# IMAGE SEGMENTATION WITH U-NET

**SRAVYA KAKUMANI, MOHANA BHAVANI TATIKONDA**
Department of Industrial Engineering, University of South Florida,TL

## Abstract

With applications ranging from remote sensing and driverless vehicles to medical imaging, image segmentation is a critical task in computer vision.It ultimately comes down to segmenting an image into meaningful pieces so that computers can comprehend what's inside. In this project. we use the U-Net architecture to perform image segmentation on an automotive driving dataset and analyse why it is better over fully connected network. This dataset contains 1,060 RGB images of driving scenes, along with their corresponding masked images. The masks indicate which parts of the scene are roads, vehicles, pedestrians, and other important elements.U-Net has encoder-decoder structure and skip connetions , which preserve high-resolution information throughout segmentation, making it ideal for this task. We train the U-Net model on this dataset to accurately identify and segment different components of driving scenes.Our goal is to improve the model's ability to recognize various elements in complex driving environments, which is key for applications like autonomous vehicles and advanced driver-assistance systems (ADAS).

## 1.Introduction

Accurate interpretation of the surrounding environment is crucial in the transition to fully autonomous vehicles. This can be made possible through segmentation, which gives automated systems a thorough analysis of an image so they can decide what to do. For instance, by segmenting an image, a car's on onboard system can identify where the road is, recognize other vehicles, and detect pedestrians crossing the street. This level of understanding is essential for ensuring safe and reliable autonomous navigation, as well as for improving advanced driver-assistance systems (ADAS).In this project, we focus on semantic image segmentation, an approach that assigns a label to each pixel in an image to classify it into a specific predefined category. This produces precise masks by identifying each and every pixel in the image, in contrast to standard object detection, which uses bounding boxes that may incorporate excess background or overlap with other objects. This degree of accuracy makes it possible to clearly grasp images, which is essential for advanced driver-assistance systems (ADAS) and autonomous driving.

We employ U-Net, a Convolutional Neural Network (CNN) renowned for its accuracy and efficiency in segmentation tasks, to accomplish this task . The encoder-decoder architecture and skip connections of U-Net help in maintaining high-resolution data during the procedure, allowing the network to predict a label for every pixel.board system can identify where the road is, recognize other vehicles, and detect pedestrians crossing the street. These understandings are essential for trustworthy and secure autonomous navigation.

We use CameraRGB and CameraMask datasets which consists of 1,060 RGB photos of different driving scenarios and masked images associated with it respectively. The masked images indicate which parts of the scene represent pedestrians, cars, roadways, and other important components. By training the U-Net model on this dataset, we aim to achieve accurate semantic segmentation that can help improve the safety andreliability. .

## 1.Related Work

[1] Jonathan Long, Evan Shelhamer, and Trevor Darrell - "Fully Convolutional Networks for Semantic Segmentation":

In their pivotal paper "Fully Convolutional Networks for Semantic Segmentation," Long, Shelhamer, and Darrell introduced Fully Convolutional Networks (FCNs) as a breakthrough in the field of semantic segmentation. Their research demonstrated how FCNs could be trained end-to-end, pixel-to-pixel, outperforming traditional methods. The FCNs adapted existing classification networks like AlexNet, VGG Net, and GoogLeNet, transforming them for segmentation tasks. This work set a new standard, achieving state-of-the-art results on challenging datasets like PASCAL VOC, without the need for complex post-processing.

[4] Marius Cordts et al. - "The Cityscapes Dataset for Semantic Urban Scene Understanding":

"The Cityscapes Dataset for Semantic Urban Scene Understanding," Cordts and his team developed a large-scale dataset tailored for urban scene understanding, enabling advancements in autonomous driving applications. The dataset contains over 5,000 finely annotated images and 20,000 coarsely annotated images from 50 cities, providing rich scene variability and complexity. This dataset, along with their benchmark suite, has become a critical resource for training and testing pixel-level and instance-level semantic labeling methods, influencing future research in the field. The accompanying empirical study offered insights into the dataset characteristics and evaluated state-of-the-art approaches. They analysed the performance of these dataset using FCN architecture and compared it's performance with different filter values.

