

Semantic Segmentation

Aniket Bagaitkar
Department of Computer Science
College of Arts and Science
University of Alabama at Birmingham
Birmingham, United States
aniketb@uab.edu

Sagar Panchal
Department of Computer Science
College of Arts and Science
University of Alabama at Birmingham
Birmingham, United States
sagarp95@uab.edu

Saylee Raut
Department of Computer Science
College of Arts and Science
University of Alabama at Birmingham
Birmingham, United States
saylee30@uab.edu

Varun Guvvala
Department of Computer Science
College of Arts and Science
University of Alabama at Birmingham
Birmingham, United States
varun505@uab.edu

Abstract— This document provides an overview of Deep learning techniques for semantic segmentation are reviewed and applied in a wide variety of applications. We show that without any additional equipment, a fully convolutional network (FCN) trained final stage, pixels-to-pixels on semantic segmentation outperforms the state-of-the-art. and Semantic segmentation or scene comprehension are examples of fields or applications related to computer vision.

Keywords— FCN, colonies, segmentation, deep learning, fully convolutional network, and Segnet, pixels-to-pixels.

I. INTRODUCTION

Researchers in computer vision and machine learning are becoming increasingly interested in image semantic segmentation. Many new applications, such as autonomous vehicles and navigation systems, require precise and reliable segmentation algorithms. To mention a few, virtual or augmented reality systems. This need is occurring at the same time as deep learning technologies are becoming more prevalent in practically every industry to our understanding, this is the first study to train FCNs from controlled pre-training to pixelwise prediction. Existing layers' fully convolutional versions anticipate dense outputs from arbitrarily defined inputs. First check all segmentation masks. Then function uses opencv rather than PIL to load image. Once the images are load, we need to Visualize a random file then Load the data and creates a tuple of images. Further masks to be later made into Tensorflow data object. Resize mask and image to common size and Resize Image to 256x256. The graphic is the very first layer, with pixel sizes of h w and d channels. The receptive fields of higher layers approximate to the locations in the image to which they are route. Convents are fundamentally non-translatable. Convolution, pooling, and training algorithm work on enhanced local regions and are only dependent on relative grid points. A web with simply layers of this structure figures a regressive channel, which we term a thorough channel or totally modeling technique, whereas an overall net registers an overall nonlinear capacity. An FCN analyzes every size contribution and returns a set of related spatial measures. Such coarse yields must be linked back to

the pixels. The notion that an expanding range of applicants rely on insinuating knowledge from imagery emphasizes the importance of scene understanding as a key computer vision impairment. Driverless cars, human contact, , image search engines, and virtual reality are just a few of the uses. Various traditional computer vision and machine learning techniques have been used to solve this challenge in the past. Despite their prominence, the deep learning movement has turned the tables, and many machine vision issues, including semantic segmentation, are now being addressed using deep architectures, most commonly Convolutional Neural Networks (CNNs)

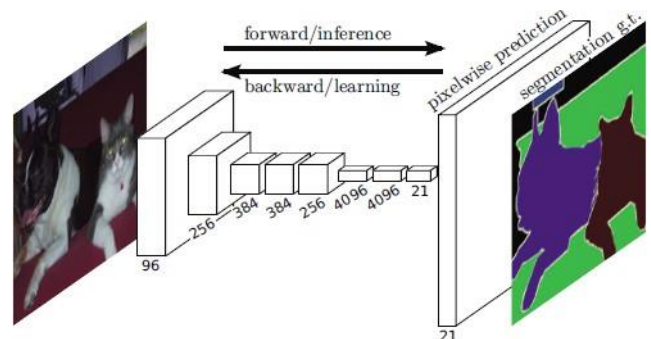
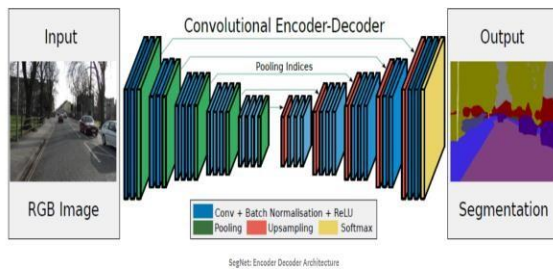


FIG 1: CONVOLUTIONAL NEURAL NETWORK

**Input → Convolution + Relu → Pooling → Convolution
+ Relu → Pooling → Flatten → Fully Connected →
Softmax.**

Models implemented in the paper

1. SEGNET

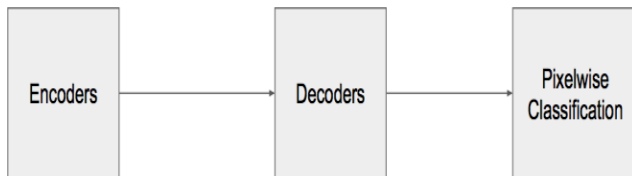


SegNet is an image segmentation architecture that combines an encoder-decoder design.

