Post-Disaster Segmentation Using FloodNet

Kushagra Gupta
Department of Statistics
Stanford University
kushgpt@stanford.edu

Priya Mishra
Department of Computer Science
Stanford University
priyamis@stanford.edu

Abstract

Our project aims to make post-disaster management easier while also providing valuable knowledge about vulnerable areas. We use semantic segmentation for scene parsing of high-quality, low-resolution post-disaster images from AUVs. We segment the images into ten output classes that distinguish between flooded and non-flooded structures. Using the FloodNet dataset for our task, we train models of increasing complexity for segmentation and try to cover up the lackings of previous models. We started with UNet and FCN, which prompted us to try pre-trained models that capture multiple resolutions and build a context for each pixel. Some challenging aspects of our task are the vital role of context in distinguishing flooded from non-flooded structures and the attention to finer details demanded by small-sized classes. DeepLabV3, PSPNet, and Segformer architectures overcome the shortcomings of our baseline models and tackle the challenging aspects, improving the mean IoU and the class-wise IoU scores.

1. Introduction

Deep learning is emerging as an important tool for disaster risk management and post-disaster response. This push is also reflected in the goals of global organizations like the United Nations and World Bank, aiming to improve the infrastructure resiliency against shocks in developing countries by highlighting at-risk areas [1]. An integral part of this analysis is identifying the worst affected areas from natural disasters and developing a decision support system. Computer vision and visual scene understanding provide a way to quickly interpret post-disaster scenarios to help design immediate and effective relief while forming the basis for recognizing the weak points in the infrastructure.

Visual understanding datasets for post-disaster assessment generally come in three flavours - satellite images [2], social media images [18], and unmanned aerial vehicle (UAV) images m [14]. Satellite images are expensive to procure and suffer from low resolution and noise due to the high altitude. Social media images are low quality, noisy and not scalable for deep learning approaches. In light of these challenges, we decided to work with the third category of images to get high-resolution images taken from low altitudes.

Aerial image classification is a two-step approach involving semantic segmentation of images, followed by pixel-wise classification into predefined classes. Semantic segmentation divides an image into homogeneous regions, which helps understand the image's local structure and identify boundaries (lines, curves, etc). More precisely, image segmentation is the process of assigning a label to every pixel in an image such that pixels with the same label share certain visual characteristics.

The inputs in our algorithms are images. We then use multiple convolution-based and transformer-based models to predict the segmentation masks for the images. More specifically, we use fully convolution networks (FCN), U-Nets, PSPNet, DeepLabv3 and Segformer in our work. The segmentation masks are the predictions given by the model specifying the class of each pixel in the image.

2. Related Work

For our project, we survey the original paper introducing the FloodNet dataset [14] that we use in our work. FloodNet dataset has the advantage of being both high-resolution and images being obtained from lower altitudes which makes it particularly useful for analysing post-disaster damage. The dataset can be used for a classification task focused on distinguishing between flooded and non-flooded images, segmentation task focused on fine-grained classification of images into 10 classes, and a visual question answering task. In our project, we focus on the segmentation task. The paper uses three models – ENet, PSPNet, and DeepLabV3+ for the segmentation task of which PSPNet performs the best. All the models perform poorly on distinguishing between flooded and non-flooded buildings since it is challenging to make this distinction based on only the top view of the buildings present in the images. Similarly the distinction between flooded and non-flooded roads depends on the

overall context of the image and is a difficult class. Smaller objects such as pool and vehicle have lower performance due to their small sizes.

We first use a simple UNet [16] architecture trained from scratch for the segmentation problem. UNets capture contextual information through an autoencoder-decoder type architecture with skip-like connections between the downsampling and upsampling layers. UNets show promising results for segmentation tasks which rely on the global context for per-pixel segmentation with moderately large datasets. Since UNets do not necessarily capture local context level information,, we start with a smaller sized UNet as compare to the original architecture to test the reliability of predictions, particularly for classes that are difficult to predict (like flooded vs non-flooded and smaller classes like the vehicle).

Our next approach was to try a Fully Convolutional Network (FCN) [9]. FCNs are an important class of models for semantic segmentation that fundamentally changed how people approached the problem, in addition to opening various avenues for future research. FCN replaces the fully connected layers at the end of an image classification network with upsampling followed by convolutional layers. This allows the features learned from the image to be used for pixel level classification, rather than classifying the image as a whole.

DeepLabv3 [4, 5] is one of the most popular state-of-the-art models used for semantic segmentation. It avoids losing spatial context caused by repeated pooling and striding using atrous convolutions. This is important for our dataset since the labelling of certain classes such as "building-flooded" is based on the nearby context, specifically, whether it is surrounded by flood water. Further, deeplabv3 generates multi-scale feature maps which lead to greater performance when the objects in the image have different sizes, as is the case for our dataset. Hence, we use DeepLabv3 as one of our proposed models for this dataset.

We also propose using SegFormer [20], a recent transformer-based model for image segmentation. SegFormer employs a hierarchical encoder to obtain both local and non-local attentions. Its effective receptive field is larger than that of deeplabv3 at its deepest stage. It also is less computationally expensive. Further, it has been shown that SegFormer produces finer-detailed masks around object boundaries. This shows potential for achieving good results on our dataset since the classification of categories such as "flooded-building" is closely related to the other classes predicted on its boundaries due to reasons described above. We finally tried PSPNet [22] given its popularity for problems that rely heavily on the context of pixels. The idea behind PSPNets is similar to inception type layers in GoogleNet. The pyramid pooling performed on sub-regions of different sizes focus on particular parts of the regions around a pixel, acting similar to how attention works in transformers. The pooled feature maps made from varying levels of granularity of subregions fused together capture the local as well as global context, forming a global contextual prior. In addition, PSPNets use auxiliary loss (similar to GoogleNet) to speed up the training of the model. PSPNets have worked well for complex scene parsing problems, similar to our post-disaster semantic segmentation problem [13] [8].

[15] studies a related problem of detecting the flooded areas in Houston from data collected by UAVs. This work proposes an integration of densely connected RNN and CNN which addresses the difficulty in labelling smaller objects and semantic ambiguities across object boundaries. These challenges are relevant to the FloodNet dataset since it includes classes which have much smaller sizes than others (e.g. vehicles, ponds). The classification of a building as flooded depends on whether it touches flood water on any side and hence improving the accuracy along object boundaries should be useful in improving performance on this class.