## 2.Method

U-Net, a highly utilized deep learning framework, was initially presented in the research paper titled "U-Net: Convolutional Networks for Biomedical Image Segmentation." Its main objective was to tackle the issue of scarce annotated data in medical applications. The network was crafted to efficiently utilize a reduced data set while upholding both swiftness and precision.U-Net builds upon the Fully Convolutional Network (FCN) architecture, which replaces dense layers with transposed convolutions for up sampling, preserving spatial information crucial for image segmentation tasks. Unlike dense layers, which can destroy spatial details, transpose convolutions allow for flexible input sizes. However, FCN's final feature layer suffers from significant information loss, resulting in rough outputs after up sampling. U-Net improves upon FCN by employing a similar design but with multiple convolutions for down sampling and transposed convolutions for up sampling. Additionally, U-Net integrates skip connections to retain information lost during encoding, preventing information loss and model over fitting.
The structure of U-Net is distinctive in that it comprises a contracting path and an expansive path. The contracting path includes encoder layers responsible for capturing contextual details and diminishing the spatial resolution of the input. Conversely, the expansive path incorporates decoder layers, which decipher the encoded information and utilize data from the contracting path through skip connections to produce a segmentation map.

**Model Architecture:** U-net model used here is implemented as follows:

*Contracting Path:*The contracting path, or encoding path, compresses the input data while extracting key features. It starts with the input layer and advances through a series of convolutional blocks (conv_block). The first block is given 32 filters in this case. The subsequent blocks double the number of filters at each step, reaching 64 filters in the second block, 128 filters in the third, and 256 filters in the fourth.
Each convolutional block contains two convolutional layers with ReLU activation, "same" padding, and a "he_normal" initializer for the kernel weights. After each block, a max-pooling layer halves the spatial dimensions, allowing the model to focus on higher-level features.
To prevent overfitting, a dropout layer with a dropout rate of 0.3 is added starting from the fourth convolutional block. In the final block of the contracting path, dropout is used again, and max-pooling is

turned off, signaling the transition to the expanding path. This last block uses 512 filters, providing a high level of abstraction for the model's bottleneck.

*Expanding Path:*The expanding path, or decoding path, restores the spatial dimensions through transposed convolutions, commonly known as deconvolutions. This path aims to reconstruct high-resolution outputs by combining compressed features with high-resolution information from the skip connections.
Each step in the expanding path involves an upsampling operation (upsampling_block) followed by concatenation with the corresponding skip connection from the contracting path. The upsampling blocks work with decreasing numbers of filters, reflecting the inverse of the contracting path: 256, 128, 64, and finally 32 filters at the last step.

*Final Convolutional Layer:*
After the expanding path, there is a Conv2D layer (conv9) with 32 filters, ReLU activation, and "same" padding. The final Conv2D layer (conv10) has one filter for each of the 23 defined classes , with a kernel size of 1 and "same" padding to maintain the original spatial dimensions.

*Model Output:*
The final output of the unet_model is a multi-class segmentation map where each pixel is assigned to one of the specified classes. This structure makes the model suitable for various applications requiring high-resolution segmentation, such as medical imaging, satellite imagery, and autonomous vehicle scenarios. The U-shaped design, combined with skip connections and effective dropout usage, contributes to robust and accurate results. The result of plot_model is depicted in fig1