Architecture

The encoder element of this approach is based on a pre-trained VGG16 model. VGG16 is utilized to create all 13-convolution operation. There is a comparable decoder network for each encoder layer that up samples the image to its original size. The decoder network is proceeded by a pixelwise classification model, the SoftMax unit.

Encoder-Decoder pairs are used to create feature maps for classifications of different resolutions.

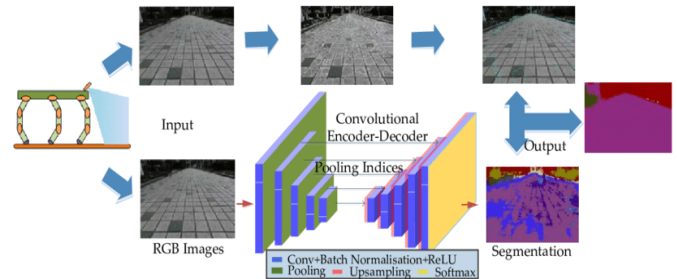


Encoder – VGG16 Conv layers 13 Encoder This decreases factors from 134M to 14.7M because they are not fully connected. Because good beginning weights are available, certain layers have been rendered untrainable. The uniqueness comes in the subsampling stage, where Max-pooling is utilized to achieve satisfactory performance over tiny spatial movements in the image, and Techniques for large is being used to give each pixel control over a broader input image context (spatial window). These methods improve accuracy of classification while reducing feature map size, resulting in lossy input images with blurring boundaries, which is ineffective for segmentation. It is required that the output resulting image be the same as the input image. To accomplish this, SegNet uses Upsampling in its decoder, which necessitates the storage of some data.

Decoder-Higher-resolution extracted features with sparse features were created. There is a decoder for every one of the 13 encoders, which upsamples the image representation using max-pooling indices that are memorized. To create dense image features, sparse maps are processed through a learnable filter bank. The final decoder is linked to a softmax classifier, which assigns a classification to each pixel.

Adam Optimizer-

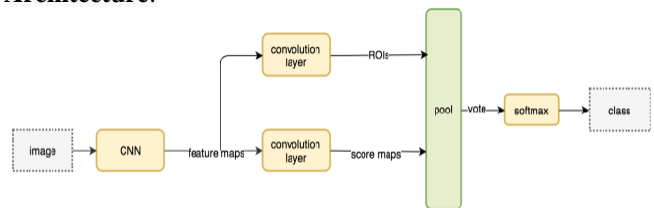
The Adam Optimizer is a strategy for adjusting learning rate based on squared gradients. Because it is an adaptive learning rate approach, it calculates particular learning rates for various factors. The Adam optimizer is the foundation for future model adjustment, and the basic variable definitions are listed below.



2.FCN

The FCN model retrieves image features using a CNN, then employs a 1111 convolutional layer to transform the number of channels into the number of classes, and then utilizes transposed convolution to translate the height and width of the feature maps to those of the input picture. As a result, the model output is the same height and breadth as the input image, with the predicted classes for the input pixel at the same spatial location in the output channel. A convnet's data layers are each a three-dimensional array of size $h \times w \times d$, with h and w being spatial dimensions and d being the feature or channel dimension. The image is the first layer, with pixel sizes of $h \times w$ and color channels of d . Higher-layer locations correlate to their receptive fields, which are the areas in the image to which they are path-connected.

Architecture:



Translation invariance is the foundation of convnets. Convolution, pooling, and activation functions perform on local input regions and are only dependant on relative spatial coordinates. These functions compute outputs y_{ij} by $y_{ij} = f_{ks}(x_{si+i, sj+j} \otimes k)$, where k is the kernel size, s is the stride or subsampling factor, and f_{ks} determines the layer type: a matrix multiplication for convolution or average pooling, a spatial max for max pooling, or an elementwise nonlinearity for an activation function, and so on for other types of layers. Under composition, this functional form is preserved, with kernel size and stride obeying the transformation rule $f_{ks} g_{k_0 s_0} = (f \otimes g)_{k_0 + (k_1)s_0, ss_0}$. A net with only layers of this form computes a nonlinear filter, which we call a deep filter or completely convolutional network, whereas a general deep net computes a generic nonlinear function. An FCN runs on any size input and produces an output with the same

(potentially resampled) spatial dimensions. A task is defined by a real-valued loss function combined with an FCN. The gradient of the final layer will be a sum of the gradients of each of its spatial components if the loss function is a sum over the spatial dimensions of the final layer, $\nabla_{(x_i)} = \sum_j \nabla_{(x_{ij})}$. As a result, stochastic gradient descent on $\nabla_{(x_i)}$ computed on whole images will be the same as stochastic gradient descent on $\nabla_{(x_{ij})}$, taking all of the final layer mini batch images into account.

