Semantic Segmentation using U-Net

KiranPriya Chigurupati*
Rajat Mehta†
Vignesh Sivakumar‡

Abstract

In this study, we present a simple, flexible and versatile U-Net-based method for image segmentation using the TGS salt identification challenge dataset, which is a dataset used for object detection and segmentation tasks. The U-Net architecture is a convolutional neural network (CNN) that has been proven effective in various image segmentation tasks. Our aim is to accurately and automatically identify whether a subsurface target is composed of salt or not. By leveraging machine learning, we aim to improve the accuracy of seismic images and 3D renderings, and ultimately enhance safety for oil and gas drilling operations. Our proposed approach uses the U-Net model trained on this dataset, with pre-processing to generate input images and ground truth masks. The model is trained using pixel-wise cross-entropy loss for improved robustness. Our experimental results demonstrate that our U-Net-based approach achieves competitive performance on our dataset, with high accuracy and fast inference times.

1 Introduction

Deep learning has been increasingly used in medical image analysis due to advances in computer vision. Over the past two years, deep convolutional networks have surpassed the previous state of the art in various visual recognition tasks. However, challenges remain, and researchers continue to develop novel techniques.

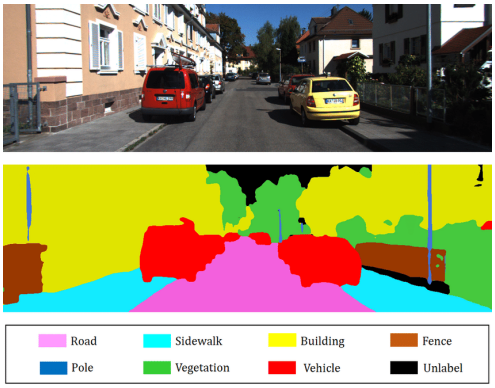

Figure 1: Semantic Segmentation

A deep learning technique that connects a label or category with each pixel in an image is called semantic segmentation [12]. We have a powerful baseline in U-NET and we have a few base references which are (faster-RCNN[11, 4] and mask-RCNN [5]framework) for pursuing outcomes of object detection and semantic segmentation. The U-net, a widely adopted deep learning technology in the medical imaging community will be discussed in more detail in this review as one such breakthrough method. In this process, we take input as an image and classify objects by general or by the instance, by doing this we can predict the object and its location. In the segmentation task, we present the output and check if the dimensions of the image are the same as the original input, this is possible with the help of encoders and decoders and hence we use this process as background for U-net.

Semantic segmentation[12] is a crucial component in computer vision for accurate inference in tasks

*chigurupati.k@northeastern.edu
†mehta.rajat@northeastern.edu
‡sivakumar.vig@northeastern.edu

such as self-driving cars and robotic navigation. It involves grouping pixels into meaningful categories, such as roads, people, cars, or trees, through pixel-level categorization. This enables models to understand the context of their environment and make precise decisions. Pixel accuracy is essential in providing insights into the scene, allowing for fine-grained analysis of the visual input.

The paper introduces a more elegant architecture called the "fully convolutional network" [7] that modifies and extends the existing architecture to work with very few training images and yield more precise segmentations. The main idea is to supplement a contracting network with successive layers where pooling operators are replaced by upsampling operators to increase the resolution of the output. High-resolution features from the contracting path are combined with the upsampled output to allow for localization.

One important modification in this architecture is the inclusion of a large number of feature channels in the upsampling part to propagate context information to higher resolution layers. This results in a U-shaped architecture that is more symmetric to the contracting path. The network does not have any fully connected layers and only uses the valid part of each convolution, allowing for seamless segmentation of arbitrarily large images using an overlap-tile strategy.

2 Background and Related Work

The U-Net neural network architecture was specifically designed for the task of semantic segmentation, which involves predicting the pixel-level classification of objects in an image. U-Net has been widely used in instance segmentation [1] and panoptic segmentation [6] tasks too due to its ability to capture fine-grained details and its efficiency in processing images at different resolutions. The U-Net architecture incorporates skip connections to fuse features from multiple scales in the image, mitigating information loss during downsampling. This allows the U-Net to capture both local and global context, resulting in accurate and detailed segmentation masks, making it a powerful tool for image segmentation tasks.

2.1 Encoder and Decoder

The encoder in a U-Net is responsible for extracting features from the input image. It typically consists of several convolutional layers followed by activation functions such as ReLU (Rectified Linear Unit) to introduce non-linearity.

The decoder in a U-Net is responsible for upsampling the feature maps obtained from the encoder to their original spatial dimensions, while also reducing the number of feature channels.

