### Multi-class Semantic Segmentation

*Tooba Mukhtar*

*Ammar Ahmad*

#### ABSTRACT

*Semantic segmentation has been a challenging computer vision problem since a few years.*

*This paper proposes a semantic segmentation neural network which combines the strengths of U-Net and residual learning for image segmentation. The network has a similar architecture to that of U-Net.*

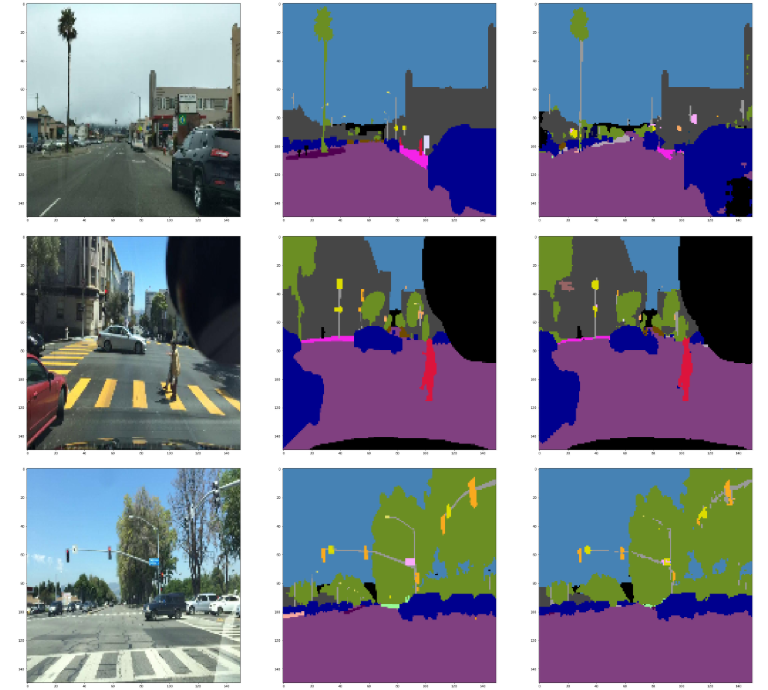
*This model has two benefits: first, the use of residual units ease the training of deep networks. Second, the skip connections within a residual unit will facilitate information propagation without degradation which makes it possible to design a network with much fewer parameters to achieve better results on semantic segmentation*. *We test our network on deep drive dataset and compare it with other state of the art deep learning based image segmentation methods. The proposed approach outperforms all the comparing methods in this paper.*

***Index Terms —*** Deep Convolutional neural networks, Semantic Segmentation, Residual Connections.

**1. INTRODUCTION**

Semantic segmentation is the task of partitioning the image into semantically meaningful parts.It has been used in various applications including medical applications, augmented reality, and most prominently automated driving.

Recently there have been an increasing interest in self driving cars. An important aspect of autonomous driving is to gain an understanding of the surroundings in which a car is moving. Semantic image segmentation is an important tool for capturing the complex relationships of the semantic entities usually found in street scenes, such as cars, road, pedestrians and sidewalks.Many of these applications require precise region boundaries. The goal of this paper is to achieve high-quality semantic segmentation with precise boundary adherence. A lot of research have suggested that a deeper network would have better performance on the results of semantic segmentation.However, it is very difficult to train a very deep architecture due to problems such as vanishing gradients. To overcome this problem there is a proposed deep residual learning architecture which uses identity mapping to facilitate training. Instead of using skip connection in Fully Convolutional Networks[1], Ronneberger et al. [2] proposed the UNet that concatenates feature maps from different layers to improve the image segmentation accuracy. This approach combines low level detail and high level semantic information resulting in high accuracy on the biomedical image segmentation [2]. Based on the concept of deep residual learning and U-net we propose an architecture that take advantage of strengths from both deep residual learning and U-Net architecture. The difference between our deep ResUnet and UNet is that we have used residual units instead of plain neural units as basic blocks.



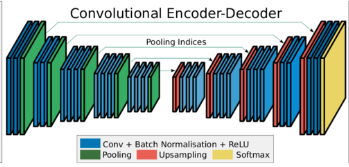
**1.1 Common Deep Network Architectures**

Several deep convolutional networks have made significant contributions to the field of semantic segmentation . The following section is the review of some of these network approaches.