II. RESULTS

A. FCN –

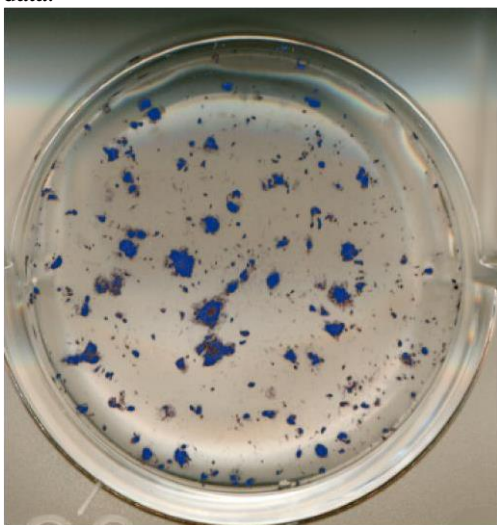
1. Image Object

The models for image classification and object recognition are trained on fixed-size input images.

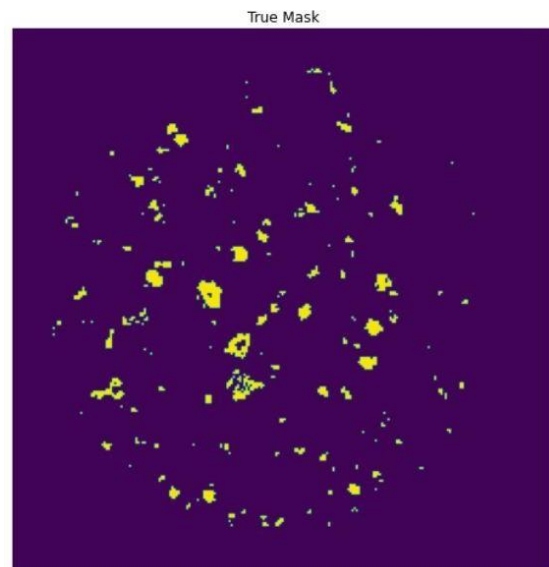
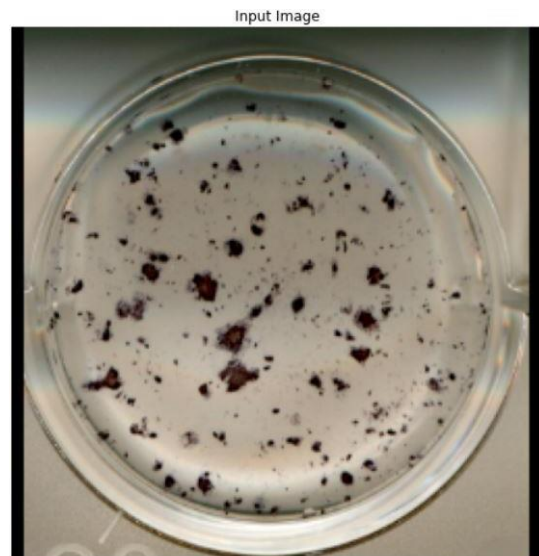


2. Masked Image

The second phase network is a Mask-based object detection network that is trained using the object image masks from the FCN output, the original signal picture, and extra label data.

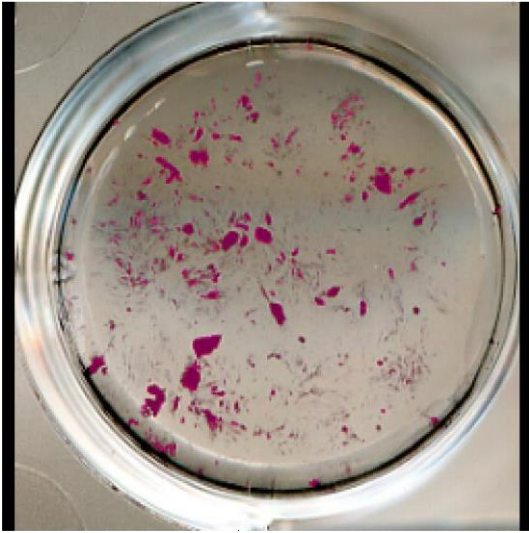


3. True Mask



4. Prediction Mask

FCN performs a final training for picture segmentation prediction. FCN, in a nutshell, eliminates fully-connected layers in favor of convolutional and pooling layers. This tries to encode the input image using a sequence of convolution layers at first. The categorization output of fully-connected layers is now replaced with a pattern of object classes, owing to the replacement of fully-connected layers with convolution layers.



