout effect in convolutional neural networks. In *Computer Vision–ACCV 2016: 13th Asian Conference on Computer Vision, Taipei, Taiwan, November 20-24, 2016, Revised Selected Papers, Part II 13*, pages 189–204. Springer, 2017.

[7] Pavel Iakubovskii. Segmentation models. `https://github.com/qubvel/segmentation_models`, 2019.

[8] Ozan Oktay, Jo Schlemper, Loic Le Folgoc, Matthew Lee, Mattias Heinrich, Kazunari Misawa, Kensaku Mori, Steven McDonagh, Nils Y Hammerla, Bernhard Kainz, Ben Glocker, and Daniel Rueckert. Attention u-net: Learning where to look for the pancreas, 2018.

[9] Yingkai Sha. Keras-unet-collection. `https://github.com/yingkaisha/keras-unet-collection`, 2021.