Road Extraction from Satellite Images

Saksham Jindal
jindal.saksham@outlook.com

I. Overview

Road Extraction and land cover mapping, in general, from satellite imagery, is an important tool for monitoring and efficient map generation for an intelligent transportation system, automobile navigation and emergency support in times of natural disasters. With the recent advancement of technology in satellite imagery, remote sensing systems can provide data with very high spatial resolution which allows us to have high precision ground information and permit large-scale monitoring of roads. Automating road extraction plays an important role in dynamic spatial development and plays an important role in large scale mapping and urban planning.

Image segmentation methods for automated road extraction and can be broadly classified into - classical computer vision techniques and machine learning algorithms. Traditional image processing techniques for image segmentation such as thresholding and edge detection have been successful in mapping outlines of the roads but require extensive manual intervention. On the other hand, we have seen a wide adoption of deep learning technologies owing to acceleration in development of computing technologies and promising scientific research in machine learning in the last decade.

This document investigates the use of deep learning methods trained with labelled satellite images for automated labelling of road pixels on aerial images with semantic labels.

II. About the Dataset

Massachusetts Roads Dataset [1] consists of 1171 aerial images of the state of Massushussets, each 1500 x 1500 pixels in size. We had only 804 labelled images in the training set which is split into a set of 643 images for training and 161 for validation. Further, the algorithms discussed are evaluated on a test set of 14 images.

III. Methodology

We use deep learning based semantic segmentation network architectures to train satellite images labelled with masks at a pixel level. The following document experiments with mainly four training strategies while training the models -

A) Data Normalisation
B) Data Augmentation
C) Loss Functions
D) Schedulers and Optimizers
E) Network Architectures.

The following section will aim to give a brief walkthrough the training techniques adopted for experimentation as a part of converging on an approach to segment the satellite images.

A. Data Preparation

The pixel values in the images must be scaled prior to providing the images as input to a deep learning neural network model during the training or evaluation of the model. This ensures each input parameter (pixel, in the case) has a similar data distribution which makes convergence faster. Also, networks process inputs using small weight values and inputs with large integer values can slow or disrupts the learning process,

1) Normalisation : The pixel intensity values in the images, present in 8-bit format, were normalised to bring the pixel values in the range 0-1 by dividing by 255.
2) Centering : Further, the images processed by subtracting per-channel mean from pixel values calculated on the training set
3) Standardisation : Finally, the values obtained by subtracting per-channel mean were divided by per-channel standard deviation calculate on the training set

This ensures that distribution of pixel values follows a normal distribution with mean 0 and standard deviation 1. In the current setup, per channel mean and per channel standard are pre-calculated for the training set and we obtain mean values and standard deviation values of () and () for RGB channels respectively.