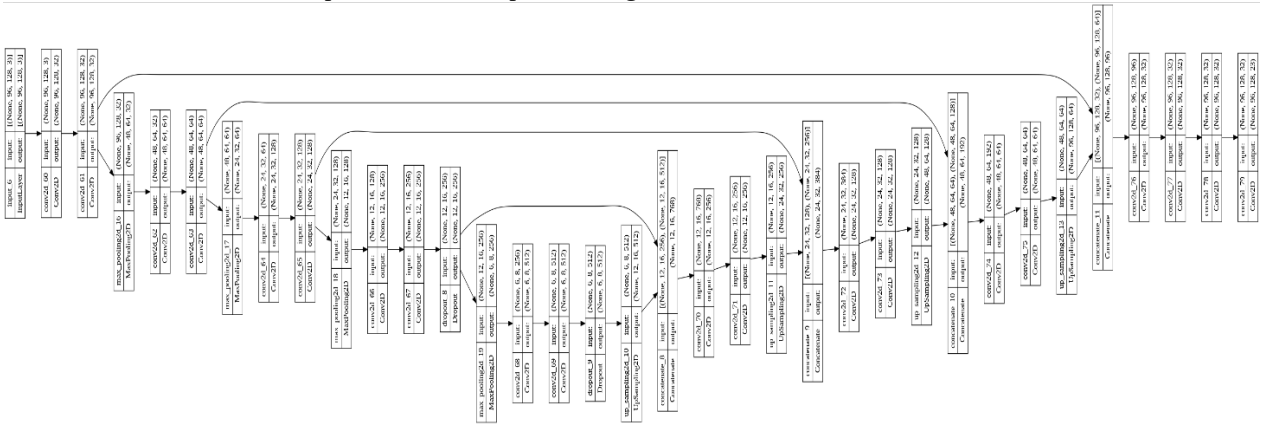


Fig1: Result of plot_model for U-Net

The model is then compiled with Sparse Categorical Cross-Entropy as the loss function and Adam as the optimizer.

*Loss Function:*
Sparse categorical cross entropy is a loss function commonly used in multi-class classification tasks where each example belongs to exactly one class. It's particularly suited for scenarios where the classes are mutually exclusive. The formula for sparse categorical cross entropy loss is derived from the cross entropy between the true distribution and the predicted probabilities. Formula or this loss function is depicted in Fig2.

$$L\left(y,\hat{y}\right) = -\frac{1}{N}\sum_{i}^{N}\sum_{j}^{C} y_{ij} log\left(\hat{y}_{ij}\right)$$

Fig2: Formula for Sparse categorical cross entropy

*Adam Optimizer:*
Adam optimizer combines elements of RMSprop and stochastic gradient descent with momentum. It adapts the learning rate for each parameter by utilizing estimates of the first and second moments of the gradient.

These moments are computed using exponentially weighted moving averages of the gradients evaluated on the current mini- batch. The algorithm initializes moving average vectors with zeros and then updates them iteratively based on the gradients. However, because these averages are initialized with zeros, they are biased towards zero. To correct this bias, Adam applies bias correction to the estimators of the first and second moments. Finally, it scales the learning rate for each parameter using the corrected moment estimates, resulting in individual learning rates tailored to each parameter of the neural network. This adaptive learning rate method enables efficient optimization of neural network parameters, making Adam a popular choice for training deep learning models. Formula or this loss function is depicted in Fig3.

$$w_t = w_{t-1} - \eta \frac{\hat{m}_t}{\sqrt{\hat{v}_t} + \epsilon}$$

Fig3: Formula for Adam Optimizer

## 3.Experiments

*Dataset:* We use a dataset called "CameraRGB," consisting of 1,080 images depicting various scenarios a car might encounter while driving. These images serve as the primary input for our semantic segmentation tasks. Alongside this, we have the "CameraMask" dataset, containing corresponding masked images where each pixel is labeled according to its class, such as "Car," "Person," or other categories.Semantic image segmentation allows us to predict a detailed mask for each object in an image by assigning a specific class to each pixel. The term "semantic" refers to the meaning or context behind the objects in the image. For example, in this type of segmentation, a "Car" might be represented by a dark blue mask, while a "Person" is depicted with a red mask. These color-coded masks help visualize and distinguish different objects within the same image, providing a clear understanding of the scene's components.Fig 4,shows an example of CameraRGB dataset and it's corresponding colored CameraMask