[12] apply semantic segmentation on a dataset, named Volan2019, which includes 16 videos collected from recent natural disasters, containing people, flooded area, building roof (damaged and undamaged), car, debris, vegetation, road, and boat. The dataset shares some common classes with FloodNet such as vehicles, roads, flooded areas and trees. This work experiments with Mask-RCNN and PSPNet for their dataset.

3. Dataset and Features

In this work, we use the FloodNet [14] dataset. The dataset contains 2343 high-resolution UAV images and their segmentation masks. The dataset is split into train, validation and test sets. The train, validation and test sets include 1445, 450, and 448 images respectively. The segmentation mask images are of the same size as the input images. The segmentation masks have 10 output classes – background, building-flooded, building-non-flooded, road-flooded, road-non-flooded, water, tree, vehicle, pool, grass.

The class "water" represents natural water bodies such as lakes or rivers and distinguishes them from flood water. A building is classified as flooded if it is in contact with flood water on any side. To distinguish natural water from flood water, the 'water' class has been included to represent any natural water bodies.

The image and mask sizes in the dataset can be 3000 x 4000 or 3072 x 4592. For the models U-Net [16], FCN-ResNet, PSPNet, and DeepLabv3 [4, 5], we resize the images and their ground truth segmentation masks to 512 x 512. The pretrained models used further require the input images to be in the range [0,1], and normalized using the mean = [0.485, 0.456, 0.406] and std = [0.229, 0.224,

0.225]. Hence, for these models, we preprocess the dataset to resize the input images and their masks, and then normalize the images using the specified mean and std.

The SegFormer model expects specific shape and names for the input features. We use the SegFormerFeatureExtractor to transform the input into the required format. The feature extractor, by default, resizes the image to be 512 x 512, and normalizes the values using the mean and std mentioned above. This transformation is hence, exactly the same as those for the other models. The output of the feature extractor is a dictionary with the fields "pixel_values" and "labels".

We also experiment with applying additional data augmentations and evaluate their impact on the performance. These transforms include horizontal and vertical flipping with a probability of 0.5, and color jitter with brightness=0.25, contrast=0.25, saturation=0.25, and hue=0.1. For all the transformations (resizing, normalization, flipping, color jitter), we use the torchvision library [10].

4. Methods

4.1. FCN

Fully convolutional networks (FCN) changed the scene for semantic segmentation when it was published in 2015 [9] and paved the way for numerous advancements in architectures for semantic segmentation. FCN are a natural extension to image classification tasks, where instead of outputting a single label for the whole image we output a label for each pixel. FCN, therefore, essentially extend the image classification task to each pixel by replacing fully connected layers with fully convolutional layers. Fully connected layers reduce the dimensions of feature maps and allow the intermediate layers to learn complex features, which instead of being passed to fully connected layers for classification go through upsampling via deconvolution to return output of the same size as the input image. The complex intermediate features guide the pixel-level classification.

Architecture

FCN follow the architecture of CNNs used for image classification, with the latter layers replaced by convolutional and upsampling layers (figure 8). To reduce the computational burden of training a network from scratch, we used transfer learning with pre-trained ResNet-50 backbone for feature extraction from input images. Images produced with this architecture are coarse, as spatial location is lost by going deeper to generate more complex features. To get back the location information from shallower layers, the output from fine layers and coarser layers are combined to make local predictions that respect global structure. The combination is done by 2x upsampling of a pooling layer fused with the output of the next layer, and the fusion appropriately upsampled to return to the input image size. This is akin to boosting, where the outputs from this fusions of global and coarser layers are added as outputs from multiple models to improve accuracy.

4.2. UNet

We trained a small U-Net model as a baseline. U-Net is a popular architecture for semantic segmentation, and has been a popular choice for the FloodNet dataset [7]. U-Nets tend to perform better than comparative models in environments of low to medium quantities of training data, which matches the number of images in FloodNet. The architecture for U-Nets is similar to that of convolutional autoencoders, but with skip-like connections between feature maps of downsampling and upsampling layers [16]. This allows information to be recovered from the encoder part before the compressive bottleneck. This information is concatenated with the output states of the decoder to provide context for the next decoding layer. This removes the burden of generalizing the information from previous layers, and allows the U-Net to focus on extracting information.

Architecture

The network consists of symmetric contracting and expansive paths (figure 9). The contracting path follows the typical architecture of a convolutional network. It consists of the repeated application of two 3x3 convolutions, each followed by a batchnorm and rectified linear unit (Re-LU) layer (collectively called a two-conv layer in our model) . A 2x2 max pooling operation attends the set of convolutions with stride 2 for downsampling. At each down-sampling step, we double the number of feature channels. Every step in the expansive path consists of an up-sampling of the feature map through an inverse convolution operation with a transpose convolution layer (Conv2DTranspose). After using padding for dimension compatibility, the output from upsampling is concatenated with the corresponding feature map from the contracting path. The upsampling is followed by a two-conv layer resulting in output with half the number of channels as the input. A 1x1 convolution maps the feature vector to the desired number of classes at the final layer. The architecture is a slightly modified version of a scaled-down official UNet architecture.

4.3. DeepLabv3

Convolution-based models for semantic segmentation usually include repeated pooling and striding operations, which results in low resolution feature maps. DeepLabv3 instead proposes using atrous convolutions to obtain dense feature maps which are more useful for the semantic segmentation tasks where we need better spatial information.

Figure 1. Example images and their ground truth segmentation masks. The color mapping for the 10 classes is shown on the right.

Atrous convoluion does not add any parameters to the model and hence maintains the same computational complexity. Further, by varying the rate in atrous convolutions, we can learn features at different input scales. This is important since various classes in the image may have different sizes. This is relevant to our dataset, where certain classes such as pool and vehicle are smaller in size as compared to other classes, and are hence more difficult to label.