5. Ground Truth

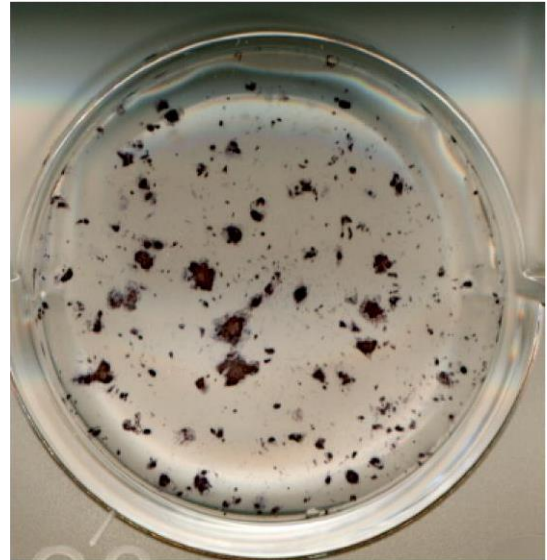
Image data can now be linked to real-world elements and materials. A collection of measurements known to be substantially more precise than data from the system under test is referred to as "ground truth."



B. SegNet –

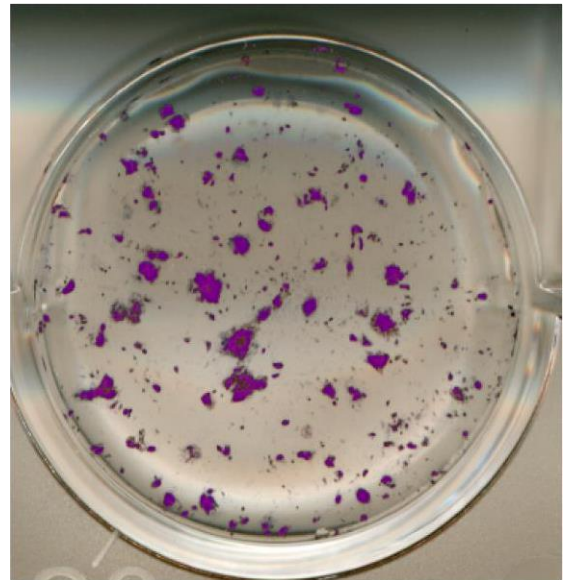
1. Image Object

The models for image classification and object recognition are trained on fixed-size input images.



2. Masked Image

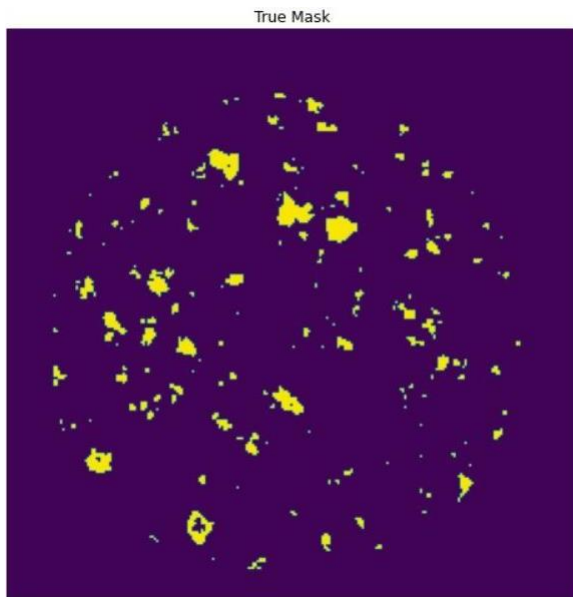
The second phase network is a Mask-based object detection network that is trained using the object image masks from the SegNet output, the original signal picture, and extra label data.