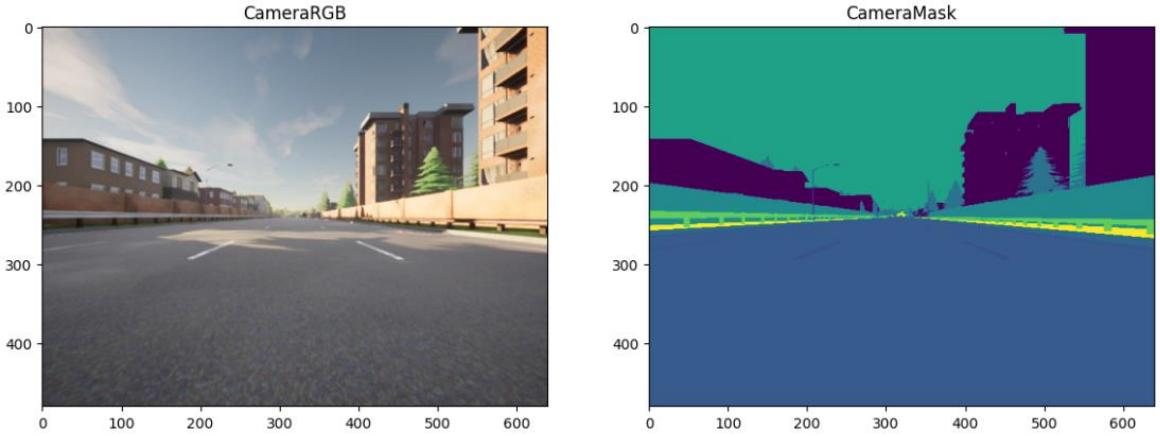



Fig 4: An example from the dataset

we've prepared separate folders named "train" and "test," each containing both images and masks. The training set, comprising 80% of the data, will be utilized to train the model, enabling it to learn the underlying patterns in the data. Meanwhile, the testing set, constituting 20% of the data, will serve as an independent validation source to gauge the model's performance on new, unseen examples.

*Data preprocessing:* The data preprocessing process begins by identifying the locations of directories containing RGB images and their corresponding segmentation masks. After obtaining the list of file names for both images and masks, these are converted into a format suitable for TensorFlow's dataset processing. The first step in processing involves reading the image and mask files from their respective directories. The images are then decoded from their original PNG format and converted to a floating-point representation to standardize the data, with the pixel values normalized to a range between 0 and 1. This

normalization is crucial for training machine learning models, as it ensures consistency across the dataset. Next, the segmentation masks are adjusted to create a single-channel representation. This is achieved by extracting the maximum value across the color channels, simplifying the masks into a format suitable for segmentation tasks where distinct regions are defined by unique pixel values.

*Training Curves**:***
The processed image dataset is batched with a size of 32, allowing the model to process multiple samples simultaneously, which improves efficiency during training. The dataset is cached to reduce disk reads, speeding up the training process, and shuffled with a buffer size of 500 to randomize the order of the data, ensuring a varied and unbiased training experience. This shuffling is key for the model to generalize well, as it provides a diverse set of examples during training. Once the training dataset is prepared, the model is trained using the fit () method, with the training data as the input and a total of 40 epochs. This training process involves 40 complete cycles through the dataset, which helps the model learn and refine its weights over time. U-Net and FCN are trained with the same parameters and hyperparameters, allowing a direct comparison of their loss during training**.** Their training curves where the accuracy is depicted for various epoch is shown in Fig 5. By comparing the figures, It can be seen that U-Net converges faster and achieves higher final accuracy compared to FCN in image segmentation tasks, owing to its use of skip connections and symmetric architecture. FCN, while simpler, may require more epochs to reach comparable accuracy and might not achieve the same level of precision due to its design.