Architecture

The initial blocks of DeepLabv3 follow the architecture of most CNNs. The model includes a atrous spatial pyramid pooling (ASPP) after block 4. The ASPP module consists of one 1 x 1 convolution and three 3 x 3 atrous convolutions with rates of (6, 12, 18) to effectively capture information at different scales. The ASPP module combines these with image-level features to include more global context. The image-level features are obtained by global average pooling followed by 1x1 convolution with 256 channels, batch normalization, and a bilinear upsampling to obtain the correct spatial dimension. The features from the atrous convolutions and the image-level features are then concatenated, followed by 1 x 1 convolutional layer with 256 channel, batch norm, and a final 1 x 1 convolution to generate the final output logits.

4.4. PSPNet

Pyramid Scene Parsing Network (PSPNet) gained popularity after winning the ImageNet Scene Parsing Challenge 2016 [22]. These networks have been prevalent for complex scene parsing problems such as post-disaster semantic segmentation [13], [8]. PSPNets use the global contextual information to improve performance by using a pyramid pooling module that combines different-region based context aggregation. Contextual aggregation for semantic segmentation mainly relies on spatial pyramid pooling at several grid scales and an encoder-decoder strategy that coalesces mid-level and high-level features. PSPNets rely on the former and perform better in our problem. Information to distinguish between flooded and non-flooded roads and buildings primarily comes from the structure's context, as their vertical view looks identical. We can make this distinction only when a structure is waterlogged on one side. In addition, PSPNets promote faster training with auxiliary loss [17] by helping gradient flow.

Architecture

PSPNet begins with a pre-trained ResNet backbone with dilated network strategy to extract the feature map from the input image (figure 3). The feature map size is 1/8th of the input image. Feature extraction is followed by the pyramid pooling layers, which fuse features under four different pyramid scales. Similar to the architecture in the original paper, we use bin sizes of 1x1, 2x2, 3x3, and 6x6 for the sub-region average pooling of each feature map. A bin of size pxp divides the feature map into pxp sub-regions which undergo average pooling. The feature maps generated by the pyramid pooling represent the global scenery prior. A 1x1 convolution follows this step to reduce the number of feature maps. These feature maps are upsampled using bilinear interpolation and concatenated with the feature map from the backbone. This forms the final feature representation, which carries local and global context information. Finally, we use a convolution layer to generate the prediction.

4.5. SegFormer

Transformers are increasingly being applied to problems in computer vision based on their great performance in natural langauge processing tasks. Vision transformer, proposed for image classification task, performed well on ImageNet and showed promise for applying transformers to vision tasks. Another model, SETR was proposed for semantic segmentation which used ViT as a backbone. However, these models are computationally expensive. SegFormer is a recently proposed model which addresses some of the limitations of past transformer-based architectures. SegFormer uses a hierarchial encoder to generate multi-scale features

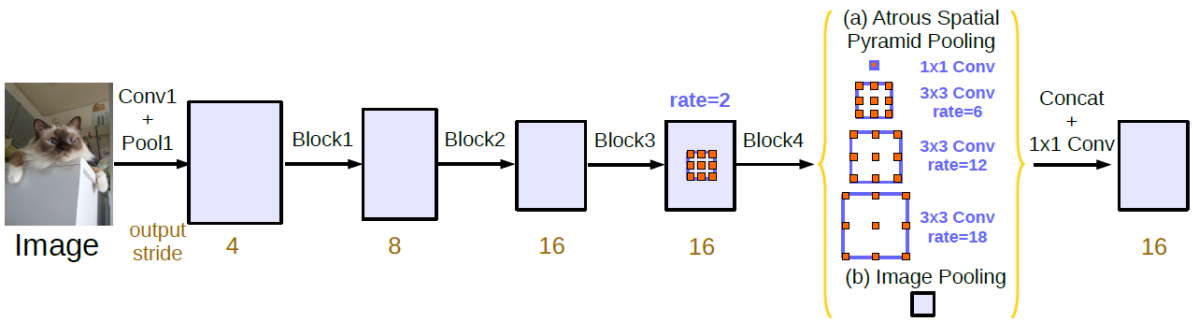

Figure 2. DeepLabv3 architecture. Source: [4]

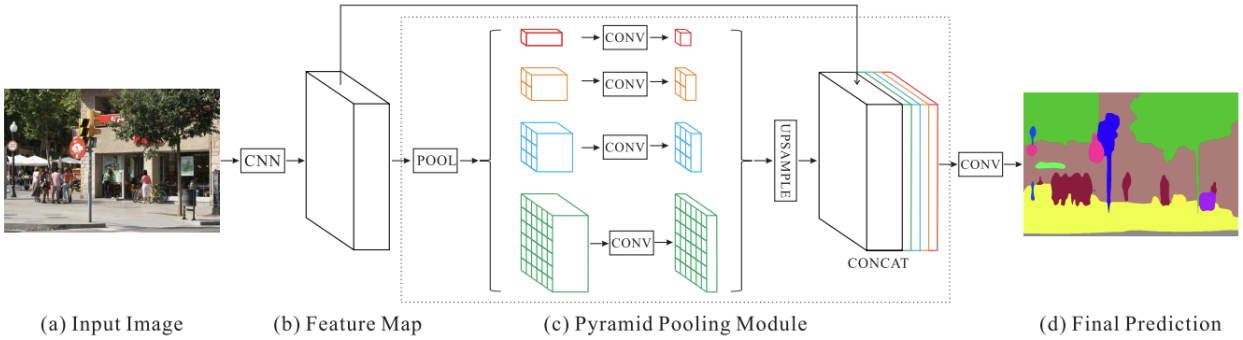

Figure 3. PSPNet architecture. Source: [22]

as opposed to the single-scale features generated by ViT or SETR. The encoder is hence, able to produce both highly local and non-local attentions, and has a larger effective receptive field than convolution models such as DeepLabv3 at their deepest stage. It uses a lightweight decoder which makes it computationally less expensive. Further, it does not use positional encodings which allows it maintain performance even when the test images may vary in resolution from the training images.

Architecture

SegFormer is a transformer-based model for semantic segmentation. It consists of a hierarchial encoder which allows it to obtain multi-scale features and a lightweight multilayer perceptron (MLP) decoder.