*1.1.1 SegNet*

# An illustration of the SegNet architecture is shown in Figure 1. SegNet has two main parts an encoder part and a decoder part. The encoder part converts the inputs to low dimensional features by the use of max pooling layers after each convolutional layer. A decoder upsamples its input using the transferred pool indices from its encoder to produce a sparse feature map(s). It then performs convolution with a trainable filter bank to densify the feature map. The final output is fed to a softmax classifier for pixel-wise classification.

# SegNet disregards the fully connected layers which reduces the number of parameters. As a result the Segnet encoder network is significantly smaller and easier to train than many other recent architectures.



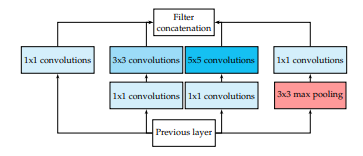
*1.1.2 Deep Residual U-Net*

This is a semantic segmentation neural network which combines the concepts of both the U-Net and residual neural network. It consists of a series of stacked residual units instead of plain neural units as basic blocks. The encoder, bridge and the decoder part of the network are built with residual units which consist of convolution blocks and an identity mapping. The identity mapping connects input and output of the unit. Batch Normalization layers are used to make the model converge faster.

This network addresses the problem of deeper networks with vanishing gradients. The integration of skip connections within the network facilitates information propagation without degradation in addition to less parameters and better performance.

*1.1.3 GoogleNet*

It is a 22 layer network based on the NiN (Network in Network) approach. Its main focus is the problem of trade

off between network size and computational cost. It reduces the number of computations and thus is computationally cheaper than other networks of similar sizes. The architecture is based on blocks of layers called inception modules. But the inception module is different from a sequential block of layers. In an inception module, the input is passed on to different paths and then results of all paths are concatenated or stacked. Each path can have convolutional layers, max pooling layers etc. Each path can be thought of as a mini network, hence Network in Network. 

GoogleNet is composed of these kind of inception modules placed sequentially after one another. The benefit of this kind of architecture is that each layer learns multiple features. So each inception module has one input and one output that goes into another inception module. But some inception modules in the end of the network in GoogLenet have two outputs, one goes into the next inception module while the other goes to a fully connected layer and then a softmax. This softmax predicts the classes at this stage of the network. In this way, predictions at different stages of the network with different number of learned features can be fetched and compared. This also facilitates in analyzing which layers are learning important features essential for predicting classes.

**2. SEMANTIC SEGMENTATION**

*2.1 Naive Segnet*

Segnet architecture is based on an auto encoder. There are two parts of this CNN. First is the encoder part. In this part, there are convolutional layers which are followed by Batch normalization and ReLU activations. After each layer features are downsampled to half by pooling layer. The process is repeated a few times after which features are reduced to resolution 38x38. This completes the encoder part of the network. Then starts the decoder part of the network where each layer is again a convolutional layer followed by Batch Normalization and ReLU. After applying convolutions, features are upsampled to double their size. This part has the same number of layers as the encoder part. It has one layer corresponding to each encoding layer. In the end the output size is the same as the input size but it only differs in the third dimension (number of channels) which is equal to the number of classes (41). Softmax is applied to the final decoder output to convert the features into class probabilities. In this sense, this architecture can be viewed as naive segnet.

**Naive SegNet Architecture**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Layer** | **Filter** | **Stride** | **Output** |
|  | Input |  |  | 150 x 150 x 3 |
| 1 | Conv2D | (3x3)/32 | 1 | 150 x 150 x 32 |
| 2 | Max Pool | 2 x 2 | 2 | 75 x 75 x 32 |
| 3 | Conv2D | (3x3)/64 | 1 | 75 x 75 x 64 |
| 4 | Max Pool | 2x2 | 1 | 38 x 38 x 64 |
| 5 | Conv2D | (3x3)/128 | 1 | 38 x 38 x 128 |
| 6 | Conv2D | (3x3)/128 | 1 | 38 x 38 x 128 |
| 7 | UpSampling 2D | 2x2 | 2 | 76 x 76 x 128 |
| 8 | Conv2D | (3x3)/64 | 1 | 76 x 76 x 64 |
| 9 | UpSampling 2D | 2 x 2 | 2 | 152 x 152 x 64 |
| 10 | Conv2D | (3x3)/32 | 1 | 152 x 152 x 32 |
| 11 | Conv2D | (3x3)/41 | 1 | 150 x 150 x 41 |
| 12 | Softmax |  |  | 150 x 150 x 41 |