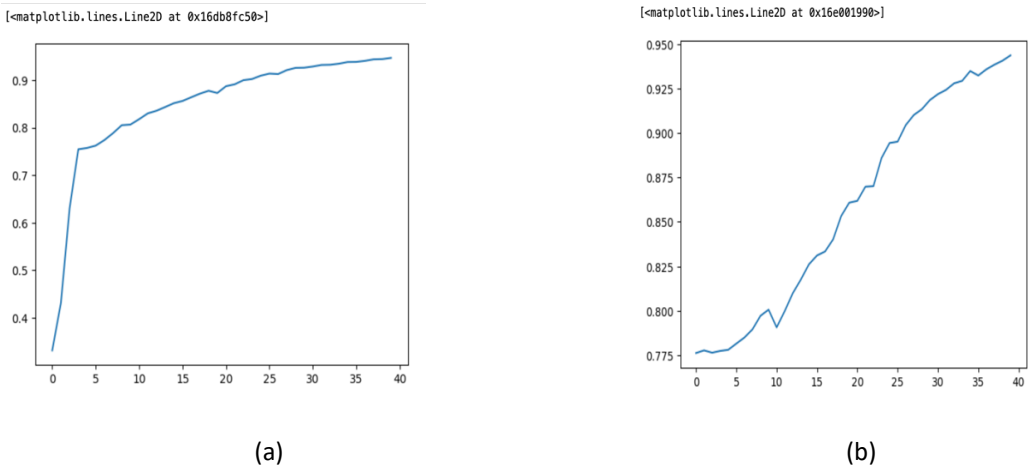



Figure 5: Training curve for image segmentation using (a) U-net and (b) FCN.

*Results:* Input image, true mask and predicted mask of the system trained with U-Net are shown in Figure 6.

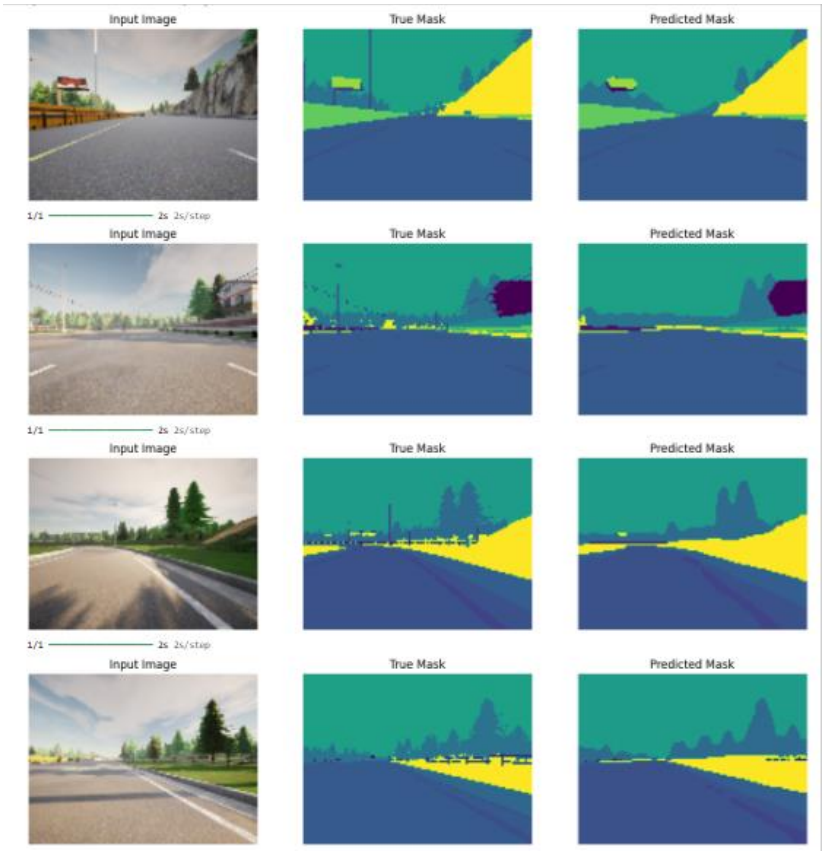



Fig 6: predicted masks

***Ablation Study*:**

1.*Testing accuracy*: U-Net gives higher testing accuracy than FCN due to its skip connections and symmetric architecture , which helps it to get high spatial-information .The testing accuracy of U-net and FCN are 93.33% and 90.14% respectively as shown in Fig 7 and Fig 8.

```
In [14]: test_dataset = test_processed_image_ds.cache().batch(32)
         test_loss, test_accuracy = unet.evaluate(test_dataset)

         print("Test Loss:", test_loss)
         print("Test Accuracy:", test_accuracy)

         7/7 ━━━━━━━━━━━━━━━━━━━━ 4s 457ms/step - accuracy: 0.9316 - loss: 0.2287
         Test Loss: 0.22409619390964508
         Test Accuracy: 0.9333934187889099
```