B. Data Augmentation

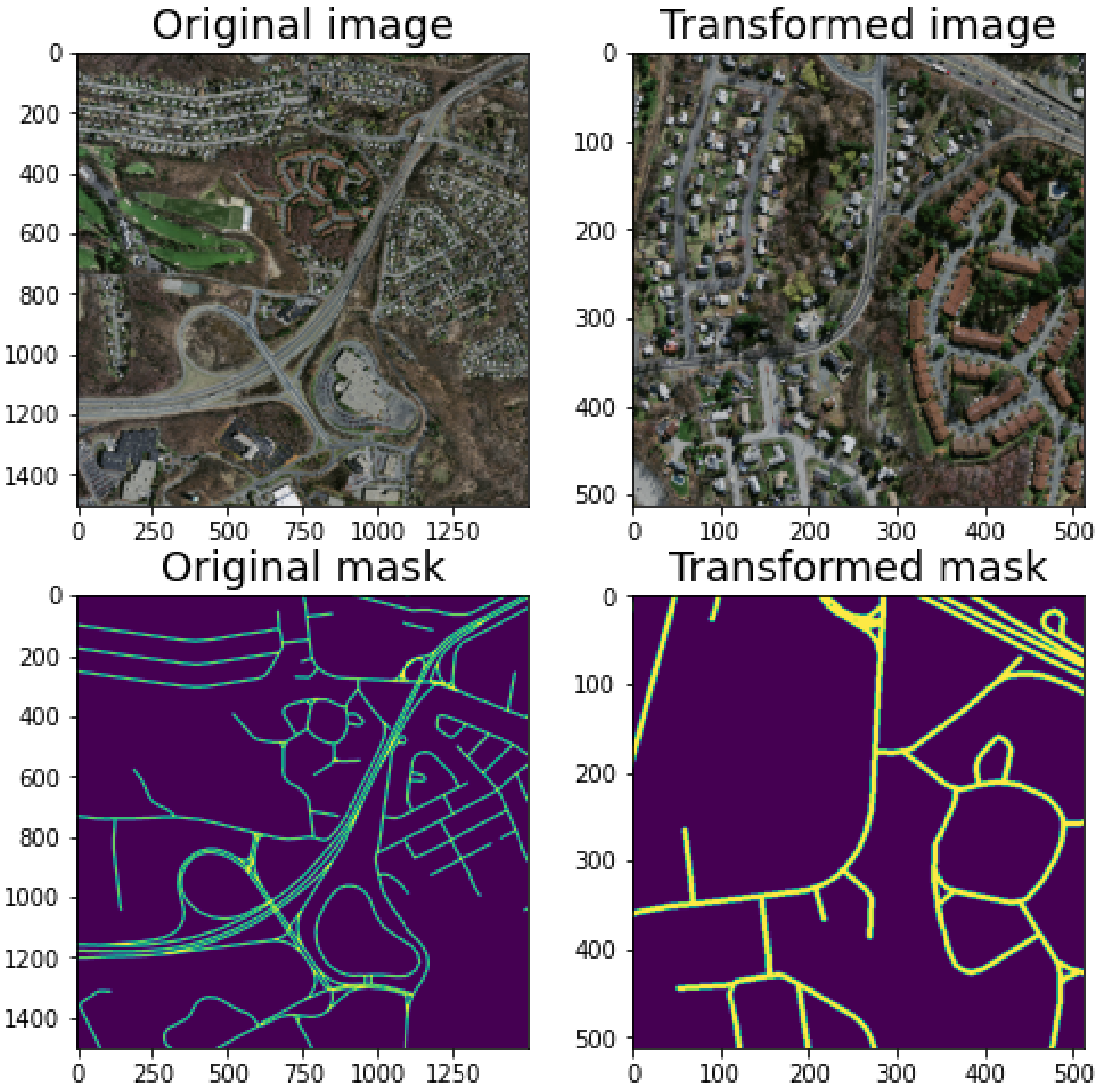
Data Augmentation is a strategy that enables practitioners to significantly increase the diversity of data available, without actually collecting new data. It aims to address 2 important requirements while training supervised models - diversity of training data and amount of data. There are 2 ways which the data augmentation can be achieved in our pipeline - offline augmentation (perform all necessary augmentations beforehand) and online augmentations (performing augmentation on a mini-batch just before feeding data to the model)

In this document, we have performed online augmentations using albumentations library [2]. Following augmentations were applied on each mini-batch of training data before loading to feed into the model

1) Random Crop - Images are randomly cropped at 512 x 512 pixel size
2) Random Horizontal Flip (with probability 0.5)
3) Random Vertical Flip (with probability 0.5)
4) Random Rotate (with probability 0.5)

5) Transpose (with probability
6) Random Shift-Scale-Rotate
7) Random Brightness and Contrast added/removed
8) Random Gamma transformations
9) Random Blur (with probability of 1%)

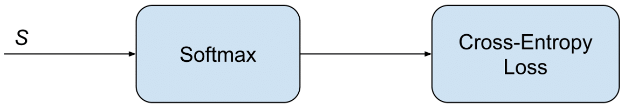
An illustration of augmentation performed on an input image of size 1500 x 1500 pixels to get augmentation on output image size of 512 x 512 pixels can be seen in the following set of images. A detailed breakdown of the augmentations performed is illustrated in the jupyter notebook attached with the solution or can be accessed by clicking here.



C. Loss Functions

Semantic Segmentation is a classification task on pixel level. The choice of loss function or objective function in non-convex optimisation is extremely important as it influences

1) Categorical Cross Entropy Loss : The most commonly used loss function for the task of image segmentation is a pixel-wise cross entropy loss. In the current setup, we have assigned a class label of 0 to background and 1 to roads and the aim to



$$f(s)_i = \frac{e^{s_i}}{\sum_j^C e^{s_j}} \qquad CE = -\sum_i^C t_i log(f(s)_i)$$

Gently borrowed from the source

2) Weighted Categorical Cross Entropy Loss

Since cross entropy loss (discussed above) evaluates class predictions for each pixel individually and then averages over all the pixels, it asserts equal learning to each pixel in the image. This could be a problem if a class is underrepresented in unbalanced datasets.

As discussed in the previous sections, it was observed that there is a class imbalance of 1:20 pixels in the training dataset, which means that for every 100 randomly selected pixels, there are only 5 pixels that correspond to roads and the rest of them belong to background class.