Each input image is divided into patches of dimension 4 x 4 which are used as input to the hierarchial encoder. As shown in Figure 2, the encoder outputs features with resolution varying from 1/4 to 1/32. This allows the model to capture both coarser and finer features. These features are then given as input to the MLP-decoder which outputs a segmentation mask of size 1/4 of the original image size.

The model does not use positional encoding, and instead introduces Mix-FFN obtained by combining 3 x 3 convolutions with a feedforward network (FFN). This Mix-FFN is formulated as:

$$x_{out} = \text{MLP}(\text{GELU}(\text{Conv}_{3x3}(\text{MLP}(\text{x}_{in})))) + x_{in} \quad (1)$$

where x_{in} is the output from the self-attention layer.

The SegFormer encoder, also called the Mix Transform encoder (MiT) is available in a series of sizes from MiT-B0 to MiT-B5. These have the same architecture but different sizes. In our work, we use MiT-B0 (3.8M parameters) and MiT-B4 (64.1M parameters) to compare the performance of the model with increasing size. We were unable to use MiT-B5 due to memory constraints.

5. Results

We use PyTorch [6] and PyTorch Lightning [11] frameworks for writing, training and testing our models. We use Weights and Biases [3] to track and log metrics for our experiments. Additionally, we use Segmentation Models Pytorch [21] for our PSPNet implementation, and HuggingFace library for SegFormer [19].

In all our models, we use the Adam optimizer. We also use the ReduceLROnPlateau as our learning rate scheduler to reduce the learning rate when the validation loss stops improving. The scheduler reduces the learning rate by a

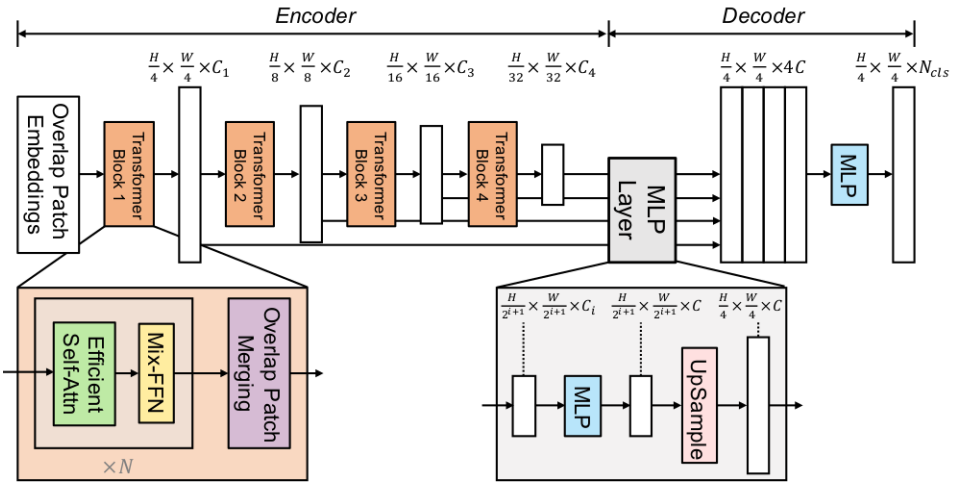

Figure 4. SegFormer architecture. Source: [20]

Model	lr	Batch Size	Epochs
UNet	1e-4	2	15
FCN	5e-5	8	50
DeepLabv3	1e-4	8	100
PSPNet	5e-5	8	100
SegFormer (MiT-B0)	5e-5	8	100
SegFormer (MiT-B5)	1e-5	4	50

Table 1. Hyper parameters used in different models.

factor of 0.1 when the validation loss has no improvement for 10 epochs. The models predict the segmentation mask for the images. We use the following metrics to compare the predicted segmentation masks and the ground truth segmentation masks present in the dataset and evaluate the performance of the different models.

1. **Pixel accuracy:** The fraction of pixels that are classified correctly.

2. **Jaccard Index (IoU):** The Jaccard index quantifies the overlap between the ground truth and the prediction segmentation mask by computing their intersection over union (Equation 2).

$$J(A, B) = \frac{|\, A \cap B \,|}{|\, A \cup B \,|} \tag{2}$$

We discuss the other hyper parameters used in the different models below (summarized in Table 1):

We started with a learning rate of 1e-4, 5e-5, and 5e-5 for the UNet, FCN, and PSPNet respectively. The batchsize was 2 for UNet, and 8 for FCN and PSPNet respectively, owing to the platforms used to run the models. FCN and PSPNet were trained on the AWS server, and the UNet being trained on Stanford's Sherlock server. In addition, we tried multiple pretrained backbones for our heavier models like PSPNet and DeepLabV3. We also had to consider the number of trainable parameters, and eventually decided to use ResNet50 as the backbone for FCN and DeepLabV3. We ran a grid search over the lighter pretrained models for PSPNet, trying ResNet50, DPN68, VGG18, DenseNet 161, and Xception and decided to choose Xception based on the performance over 5 epochs.

The results of our experiment are summarized in table 2. Our initial approach was to try out popular image segmentation models, which motivated us to use FCN. We also tried a simple UNet from scratch to leverage contextual information to help the model learn. The UNet and FCN acted as baselines for the task, given their popularity for semantic segmentation and UNets' ability to work well on moderately sized datasets. We found that the mean IoU and class-wise IoU were extremely low for the UNet, primarily attributed to training the model from scratch. The mean IoU for the test set was just 24, and the class wise IoU was 0 for classes like flooded roads and buildings. Our UNet cannot learn good feature mappings, which hurts its performance, and fails to capture the local context affecting its ability to detect flooded structures.

The results for FCN are an improvement over UNet owing to the pre-trained feature mapping of ResNet-50. Mean IoU for test set climbed up to 41. Still, the absence of contextualized information limits the learning capability of the model. The context is vital to distinguish flooded from non-flooded objects, so we see improved performance for non-flooded structures over flooded structures (40 vs 64 IoU for buildings, and 32 vs 67 for roads). In particular, the IoU scores for small classes (like vehicles and pools) are still

Model	Val IoU	Test IoU	Val Acc.	Test Acc.	Training time per epoch (min)
UNet	24.89	24.57	76.81	76.12	2.54
FCN	43.03	41.61	83.54	83.30	11.04
DeepLabv3	48.68	47.84	84.07	87.01	9.38
PSPNet	50.97	50.31	87.54	88.68	17.06
SegFormer (MiT-B0)	58.76	54.86	88.52	88.61	8.48
SegFormer (MiT-B4)	60.62	56.78	88.36	88.50	8.62

Table 2. IoU and pixel accuracy on validation and test sets of FloodNet.