2.2 Segmentation types

2.2.1 Instance segmentation

This is a computer vision task that involves detecting and segmenting individual objects or instances in an image at the pixel level, providing precise masks for each object.

2.2.2 Panoptic Segmentation

Panoptic segmentation is a computer vision task that combines instance segmentation and semantic segmentation. It involves detecting and segmenting both objects and stuff (e.g., roads, sky, grass) in an image, providing a detailed pixel-level understanding of the scene.

2.3 U-Net Variants

Several variants of the U-Net architecture have been proposed to further improve its performance, including the U-Net++, Recurrent U-Net and Dense U-Net.

2.3.1 U-Net++

This is a variant of the original U-Net architecture that was proposed by Zongwei Zhou et al. in 2018[13]. It extends the U-Net by adding more skip connections to capture features from multiple scales in a more comprehensive manner.

2.3.2 Recurrent U-Net

This work, proposed by Chen et al. in 2018[14], integrates recurrent neural networks (RNNs) into the U-Net architecture to capture contextual information across different scales and improve the model's ability to handle images with large contextual variations.

2.3.3 Dense U-Net

This variant of U-Net, proposed by Li et al. in 2019[2], introduces dense skip connections between encoder and decoder blocks to allow for better gradient flow and feature reuse, resulting in improved segmentation accuracy.

In addition to these variants, there have been numerous efforts to optimize the U-Net architecture for specific applications, such as multi-modal image segmentation, 3D image segmentation, and real-time segmentation on embedded systems. These adaptations and optimizations demonstrate the versatility and flexibility of the U-Net architecture and its potential for further advancement in various image processing tasks.

2.4 R-CNN

U-Net is specifically designed for image segmentation tasks, where the goal is to segment an image into different regions of interest. In contrast, Faster R-CNN and Mask R-CNN are designed for object detection tasks, where the goal is to detect and localize objects in an image, and optionally generate instance-level segmentation masks. However, these architectures can be combined or used in conjunction with each other for more complex tasks that require both image segmentation and object detection capabilities

2.4.1 Mask R-CNN

(Region-based Convolutional Neural Network) is a state-of-the-art deep learning model for object instance segmentation, which is a task that involves detecting and segmenting objects at the pixel level in an image. Mask R-CNN[5] is an extension of the Faster R-CNN architecture, which is a popular object detection model.

2.4.2 Faster R-CNN

(Region-based Convolutional Neural Network) is a widely used deep learning model for object detection, which is the task of detecting objects and their corresponding bounding boxes in images. Faster R-CNN was proposed by Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun in their 2015 paper titled "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks".[10]

3 Approach

3.1 Data

The data is a set of images chosen at various locations chosen at random in the subsurface. The images are 101 x 101 pixels and each pixel is classified as either salt or sediment. In addition to the seismic images, the depth of the imaged location is provided for each image. The goal of the project is to segment regions that contain salt. It is utilized in various computer vision tasks, such as object detection, instance segmentation, image captioning, key points detection, panoptic segmentation, dense pose, and stuff image segmentation.The dataset provides extensive annotations, including bounding boxes, per-instance segmentation, key points, and per-pixel segmentation masks.

3.2 Network Architecture

The architecture of the network, shown in Fig. 2, consists of a contracting path on the left side and an expansive path on the right side. The contracting path follows a typical convolutional network structure, with repeated application of two 3x3 convolutions followed by ReLU activation and 2x2 max pooling for downsampling. The number of feature channels is doubled at each downsampling step. The expansive path includes upsampling of the feature map, followed by a 2x2 convolution to halve the number of feature channels, concatenation with the cropped feature map from the contracting path, and two 3x3 convolutions with ReLU activation. Cropping is necessary due to the loss of border pixels in convolutions.