In the present document, we have examined two weighing schemes that can be used to compensate for class imbalance for road pixels

a) Inverse Number of Samples

$$w_{n,c} = 1/(Number\ of\ samples\ in\ Class\ C)$$

b) Inverse of Square Root of Number of Samples

$$w_{n,c} = 1/\sqrt{(Number\ of\ samples\ in\ Class\ C)}$$

3) Focal Loss : It applies a modulating term to the cross entropy loss in order to focus learning on hard negative examples. It is a dynamically scaled cross entropy loss, where the scaling factor decays to zero as confidence in the correct class increases.

$$\mathrm{FL}(p_t) = -(1 - p_t)^{\gamma} \log(p_t)$$

4) Dice Loss (or Jaccard Loss) : Dice loss for image segmentation is based on dice coefficients, which is essentially a measure of overlap between 2 samples. This measure ranges from 0 to 1 where a dice coefficient of 1 denotes perfect and complete overlap.

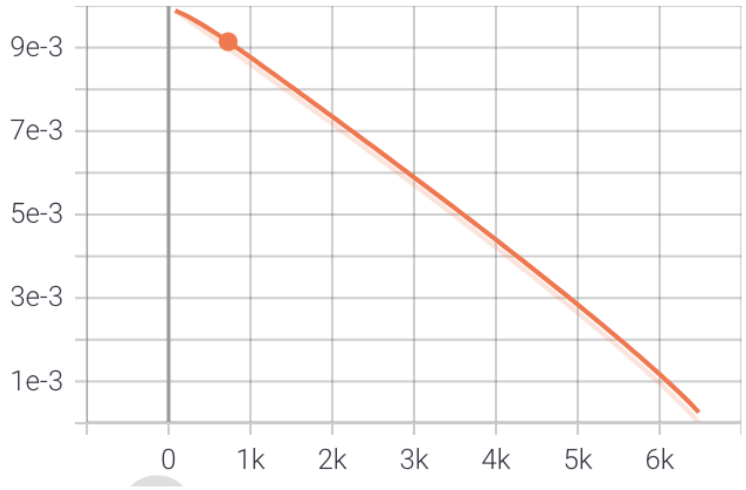
$$DICE = 2\frac{\sum_i y_i p_i}{\sum_u y_i + \sum p_i}$$

where *y* and *p* are ground truth labels and probabilities associated with model outputs

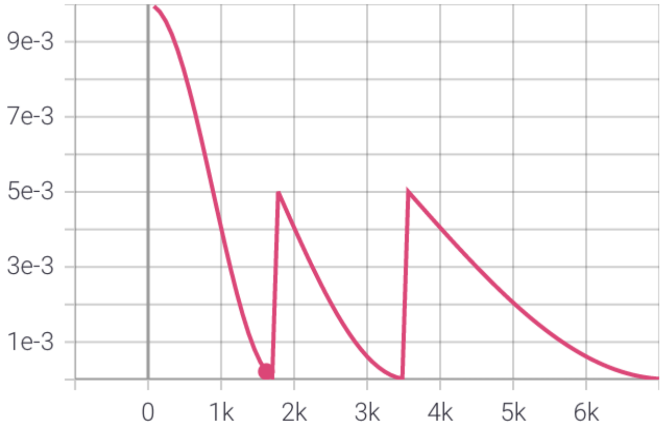
The major reason people try to use dice-coefficient or IOU directly is that the actual goal is maximisation of those metrics and cross entropy is just a proxy which is easier to maximise. An important point to be noted here, is that, since this is an *n*-class segmentation (where n = 2), the final dice loss is calculated for each class separately and then averages to yield a final score.

D. Optimizers and Schedulers

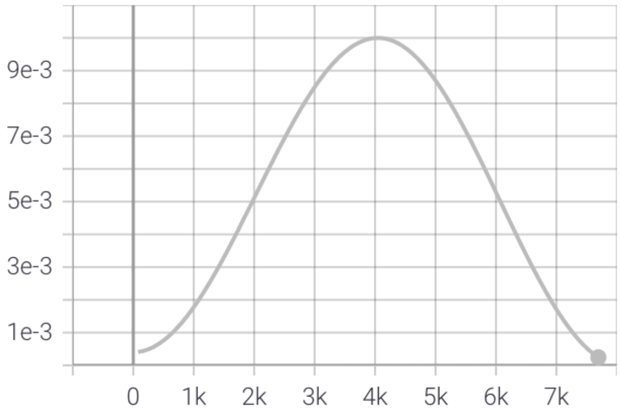
1) Stochastic Gradient Descent (Optimiser) with Polynomial LR (Scheduler)