Fig 7: Test Accuracy of U-Net

```
: test_dataset = test_processed_image_ds.cache().batch(32)
  test_loss, test_accuracy = fcn.evaluate(test_dataset)

  print("Test Loss:", test_loss)
  print("Test Accuracy:", test_accuracy)

  7/7 ━━━━━━━━━━━━━━━━━━━━ 2s 262ms/step - accuracy: 0.9000 - loss: 0.3223
  Test Loss: 0.31653696298599243
  Test Accuracy: 0.9014189839363098
```

Fig 8: Test accuracy of FCN

2.*dropout*: Testing accuracy of U-Net without dropout is 89.4% in contrast to Fig 7. This is because adding dropout improve generalization and reducing overfitting. The training accuracy might be lower due to the dropout, but the validation accuracy improves indicating a better ability to generalize to new data.

3.*filters*: U-net with 32 filters gives the upper bound training accuracy to 93.2% as shown in Fig9, in-contrast to 64 filters as shown in Fig 5(a),where the upper bound of training accuracy is 94.56%.This is because Increasing filters leads to improved accuracy and better segmentation, as the model can capture more detailed features. However, this comes at a cost: increased computational complexity, longer training times, and greater memory requirements. Running 40 epochs for this task takes is quiet hectic.

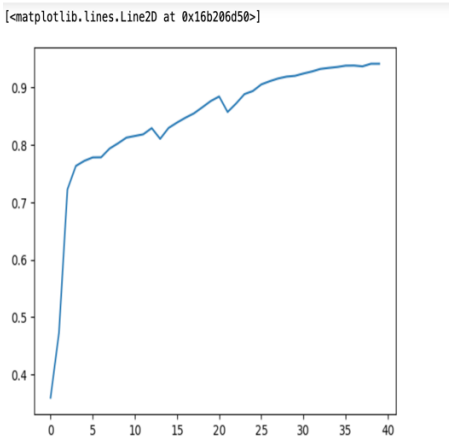
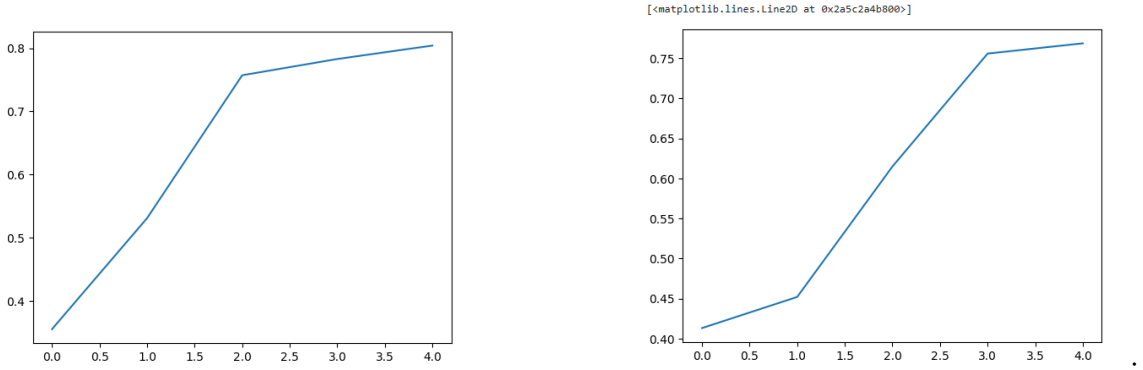



Fig 9:Traing accuracy of U-Net with 32 filters

4.*epochs*: The number of epochs is a critical parameter in training image segmentation models. It determines the training duration and impacts how well a model learns to segment images while balancing the risk of over-fitting . From the Fig 8, we can depict that u-net accuracy reaches above 80% even for 5 epochs, Whereas FCN only reaches 75% for the same 5 epochs. U-Net converges faster thus required less training to reach the required accuracy ,but FCN need to be trained more .However the simple structure make FCN less prone to overfitting even with higher epochs .