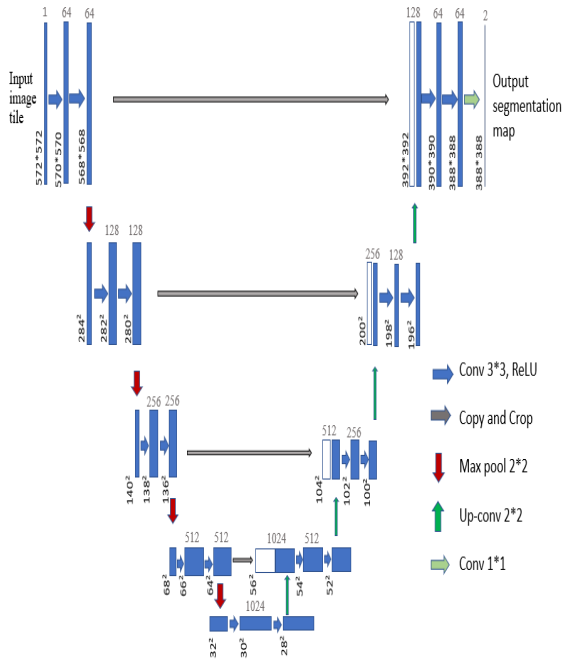

Figure 2: U-Net Architecture

The final layer uses a 1x1 convolution to map the feature vectors to the desired number of classes. Overall, the network has 23 convolutional layers. To ensure seamless tiling of the output segmentation map, the input tile size should be selected such that all 2x2 max pooling operations are applied to a layer with an even x- and y-size.

3.3 Methods

The number of channels in input is 1, number of channels in output is 1, number of levels is 3, Train-test split is 85:15, 15 percent of dataset used for testing and rest 85percent for training. We have used Adam Optimizer in our study. The Adam optimizer combines elements of both momentum-based optimization and RMSprop to adaptively adjust the learning rate for each parameter based on the gradients and squared gradients

3.4 Loss function

Since the U-Net architecture is typically used for image segmentation tasks where the target is to generate binary masks (with pixel values of 1 indicating the presence of an object), a common loss function for comparing the model's output with the ground truth is the categorical cross-entropy loss (or binary cross-entropy loss [8] in the case of a single label). This loss function measures the discrepancy between the predicted segmentation map and the ground truth segmentation map on a per-pixel basis, making it suitable for training the U-Net to generate accurate binary masks for object segmentation.

In semantic segmentation tasks, the commonly used loss function is cross-entropy. Semantic segmentation involves classifying each pixel in an image into a specific semantic class. The cross-entropy loss measures the discrepancy between the predicted probability distribution for each pixel and the expected probability distribution, which represents the ground truth.

The expected probability distribution for each pixel must first be obtained before we can calculate cross-entropy loss for semantic segmentation. Typically, a neural network with a softmax activation function on the output layer is used for this. A probability distribution over the potential classes for each pixel is produced by the softmax function.

The true probability distribution for each pixel must then be obtained. For each pixel, a one-hot encoded mask is typically created, with a value of 1 for the pixel's actual class and a value of 0 for all other classes.

$$-\sum y_t[i] * log(y_p[i]))$$

yt[i] is the true class label for class yp[i] is the predicted probability for class log(x) is the natural logarithm of x sigma denotes the sum over all classes

3.5 Up-sampling method

Another detail is the choice of the up-sampling method for the decoder. Transpose convolution[3] is one common method used for up-sampling.

Transpose convolution is a convolutional operation commonly used in neural networks for image segmentation tasks. It is also referred to as deconvolution or fractionally strided convolution. The purpose of semantic segmentation is to label every pixel in an image with a class, and achieving this requires the network to acquire sophisticated features that represent the semantic content of the image.

Transpose convolution is a useful technique in semantic segmentation networks that allows for the enhancement of feature maps' spatial resolution. Unlike traditional convolutional layers, which often result in output feature maps smaller than the input feature maps due to pooling operations, transpose convolution can expand the feature map size by adding zeros to the input feature map and then applying a convolutional operation with a larger stride

The U-Net is a popular method for semantic segmentation that involves two parts: a contracting path that reduces the size of the input image, and an expansive path that increases the size of the feature map. The expansive path uses transpose convolutional layers to gradually make the feature map larger, while also incorporating information from earlier layers in the contracting path.

4 Implementation and Results

The learning rate = 0.001, no of epochs = 40, and batch size = 64 for the 1st run, we have used Image size as 128x128 and threshold to remove weak predictions = 0.5. The learning rate = 0.001, no of epochs = 100, batch size = 64, Image size as 128x128 and Threshold is 0.5 for the 2nd run and The learning rate = 0.001, no of epochs = 10000, batch size = 64, Image size as 256x256 and Threshold is 0.5 for the third run. Filter size at respective levels - 16,32,64. The training parameters no of epochs is 100 and time taken to train model 248.06 sec.