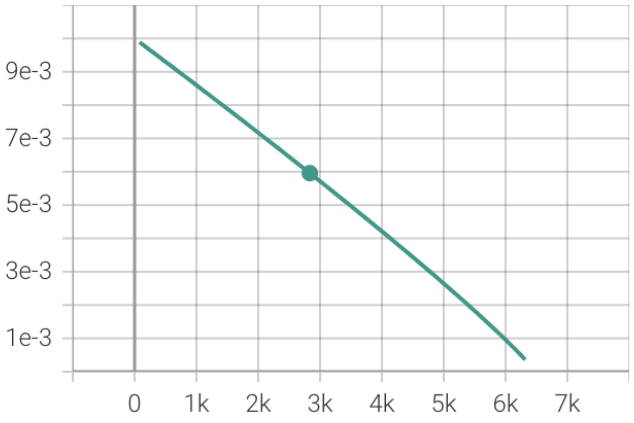
2) Stochastic Gradient Descent (Optimiser) with Warm Restarts (Scheduler)



3) Stochastic Gradient Descent (Optimiser) with One Cycle LR (Scheduler)



4) Adam Optimiser (max lr decaying by polynomial factor)

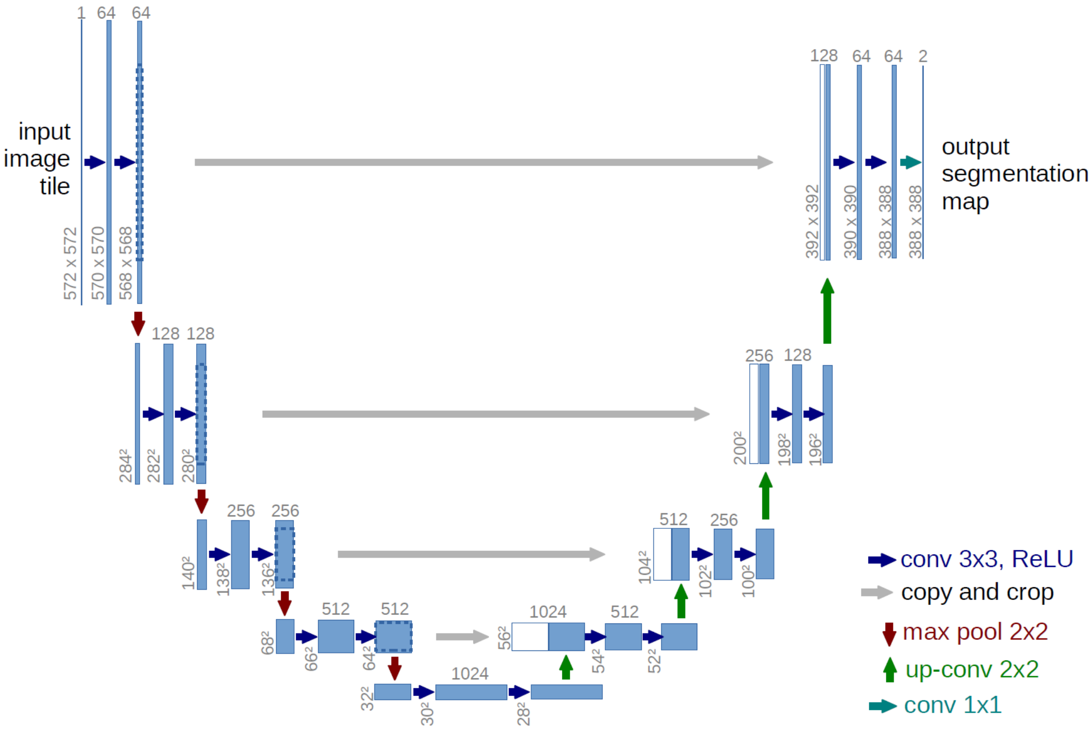


E. Network Architectures

1) UNet-Resnet50

UNet architecture revolutionized segmentation using long skip connections between each level of contracting path and expanding path. It’s like FCN is pulled upwards from both ends.