Model	Val IoU	Test IoU	Val Acc.	Test Acc
SegFormer (MiT-B0)	58.10	53.06	88.59	88.20
SegFormer (MiT-B4)	59.91	56.26	84.06	87.01

Table 3. IoU and pixel accuracy on validation and test sets of FloodNet with data augmentation.

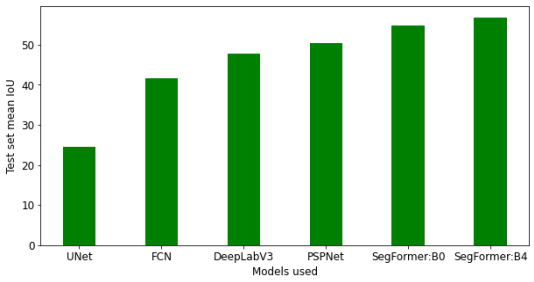

Figure 5. Test set mean IoU.

0. We resized our images from 2000 to 512, which further exacerbated the loss of quality for smaller classes. As the networks get deeper, it loses the image's finer details in longer-term dependencies, which can, therefore, not be recreated from upsampling. This distorts the predictions for smaller classes and the prediction at boundaries, which is evident in the low IoU scores and poor boundary separation in the visualization.

DeepLabV3, PSPNet, and SegFormer try to solve these problems as described in the methods section. DeepLabV3 reduces the loss from downsampling using atrous convolutions. In addition, the multi-scale resolution allows the network to detect objects of different sizes, preventing the model from overfitting to the size of classes. The mean IoU for test set increases to 47, with class wise IoU also increasing when compared to FCN. However, the score for smaller classes (like vehicles and pools) still remained 0, which prompted us to try models which capture local context better.

PSPNets are designed to fuse local and global contexts by using pyramid poolings of various sizes. This allows PSPNets to marginally improve over DeepLabV3 (48 vs 50), given its focus on building a global contextual prior, which is essential for complex scene parsing. PSPNet also increases the test set IoU for the background class from 0 to 27 over DeepLabV3, but the gain for other classes is marginal or non-existent. In addition, the auxiliary loss in PSPNets allows it to converge faster, and longer training times might reduce the difference between the two networks.

Finally, Segformer focuses heavily on improving finer-grained representation, which is evident from the inclusion of the hierarchical encoder and improved performance at the class boundary. The hierarchical encoder outputs feature at multiple resolutions, which enjoys context benefits similar to the spatial pyramid pooling in PSPNets and DeepLabV3. SegFormer gave the highest test IoU at 55 and 57 for the MiT-B0 and its heavier counterpart MiT-B4. SegFormer enjoys a larger receptive field than DeepLabV3. MiT-B4 was also the only model with non zero IoU for the small vehicles and pool classes, giving evidence to its ability to preserve finer grained details. Better contextual representation in PSPNet and Segformer improves the flooded structures' performance. This is evident in SegFormer models when they improve performance for classes like background (from 0 to 44 in SegFormer) that all previous models other than PSPNet (score 27) were struggling with. A summary of our mean IoU results is presented in figure 5. Class wise results for all models can be found in the Appendix table 4.

Our baseline models were classifying all objects with a shade of blue to water. This effect was particularly pronounced for buildings. We tried data augmentation with color jittering to make our models robust to similar effects. The results for the models trained on the augmented training set are given in 3. We didn't see a significant increase in the performance of SegFormer, and decided not to re-train our

other models on the augmented data given the large training time. In general, segmentation models need long training times with many parameters. This calls for running more complex models for longer, which is what we observe in our results. SegFormers are an exception, as they are lighter than our other models and more computationally efficient.

In Figure 10, we show the segmentation masks for the ground truth and the predictions made by various models in our work. Due to space constraints, we only show the segmentation masks for ground truth, U-Net, PSPNet and SegFormer (MiT-B4). The segmentation masks for other models are present in the appendix.

We notice that for the first example (row 1 of Fig.10), SegFormer identifies the flooded road much better than PSPNet and our baseline U-Net. PSPNet misclassifies some parts of the flooded-road (the blue segments) as water.

In the second example (row 2 of Fig.10, SegFormer accurately segments the pool and vehicle classes which are misclassified by PSPNet and U-Net. Smaller objects are difficult to segment as discussed above. The hierarchial encoder employed by SegFormer allows it to output feature maps at various resolutions improving its performance on the classes representing smaller objects.

In the third example (row 3 of Fig.10, we can see that our baseline model does not perform well on identifying flooded-buildings or flooded-roads and instead classifies them as water, possibly because of their color. PSPNet and SegFormer give significantly better performance on these classes as shown. We further see that SegFormer produces a finer-detailed mask around object boundaries. The top right corner of the image is misclassified as water by PSPNet, but accurately segmented by the SegFormer.

6. Conclusion

In this work, we aim at using deep learning for post-disaster segmentation using the FloodNet dataset. We used U-Net and FCN as our baseline models owing to their performance on common semantic segmentation benchmarks. Based on our results, we identified that it is difficult to classify flooded classes. Further, due to the several downsampling operations in the models and resizing image, we lose finer-grained information. This leads to a loss of accuracy on smaller object classes such as pool and vehicles. Based on these observations, we proposed using DeepLabv3, PSPNet, and SegFormer in our work. These models address the challenges as explained previously. SegFormer performed the best achieving a test IoU of 56.78 with MiT-B4 encoder and 54.86 with MiT-B0 encoder. Qualitatively, Figure 10 shows the better performance of our proposed models against the baselines. We also note specific improvements made by SegFormer over PSPNet and DeepLabv3 in idetifying flooded classes, smaller objects, and producing a finer-detailed mask along object boundaries. We noticed that the baseline models tended to classify blue-appearing buildings and objects to water. Hence, we tried data augmentation using color jitter. However, we did not notice any significant improvements in this case.