Our solution is written in PyTorch[9] in Colab Pro,

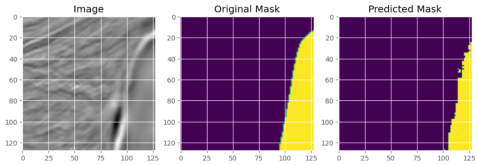

Figure 3: Run 1

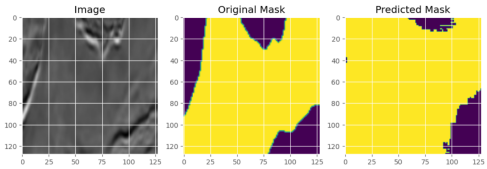

Figure 4: Run 2

which is a subscription-based service by Google that enhances the free Google Colaboratory Jupyter notebook environment with faster GPU/TPU acceleration, priority access to new features, longer session runtimes, and increased storage limits. Colab Pro offers access to GPUs and TPUs for accelerated computing, but the specific type of GPU or TPU allocated to a user's Colab Pro session may vary depending on availability and Google's infrastructure. GPUs such as NVIDIA Tesla V100 and NVIDIA Tesla T4 are commonly used in Google's data centers to provide high-performance computing capabilities to Colab Pro users.

5 Conclusion and Future Work

The U-Net architecture achieves very good performance on segmentation applications. We have successfully used our implementation. The approach proposed in the project segments the salt from the

Train Loss	Test Loss
0.23	0.27
0.26	0.29
0.27	0.29

Table 1: Train and Test Losses

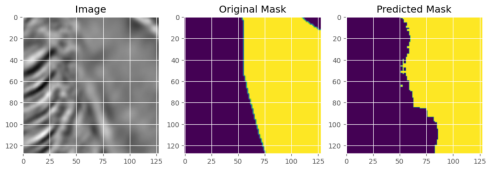

Figure 5: Run 3

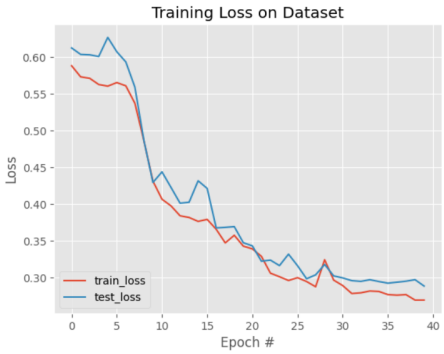

Figure 6: Loss vs Epoch

surface of the earth. Although, the results produced by the model are not great in comparison to the current state of the art models, the full potential of the proposed method is still to be explored. For example, better results can be obtained if the model is allowed to train longer on the dataset. Results will also improve if we train the model end-to-end instead of training a disparity model first and utilizing it's weights. For the future work, we would like to train the model completely end-to-end on bigger datasets to test the capabilities of the model.

6 Acknowledgement

The study was supported by our professor Dr. Jerome Braun of the Northeastern University College Of Engineering. [1]

[1]All the team members have contributed equally to this project

References

[1] C A and C A L. Pixel level instance segmentation using single shot detectors and semantic segmentation networks, 2019. 2

[2] Sijing Cai, Yunxian Tian, Harvey Lui, Haishan Zeng, Yi Wu, and Guannan Chen. Dense-unet: a novel multiphoton in vivo cellular image segmentation model based on a convolutional neural network, 2020. 3

[3] Vincent Dumoulin and Francesco Visin. A guide to convolution arithmetic for deep learning, 2018. 5

[4] Cui Jiali et al. Eye detection with faster r-cnn. In *IEEE Advances in Comp. Tech*, 2019. 1

[5] Ronghang Hu, Piotr Dollár, Kaiming He, Trevor Darrell, and Ross Girshick. Learning to segment every thing, 2018. 1, 3

[6] Brünger J, Gentz M, Traulsen I, and Koch R. Panoptic segmentation of individual pigs for posture recognitionl, 2013. 2

[7] Long J, Shelhamer E, and Darrell T. Fully convolutional networks for semantic segmentation, 2014. 2

[8] Zhe Liu. The research of feature extraction algorithm by integrating t-rank and softmax methods, 2016. 4

[9] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zach DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library, 2019. 5

[10] Girshick R, Donahue J, Darrell T, and Malik J. Rich feature hierarchies for accurate object detection and semantic segmentation detection with faster r-cnn. In *CVPR*, 2014. 3

[11] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks, 2016. 1

[12] Wang Xiaogang. Semantic object segmentation, video segmentation and its applications, 2011. 1

[13] Zongwei Zhou, Md Mahfuzur Rahman Siddiquee, Nima Tajbakhsh, and Jianming Liang. Unet++: A nested u-net architecture for medical image segmentation, 2018. 2

[14] Qiang Zuo, Songyu Chen, and Zhifang Wang. R2au-net: Attention recurrent residual convolutional neu-

ral network for multimodal medical image segmentation, 2021. 3