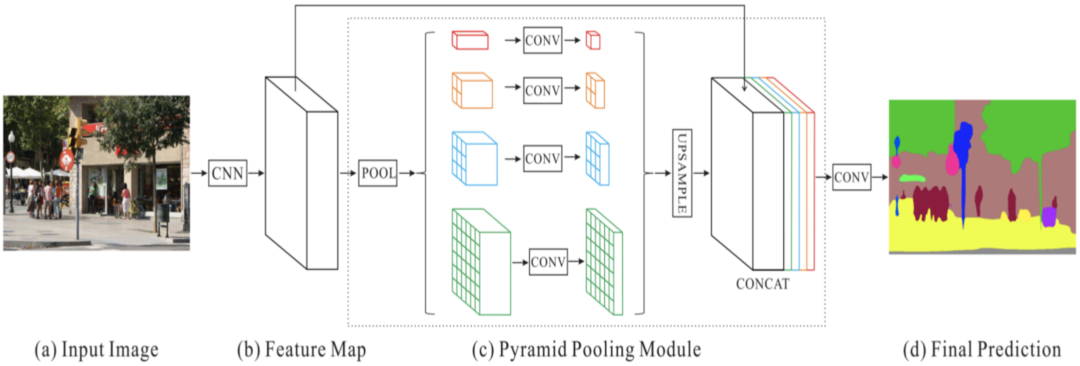
Another revolutionary advancement in computer vision was ResNet. The residual blocks in ResNet with skip connections helped in making a deeper and deeper convolution neural network and achieved record-breaking results for classification on the ImageNet dataset.



A specimen of UNet (with VGG backbone) from original paper [3]

2) PSPNet (Resnet-50 backbone)

PSPnet or Pyramid Scene Parsing Network is one of the well recognised semantic segmentation architectures. PSPNet architecture improved upon the FCN by taking into account the global context of the image to predict the local level predictions. It's encoder comprises dilated convolutions which helps in increasing the receptive field. Pyramid Pooling Module is the main part of the network as it helps the model to capture the global context in the image which helps it to classify the pixels based on global information present in the image.

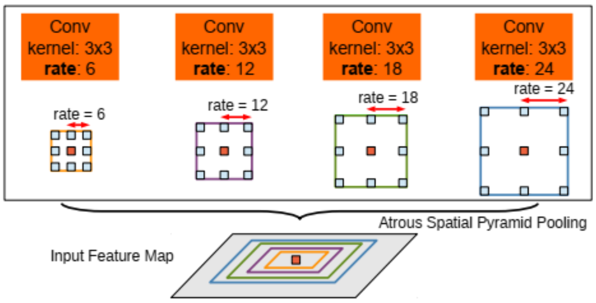


PSPNet Architecture

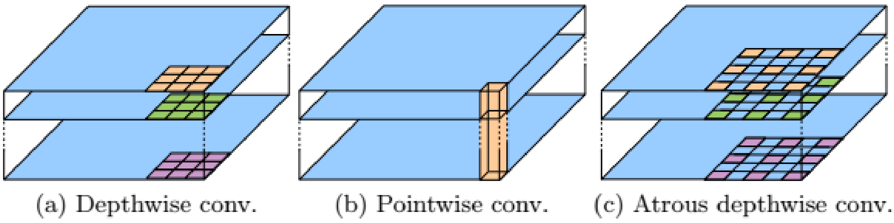
3) Deeplab V3+

Deeplab v3+ was invented at Google and belongs to the family of Deeplab series and precursors to the architecture included Deeplab v1, Deeplab v2 and Deeplab v3. Deeplabv3+ uses novel encode and decoder which are reported to capture sharp boundaries, aggregate information at multiple scales,

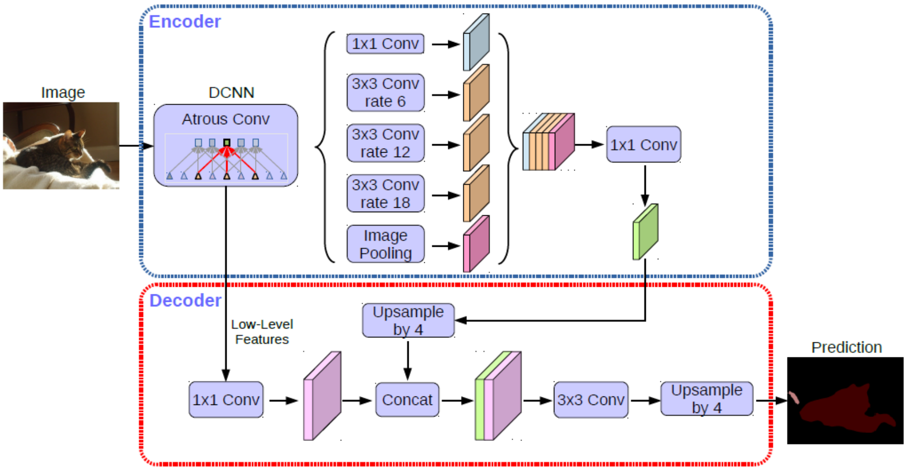
a) **Atrous Spatial Pyramid Pooling (ASPP)** - able to encode multi-scale contextual information. The idea is to apply multiple atrous convolution with different sampling rates to the input feature map, and fuse together.