*2.2 Short Skip*

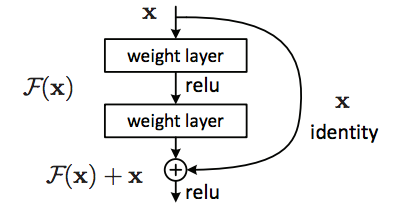
Instead of adding layers to a shallow architecture, the layers can be copied from the learned shallower model, and the added layers are identity mapping. This approach ensures that a deeper model should produce no higher training error than its shallower counterpart.

The identity shortcuts (x) can be directly used when both the input and output are of the same dimensions.

In case the dimensions change the shortcut still performs the identity mapping, with extra zero entries padded with the increased dimension.

The network is designed in blocks where each block has at least a convolutional layer, batch normalization and activation. These blocks are called ‘residual’ or ‘bottleneck’ blocks. Input of each block is connected to the output of each block.

The proposed solution**:**

**

The basic architecture was the same as the naive segnet. Only skip connections were added on top of it.

*2.3 Long Skip*

In an auto encoder or U-net spatial information is lost when images are downsampled using pooling layers. Long skip connections provide a solution to this problem. In the decoder part of the network, at each step after upsampling skip connections are made from the corresponding same sized layers of the encoder part of the network, so that spatial information which was lost during downsampling can be provided to each layer. Features from the encoding path are skipped to the equal resolution features in the decoding part. These connections are commonly known as long skip connections.

The basic architecture was the same as the naive segnet. Only long skip connections were added on top of it.

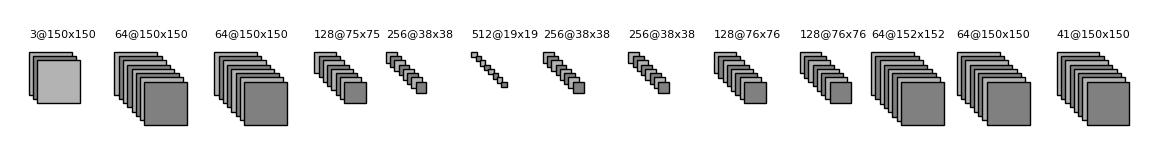
*2.4 Short and Long Skip*

This CNN architecture makes use of both short and long skip connections by combining the previous two architectures. The basic architecture was the same as the naive segnet. Both long and short skip connections were added on top of it.

*2.5 Custom Model*

This model implements both short and long skip connections. The underlying architecture is divided into residual blocks. Each residual block has a convolutional layer followed by ReLU activation and batch normalization. Each residual block’s input is added to the block’s output via a short skip connection. The network is

divided into three parts an encoder, a bridge, and a decoder. It is similar to a U-net. The outputs of the corresponding equal sized features of encoder and decoder are concatenated at the decoder layers before each block to recover lost spatial information. The following figure illustrates the network layers:

****

**Custom Model Architecture**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **Unit Level** | **Conv Layer** | **Filter** | **Stride** | **Output Size** |
| **Input** |  |  |  |  | 150 x 150 x 3 |
| **Encoding** | Level 1 | Conv 1  Conv 2 | 3 x 3/64  3 x 3/64 | 1  1 | 150 x 150 x 64  150 x 150 x 64 |
| Level 2 | Conv 3 | 3 x 3/128 | 1 | 75 x 75 x 128 |
| Level 3 | Conv 4 | 3 x 3/256 | 1 | 38 x 38 x 256 |
| **Bridge** | Level 4 | Conv 5 | 3 x 3/512 | 1 | 19 x 19 x 512 |
| **Decoding** | Level 5 | deconv1  deconv2 | 3 x 3/256  3 x 3/256 | 1 | 38 x 38 x 256  38 x 38 x 256 |
| Level 6 | deconv3  deconv4 | 3 x 3/128  3 x 3/128 | 1 | 76 x 76 x 128  76 x 76 x 128 |
| Level 7 | deconv5  deconv6 | 3 x 3/64  3 x 3/64 | 1 | 152 x 152 x 64  150 x 150 x 64 |
| **Output** |  | deconv7 | 1 x 1/41 | 1 | 150 x 150 x 41 |

**3. EXPERIMENTS**

In this section, we test the model on Deep Drive data set and perform an analysis on the importance of different network architectures.