Training curvers for U-Net and FCN for 5 epoch is depicted in Fig 10.




;

a b

Fig 10:Training curver for a)U-net and b)FCN for 5 epoch

## 4.Conclusion and future scope:

In conclusion, the U-Net architecture has demonstrated remarkable performance in image segmentation tasks, outperforming traditional methods and even contemporary architectures like Fully Convolutional Network (FCN) in various scenarios. Its superiority lies in several key factors Firstly, U-Net's use of skip connections seamlessly integrates multi-scale features, allowing the model to maintain fine details and spatial context crucial for accurate segmentation. This design, along with its symmetric encoder-decoder structure, ensures efficient information flow and precise object localization.

Additionally, U-Net's feature fusion abilities through concatenation-based skip connections enhance its capability to capture complex patterns and structures, leading to more accurate segmentation. This makes U-Net particularly effective in tasks with intricate object boundaries or small objects.

Moreover, U-Net's robust performance is supported by effective training strategies such as data augmentation, transfer learning, and regularization techniques. These methods not only prevent overfitting but also enhance the model's generalization performance across diverse datasets and real-world scenarios.

Looking forward, U-Net's impressive performance holds significant promise for various applications beyond image segmentation. In the realm of autonomous driving, where precise scene understanding is crucial for safe navigation, U-Net's capabilities can be harnessed to develop advanced perception systems. By accurately identifying objects and obstacles in real-time, U-Net-powered models can contribute to the creation of autonomous vehicles that improve road safety and reduce accidents.

.

## References


[1] Jonathan Long, Evan Shelhamer, and Trevor Darrell , “Fully Convolutional Networks for Semantic Segmentation “ ,2014

[2]Bhakti Baheti et al., "Eff-UNet: A Novel Architecture for Semantic Segmentation in Unstructured Environment", In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops, June 2020.

[3]Aysęgül Ucąr Cąğrı Kaymak, A Brief Survey and an Application of Semantic Image Segmentation for Autonomous Driving, Cham:Springer, pp. 161-200, 2019.

[4]Marius Cordts et al., "The Cityscapes Dataset for Semantic Urban Scene Understanding", arXiv: 1604.01685, 2016.

[5]Gerhard Neuhold et al., "The Mapillary Vistas Dataset for Semantic Understanding of Street Scenes", In: International Conference on Computer Vision (ICCV), 2017, [online] Available: https://www.mapillary.com/dataset/vistas.

[6]Apoorva Ojha, Satya Prakash Sahu and Deepak Kumar Dewangan, "Vehicle Detection through Instance Segmentation using Mask R-CNN for Intelligent Vehicle System", In: 2021 5th International Conference on Intelligent Computing and Control Systems (ICICCS), pp. 954-959, 2021.

[7]Vijay Badrinarayanan, Alex Kendall and Roberto Cipolla, "Segnet: A deep convolutional encoder-decoder architecture for image segmentation", IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 39, pp. 2481-2495, 2016.

[8]G. Varma, A. Subramanian, A. M. Namboodiri, M. K. Chandraker and C. V. Jawahar, "Idd: A dataset for exploring problems of autonomous navigation in unconstrained environments", 2019 IEEE Winter Conference on Applications of Computer Vision (WACV), pp. 1743-1751, 2018.

[9]L. Chen, Y. Zhu, G. Papandreou, F. Schroff and H. Adam, "Encoder-decoder with atrous separable convolution for semantic image segmentation" in CoRR, 2018.

[10]https://scholar.archive.org/work/p4lbi4ve65dnjoal5pejnfr75u/access/wayback/https://dergipark.org.tr/tr/download/article-file/1307413
aDepartment of Electrical and Electronics Engineering, Van Yuzuncu Yil University, Van, TurkeybSedad Pars Technology, Urmia, Iran
Volume 08,Issue 03September, 2020