b) **Depth-wise separable convolution** : It separates out the depth of the feature map and applies a depthwise convolution followed by 1x1 convolution to combine outputs from the depth-wise convolution. It helps in achieving computational efficiency.



c) Deeplabv3+ uses a decoder different from the networks preceded it. It used Aligned Xception as encoder (not used in the report). In the decoder, instead of upsampling by the factor of 16 (input image is down-sampled by a factor of 16), the updampled features are first upsampled by a factor of 4 and concatenated with corresponding low level features from the encoder module. Further, through a series of 1x1 and 3x3 convolutions, it is again upsampled by a factor of 4.



IV. Metrics

1) Intersection over Union (IOU)

The Intersection over Union (IoU) metric, also referred to as the Jaccard index, is essentially a method to quantify the percent overlap between the target mask and our prediction output.

In segmentation tasks the IoU is prefered over accuracy as it is not as affected by the class imbalance that are inherent in foreground/background segmentation tasks. As an example, if a ground truth image is made up of 90% background pixels, a

Usually, the IoU score is calculated for each class separately and then averaged over all classes to provide a global, **mean IoU** score of our semantic segmentation prediction, but here for evaluation, **road IOU** is considered for evaluation owing to high imbalance in the dataset

$$IoU = \frac{TP}{(TP + FP + FN)}$$

2) Pixel Accuracy

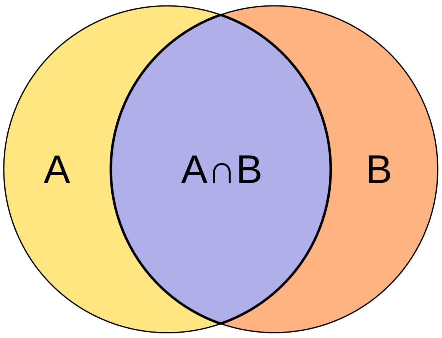
Measure of percentage of pixels in the image which were correctly classified. The pixel accuracy is commonly reported for each class separately as well as globally across all classes.

This metric can sometimes provide misleading results in cases of class imbalance as the measure will be biased in mainly reporting the negative class or background class if it is predominant

$$accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

3) Dice Score (or F1 score)

Dice Coefficient is twice the area of overlap divided by the total number of pixels in both images. It is very similar to Jaccard’s Index (IOU) . Dice coefficient double counts the intersection(TP). Both the Dice and Jaccard indices are bounded between 0 and 1. Dice score is also called F1 score which is the harmonic mean of precision and recall is by its definition well suited for unbalanced datasets.



$$Dice(A,B) = \frac{2\|A \cap B\|}{\|A\| + \|B\|}, \qquad Jaccard(A,B) = \frac{\|A \cap B\|}{\|A \cup B\|}$$

$$Dice = \frac{2TP}{2TP + FP + FN}, \qquad Jaccard = IoU = \frac{TP}{TP + FP + FN}$$

4) Precision-Recall

Precision is a measure of out of predicted positives, how many are classified correctly.

$$Precision = \frac{TP}{TP + FP}$$

Recall is a measure of out of actual positives, how many are classified correctly