Our initial results are promising, and we see a general trend of improvement in our metrics for the validation and test set. Running our models for more epochs and continuing with a learning rate schedule will improve our results for this problem. We also restricted our pre-trained backbones to models with less than 30 million parameters to reduce convergence time. Using bigger backbones will improve the feature mapping from the input, and running them for longer can potentially improve performance. Additionally, our current loss weighs all classes uniformly. Our results and the literature saw flooded vs non-flooded distinction, and smaller classes are notoriously hard to identify. Using custom loss functions that focus on these more complex tasks will possibly improve our mean and class-wise results. Additionally, we tried colour jittering to augment our dataset. It will be interesting to see how our model performance changes with alternative augmentation strategies. We can also try to break our image into overlapping patches, which would free us from having to downsize the input image to 512. This should improve the performance for smaller classes, and the overlap would ensure that the patches do not miss out on the all-important context.

7. Contributions and Acknowledgements

Kushagra Gupta scoped out the project and finalized the dataset for the problem. He worked on designing and writing the architecture for a smaller UNet as a baseline for the problem. He used pre-trained backbones for transfer learning from PyTorch and Segmentation-Models-PyTorch to make FCN and PSPNet architectures while optimizing from the available backbones under the constraints of GPU compute time.

Priya Mishra worked on setting up the PyTorch and PyTorch Lightning based training, analysis, and visualization scripts used in our work. She worked on setting up the SegFormer model from HuggingFace library and training it on our dataset. She used the pre-trained backbones from torchvision to train the DeepLabv3 model using the ResNet-50 backbone. She worked on using data augmentation with SegFormer models.

Kushagra Gupta worked with the World Bank as part of the course CME 291: Master's research and used the ICME-GPU cluster for shorter runs for hyperparameter tuning and experimentation with models. Priya Mishra is a part of the Future Data Group led by Prof. Matei Zaharia and used the Future cluster for running her experiments.

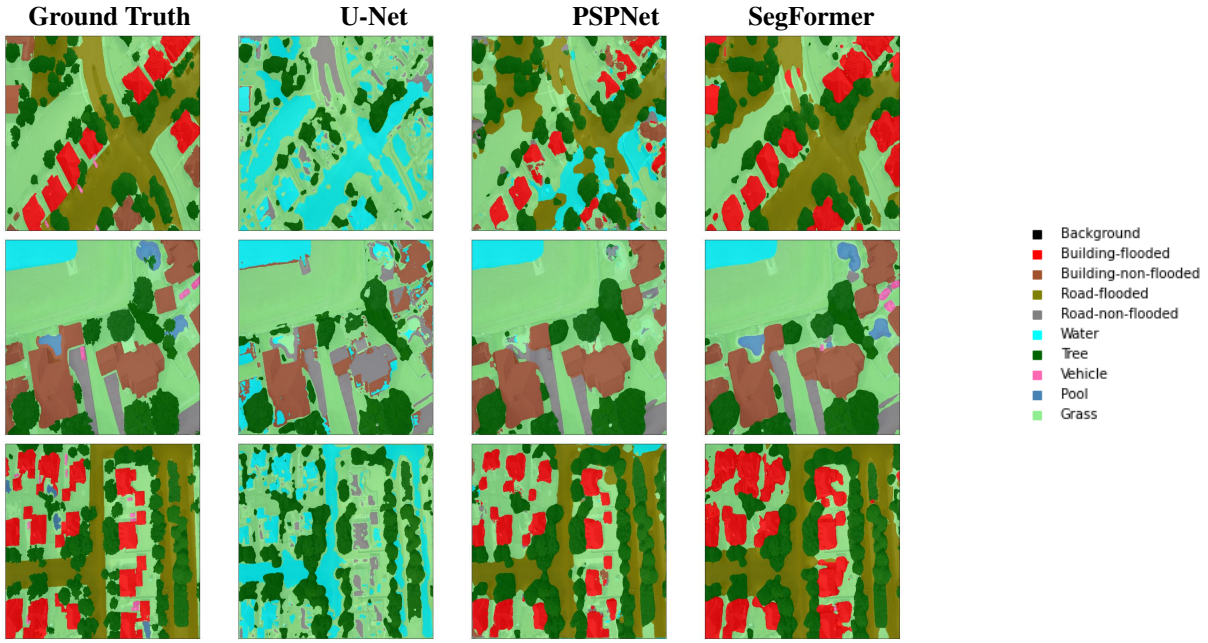

Figure 6. Ground Truth and Segmentation Masks for various models in our work.

References

[1] Frontline : Preparing healthcare systems for shocks from disasters to pandemics. https://openknowledge.worldbank.org/handle/10986/35429. 1

[2] Bischke Benjamin, Helber Patrick, Schulze Christian, Srinivasan Venkat, Dengel Andreas, and Borth Damian. The multimedia satellite task at mediaeval 2017. In *MediaEval Benchmark*, September 2017. 1

[3] Lukas Biewald. Experiment tracking with weights and biases, 2020. Software available from wandb.com. 5

[4] Liang-Chieh Chen, George Papandreou, Florian Schroff, and Hartwig Adam. Rethinking atrous convolution for semantic image segmentation. *arXiv preprint arXiv:1706.05587*, 2017. 2, 5

[5] Liang-Chieh Chen, Yukun Zhu, George Papandreou, Florian Schroff, and Hartwig Adam. Encoder-decoder with atrous separable convolution for semantic image segmentation. In *ECCV*, 2018. 2

[6] William Falcon et al. Pytorch lightning. *GitHub. Note: https://github.com/PyTorchLightning/pytorch-lightning*, 3, 2019. 5

[7] Sahil S Khose, Abhiraj Tiwari, and Ankita Ghosh. Semi-supervised classification and segmentation on high resolution aerial images. In *NeurIPS 2021 Workshop on Tackling Climate Change with Machine Learning*, 2021. 3

[8] Wei Li, Huasheng Zhu, Xiangsheng Feng, and Fen Li. Semantic segmentation-based algorithm for urban road waterlogging disaster detection. In *ICVIP 2021: The 5th International Conference on Video and Image Processing, Hayward, CA, USA, December 22 - 25, 2021*, pages 104–110. ACM, 2021. 2, 4