3. Segnet True Mask



True Mask



4. Ground Truth

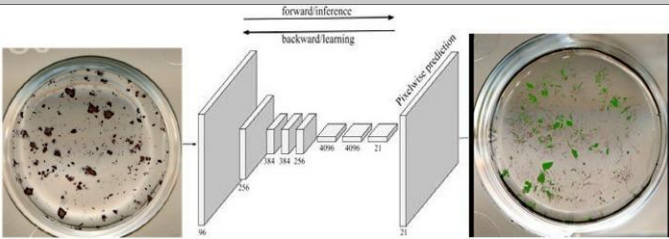
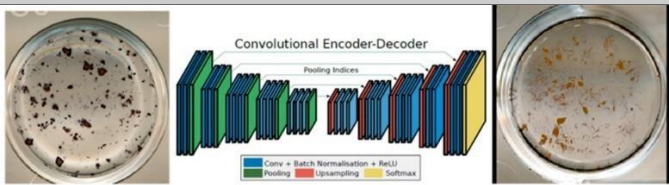
A collection of measurements known to be substantially more precise than data from the system under test is referred to as "ground truth."



III. CONCLUSION

In this paper, we presented two models of Image segmentation: Segnet and Fully convolutional neural networks. Both the models have deep convolutional architecture tested on a large dataset. In Segnet, only the pooling indices are transferred to the expansion path from the compression path, using less memory. In UNet, entire feature maps are transferred from compression path to expansion path making, using much memory. On the contrary, FCN removes any dense (fully connected) layers and only use convolution layers. To get a pixel-wise prediction, downsampling and upsampling are performed. Also, lateral connections are introduced, resulting in downsampling feature maps and upsampling feature maps on the same level. FCN and U-net are very similar. Their core model architecture idea and implementation are nearly identical, with minor differences. All three models including have the same core concept of Upsampling and Downsampling. To list the three in chronological order, it would be FCN better than U-net and Segnet with the least accuracy. The primary difference between FCN and UNet would be the concatenation of layers which done in UNet and not in the FCN.

IV.COMPARISON AND MODEL EVALUATION

Parameters	FCN	SegNet
Define	Fully convolutional networks take input of arbitrary size and produce correspondingly-sized output with efficient inference and learning. An artificial neural network that performs semantic segmentation is known as a completely convolutional network (FCN).	SegNet consists of an encoder network and a decoder network, preceded by a pixelwise classification layer. The encoder network has 13 convolutional layers. At the deepest encoder performance, they discard the completely connected layers in favor of maintaining higher resolution feature maps.
Accuracy	0.9151	0.8283
Loss	0.6931	0.9545
Val Loss	0.6931	0.9477
Val Accuracy	0.9801	0.8296
Comparison		

REFERENCES

1. Hao Wu, Jan Paul Siebert, Xiangrong Xu, March 2020 "Fully Convolutional Networks for Automatically Generating Image Masks to Train Mask R-CNN",
Ref "<https://arxiv.org/abs/2003.01383>".
2. Pytorch Team "Fully-Convolutional Network model with ResNet-50 and ResNet-101 backbones",
Ref "https://pytorch.org/hub/pytorch_vision_fcn_resnet101/"
3. Liang-Chi Hsieh, May 31, 2018 " Comparison between Mask R-CNN and FCN prediction",
Ref <https://medium.com/@viirya/comparison-between-mask-r-cnn-and-fcn-prediction-e2c142a880cd>".
4. Himanshu Rawlani "Understanding and implementing a fully convolutional network (FCN)", Jan 01, 2020
Ref "<https://towardsdatascience.com/implementing-a-fully-convolutional-network-fcn-in-tensorflow-2-3c46fb61de3b>".
5. Jonathan L., Evan S., Trevor D., 2015 – Fully Convolutional Networks for Semantic Segmentation – UC Berkeley –
Ref "<https://arxiv.org/pdf/1411.4038.pdf>".
6. Sik-Ho Tsang, 2018 – Review: FCN — Fully Convolutional Network (Semantic Segmentation)
Ref "<https://towardsdatascience.com/review-fcn-semantic-segmentation-eb8c9b50d2d1>".
7. Chadrick, 2020 - FCN, UNet, FPN comparison
Ref "<https://chadrick-kwag.net/fcn-unet-fpn-comparison/>"
8. Sik-Ho Tsang, 2019- "SegNet(Semantic Segmentation): Encoder Decoder Architecture"-
Ref "<https://towardsdatascience.com/review-segnet-semantic-segmentation-e66f2e30fb96>".
9. Abhishek Kumar, 2020 SegNet- "A Fully Convolutional Network"-
Ref "<https://medium.com/@abhishekkakiak/semantic-segmentation-segnet-a54af19b6d6>".
10. Himanshu Rawlani, 2020- "Understanding and implementing a fully convolutional network (FCN)"-
Ref "<https://towardsdatascience.com/implementing-a-fully-convolutional-network-fcn-in-tensorflow-2-3c46fb61de3b>".