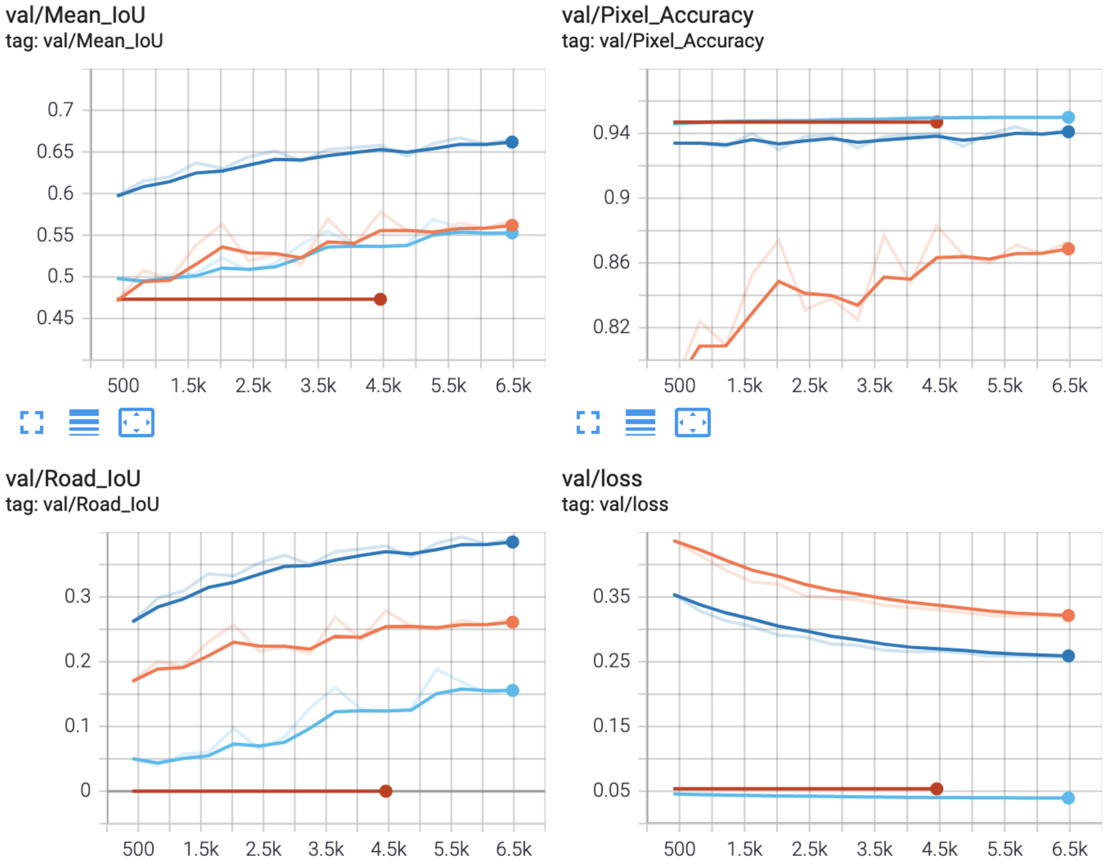
$$Recall = \frac{TP}{TP + FN}$$

V. Experiments & Results

A. Choice of Loss Function

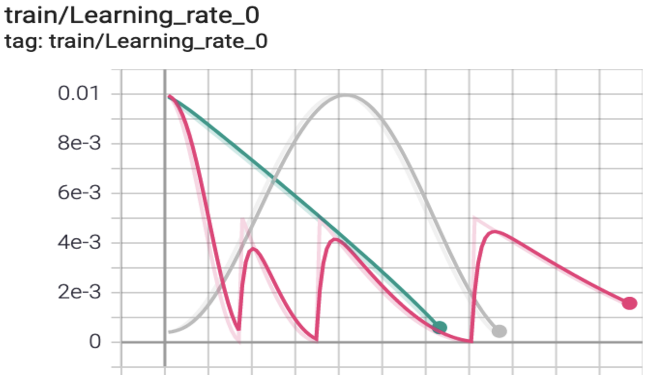
Over Experiment 1, Experiment 2 (weighted cross entropy loss), Experiment 3 (dice loss) and Experiment 4 (focal loss), the loss functions were varied with a Unet with Resnet-50 backbone.

During an initial analysis, it was observed that the ratio of pixels belonging to the road and background is 1:20 (**class imbalance**). The weighing of the cross entropy loss was experimented with 1:20 and 1/√20 and it was observed that using inverse square root of pixel distributions exceeded results on the Class IOU (Road IOU) being monitored. Further, it was found to be performing much better than Dice Loss and Focal Loss. It can be argued that weighed cross entropy loss performs better than dice loss and focal loss because the former optimised for the overall segmentation IOU and focal loss has been reported to perform better when the class imbalance is of the order of 1:1000



Exp. 1 (Weighted CE - 1:20) - Orange, Exp. 2 (Weighed CE - 1:5) - Blue,
Exp. 3 Dice Loss, Exp. 4 - Focal Loss