[9] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015. 2, 3, 12

[10] Sébastien Marcel and Yann Rodriguez. Torchvision the machine-vision package of torch. In *Proceedings of the 18th ACM International Conference on Multimedia*, MM '10, page 1485–1488, New York, NY, USA, 2010. Association for Computing Machinery. 3

[11] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary 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. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019. 5

[12] Yalong Pi, Nipun D. Nath, and Amir H. Behzadan. Detection and semantic segmentation of disaster damage in uav footage. *Journal of Computing in Civil Engineering*, 35(2):04020063, 2021. 2

[13] Maryam Rahnemoonfar, Tashnim Chowdhury, Robin Murphy, and Odair Fernandes. Comprehensive semantic segmentation on high resolution uav imagery for natural disaster damage assessment, 2020. 2, 4

[14] Maryam Rahnemoonfar, Tashnim Chowdhury, Argho Sarkar, Debvrat Varshney, Masoud Yari, and Robin Roberson Murphy. Floodnet: A high resolution aerial imagery

dataset for post flood scene understanding. *IEEE Access*, 9:89644–89654, 2021. 1, 2

[15] Maryam Rahnemoonfar, Robin Murphy, Marina Vicens Miquel, Dugan Dobbs, and Ashton Adams. Flooded area detection from uav images based on densely connected recurrent neural networks. In *IGARSS 2018 - 2018 IEEE International Geoscience and Remote Sensing Symposium*, pages 1788–1791, 2018. 2

[16] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In Nassir Navab, Joachim Hornegger, William M. Wells, and Alejandro F. Frangi, editors, *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*, pages 234–241, Cham, 2015. Springer International Publishing. 2, 3, 12

[17] C. Szegedy, Wei Liu, Yangqing Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–9, Los Alamitos, CA, USA, jun 2015. IEEE Computer Society. 4

[18] Ethan Weber, Nuria Marzo, Dim P. Papadopoulos, Aritro Biswas, Agata Lapedriza, Ferda Ofli, Muhammad Imran, and Antonio Torralba. Detecting natural disasters, damage, and incidents in the wild. In *Computer Vision – ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XIX*, page 331–350, Berlin, Heidelberg, 2020. Springer-Verlag. 1

[19] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online, Oct. 2020. Association for Computational Linguistics. 5

[20] Enze Xie, Wenhai Wang, Zhiding Yu, Anima Anandkumar, Jose M Alvarez, and Ping Luo. Segformer: Simple and efficient design for semantic segmentation with transformers. *Advances in Neural Information Processing Systems*, 34, 2021. 2, 6

[21] Pavel Yakubovskiy. Segmentation models pytorch. https://github.com/qubvel/segmentation_models.pytorch, 2020. 5

[22] Hengshuang Zhao, Jianping Shi, Xiaojuan Qi, Xiaogang Wang, and Jiaya Jia. Pyramid scene parsing network. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6230–6239, 2017. 2, 4, 5

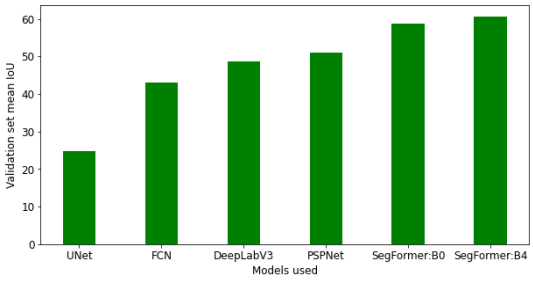

Figure 7. Validation set mean IoU.

8. Appendix

9. Class wise results

Summarizes the class wise results in tables 4 and 5.

10. Validation mean results

Plots the validations set results in figure 7.

11. Model architectures

Visualizes the model architectures in figures 9 and 8.

11.1. Additional Segmentation Masks

Class	Subset	UNet	FCN	DeepLabv3	PSPNet	SegFormer (MiT-B0)	SegFormer (MiT-B4)
Background	Val	0.00	0.97	0.12	36.44	49.36	44.21
	Test	0.00	0.61	0.00	27.30	39.07	30.58
Building-flooded	Val	0.00	49.07	58.87	53.90	58.28	56.44
	Test	0.00	39.88	45.80	41.95	44.73	44.31
Building-non-flooded	Val	14.90	62.27	72.54	69.99	69.54	70.50
	Test	18.16	63.97	71.30	71.48	68.51	69.80
Road-flooded	Val	0.00	37.28	37.29	42.18	51.77	49.49
	Test	0.00	31.95	39.92	44.71	43.54	44.50
Road-non-flooded	Val	46.82	67.21	78.16	76.71	77.67	78.04
	Test	44.85	66.56	77.45	77.91	76.41	77.62
Water	Val	44.52	60.84	73.28	66.66	71.07	70.38
	Test	45.45	62.63	75.81	73.55	75.17	74.77
Tree	Val	66.09	70.43	79.23	77.84	78.63	79.49
	Test	61.39	67.33	79.76	78.53	78.63	78.95
Vehicle	Val	0.00	0.00	0.00	0.00	0.00	22.85
	Test	0.00	0.00	0.00	0.00	0.00	19.35
Pool	Val	0.00	0.85	0.00	0.00	44.90	48.15
	Test	0.00	1.69	0.00	0.00	35.21	40.35
Grass	Val	76.60	81.39	87.31	86.02	86.35	86.61
	Test	75.85	81.51	88.39	87.67	87.30	87.57

Table 4. Class-wise IoU on validation and test sets of FloodNet.

Class	Subset	SegFormer (MiT-B0)	SegFormer (MiT-B4)
Background	Val	40.72	3.771
	Test	18.72	22.38
Building-flooded	Val	58.81	74.41
	Test	46.17	71.85
Building-non-flooded	Val	71.36	69.88
	Test	71.05	76.74
Road-flooded	Val	50.64	51.32
	Test	44.89	60.26
Road-non-flooded	Val	77.87	83.65
	Test	76.05	82.22
Water	Val	71.86	54.06
	Test	74.03	52.79
Tree	Val	79.09	65.67
	Test	78.69	82.53
Vehicle	Val	0.00	36.15
	Test	0.00	30.50
Pool	Val	43.64	76.28
	Test	0.00	0.00
Grass	Val	86.71	83.91
	Test	86.93	83.35

Table 5. Class-wise IoU on validation and test sets of FloodNet with data augmentation.