**3.1 SETUP: SUMMARY OF DATASET**

Deep Drive training data consists of 1000 images (1280 × 720 pixels).We converted the images to size 150 x 150 pixels for our network. The test set is another set of 200 images. Throughout the experiments, we used 800 images for training and 200 images for validation.For each training run, the model version with the best validation accuracy was stored and evaluated.

**3.2 COMPARISON B/W METHODS**

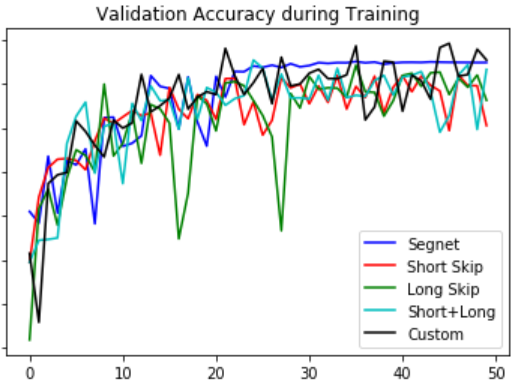
The detailed description of the highest performing architecture used in the experiments is shown in following table:

Batch size = 10

Epochs = 50

loss = Categorical cross entropy

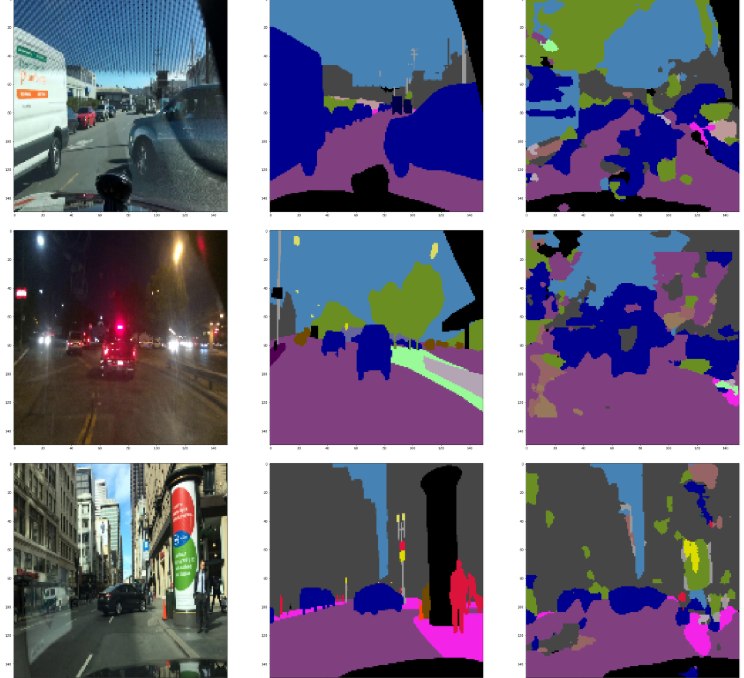
|  |  |  |
| --- | --- | --- |
| **Method** | **Parameters** | **Accuracy** |
| Naive Segnet | 345,915 | 72.5 % |
| Short Skip | 72,123 | 71.3 % |
| Long Skip | 449,883 | 72.1 % |
| Short and Long | 513,083 | 72.6 % |
| Custom | 5,227,305 | 88.7 % |



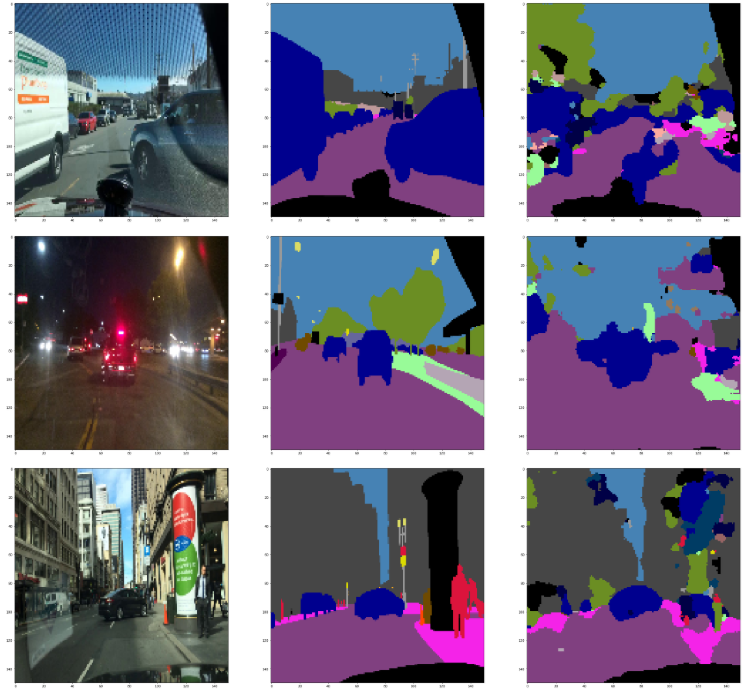
As it is visible from the graph, Custom model has the highest validation accuracy during training.