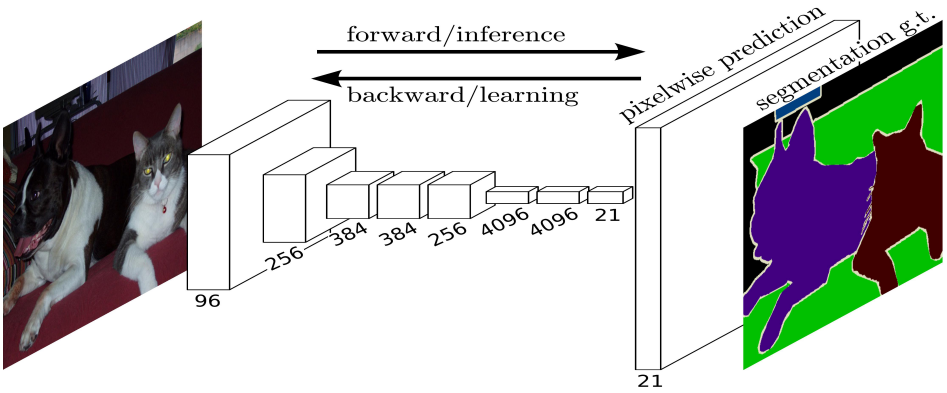

Figure 8. FCN architecture. Source: [9]

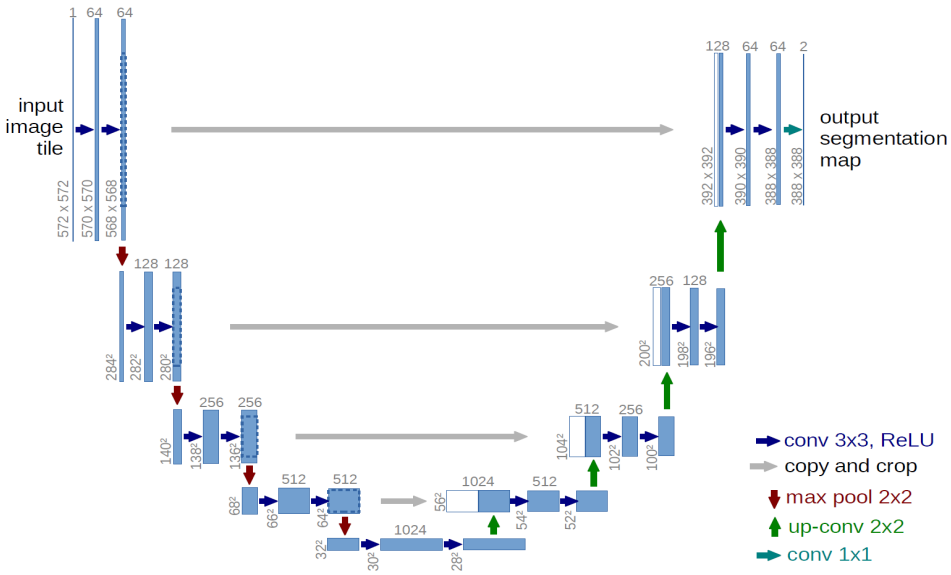

Figure 9. UNet architecture. Source: [16]

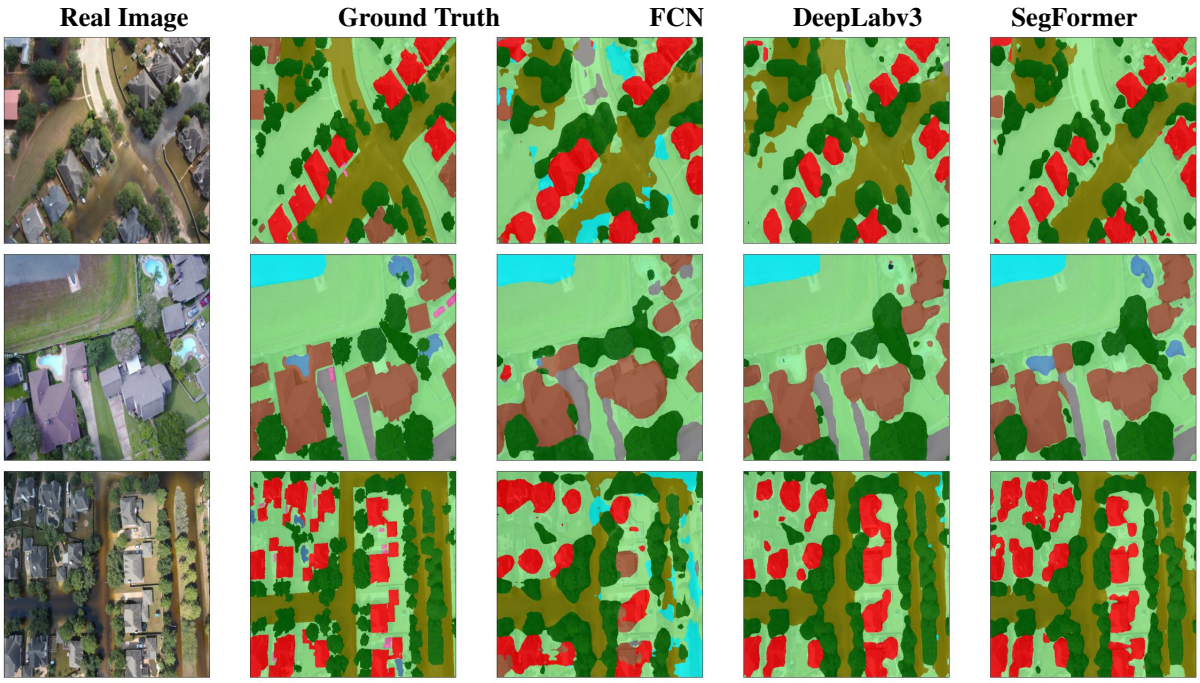

Figure 10. Ground Truth and Segmentation Masks for various models in our work.