**Results**

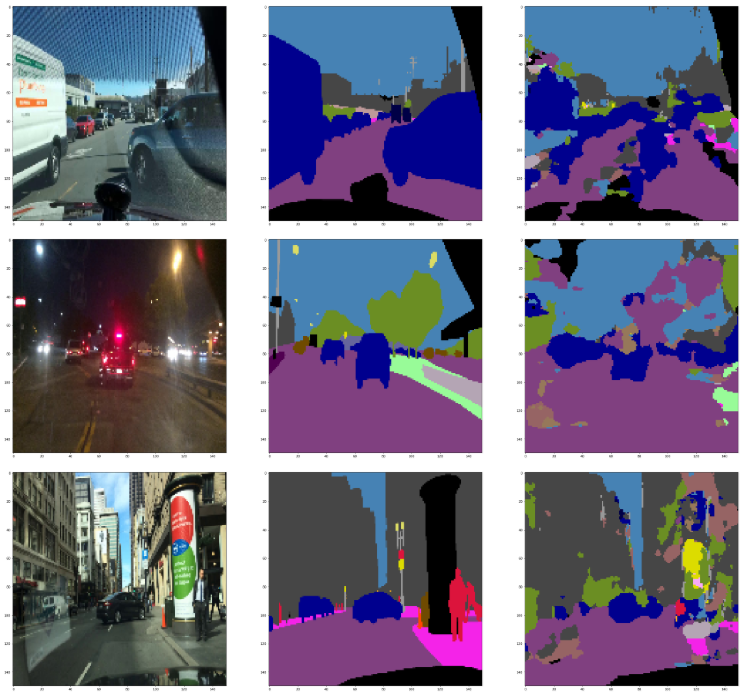
***Segnet***

**

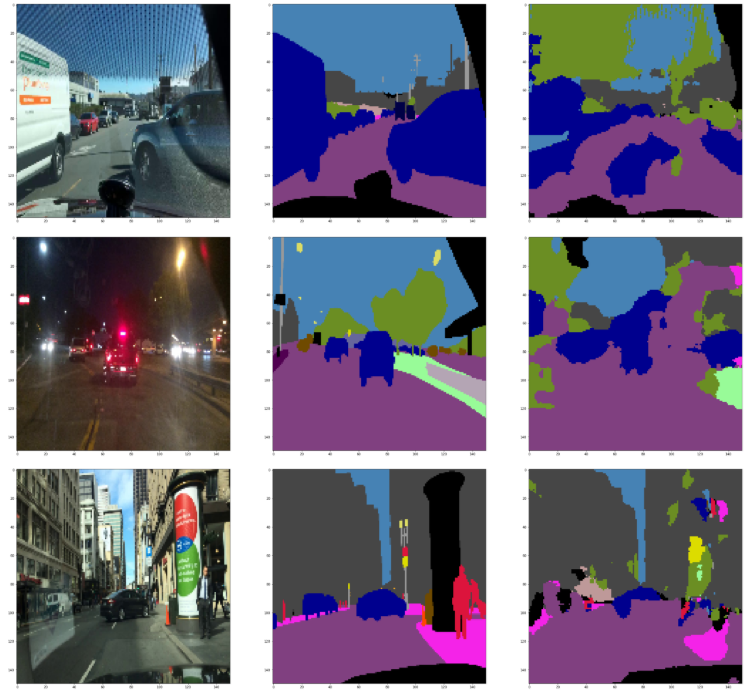
***Short Skip***

**

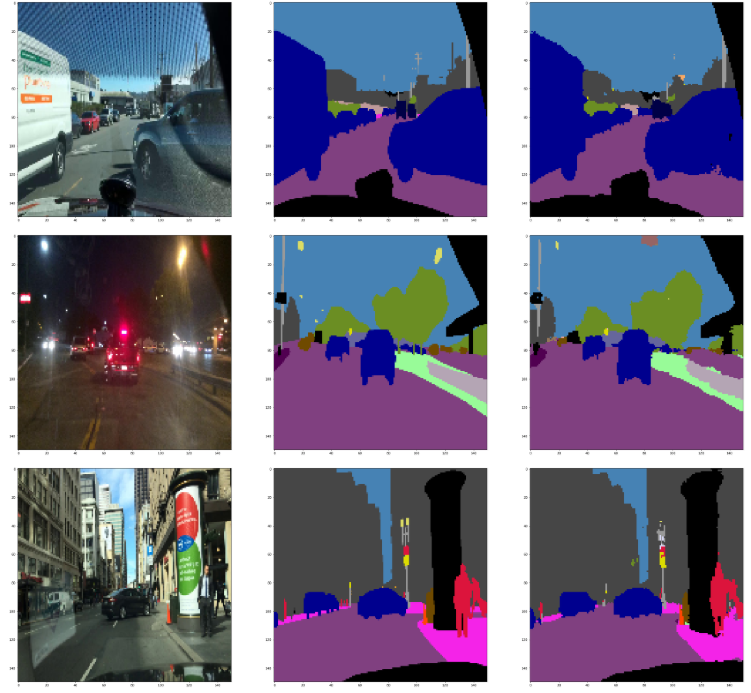
***Long Skip***

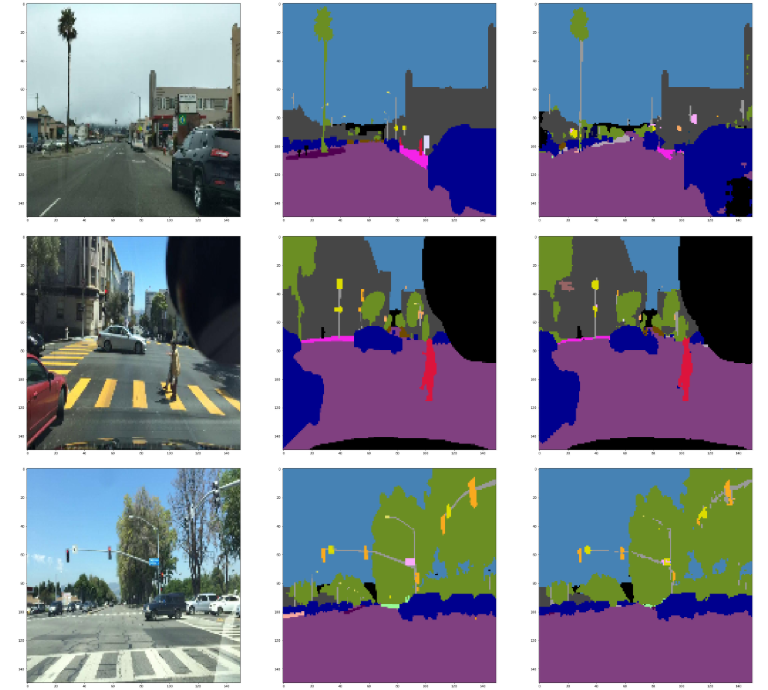
**

***Short and Long***

**

***Custom***





**4. CONCLUSION**

Through the experiments in this paper, we observed the importance of various network architectures in classifying the images for deep drive data set image segmentation. It is observed that although increasing the depth of the convolutional neural network can boost the performance of the network. However these networks results in the problem of vanishing gradients which can be solved by integrating skip connections within these networks. We combined the ideas of both short and long skip connections in our network and our experimental results demonstrate the effectiveness of our framework since the network is able to converge much faster with greater accuracy.

**5. REFERENCES**

[1] J. Long, E. Shelhamer, and T. Darrell, “Fully convolutional networks for semantic segmentation,” in CVPR, 2015, pp. 3431–3440.

[2] O. Ronneberger, P. Fischer, and T. Brox, “U-net: Convolutional networks for biomedical image segmentation,” in MICCAI, 2015, pp. 234–241