B. Choice of Optimizer and Scheduler

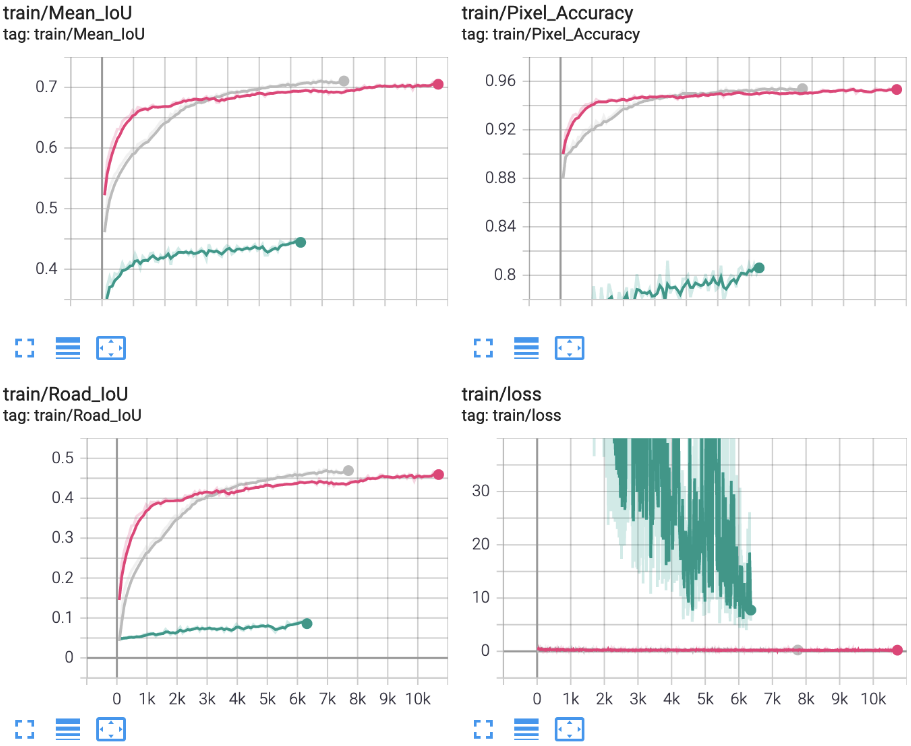


The performance of Stochastic Gradient Descent with Cosine Annealing Warm Restarts (cyclic LR) , SGD with One CycleLR (superconvergence) and Adam were compared using Resnet-50 backbone of Unet and cross entropy loss weighted in 1:5 proportions.

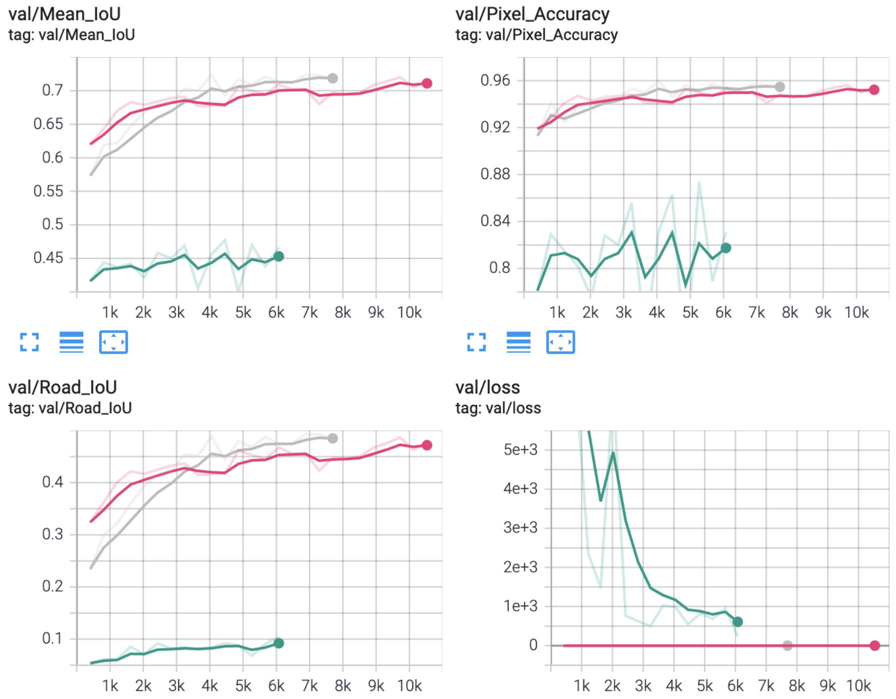
For Experiment 5, using SGD with Cosine Annealing rates, maximum and minimum LR are important hyperparameters used in the experiments. They are calculated by using the “LR range test” as discussed in Leslie Smith’s paper on Cyclic LR. Also, the learning rate is “warm restarted” at 15%, 45% and 90% of the total iterations (found by trial-and-error)

For Experiment 6, SGD with One Cycle LR anneals the learning rate from an initial learning rate (base lr same as min lr in above) to some maximum learning rate (same max lr as above) and then from that maximum learning rate to some minimum learning rate (base lr divided by a factor of 25). The learning rate was increased from base lr to max lr during phase 1 (40% of iterations) and decreased to base LR/25

For Experiment 7, instead of “SGD-scheduler” combination, adaptive learning rate technique - Adam is used instead of SGD.



Exp 5 : Pink - SGD with Warm Restarts, Exp 6: Grey - SGD with One Cycle LR and Exp 7 : Grey : Adam optimizer - Training Plots

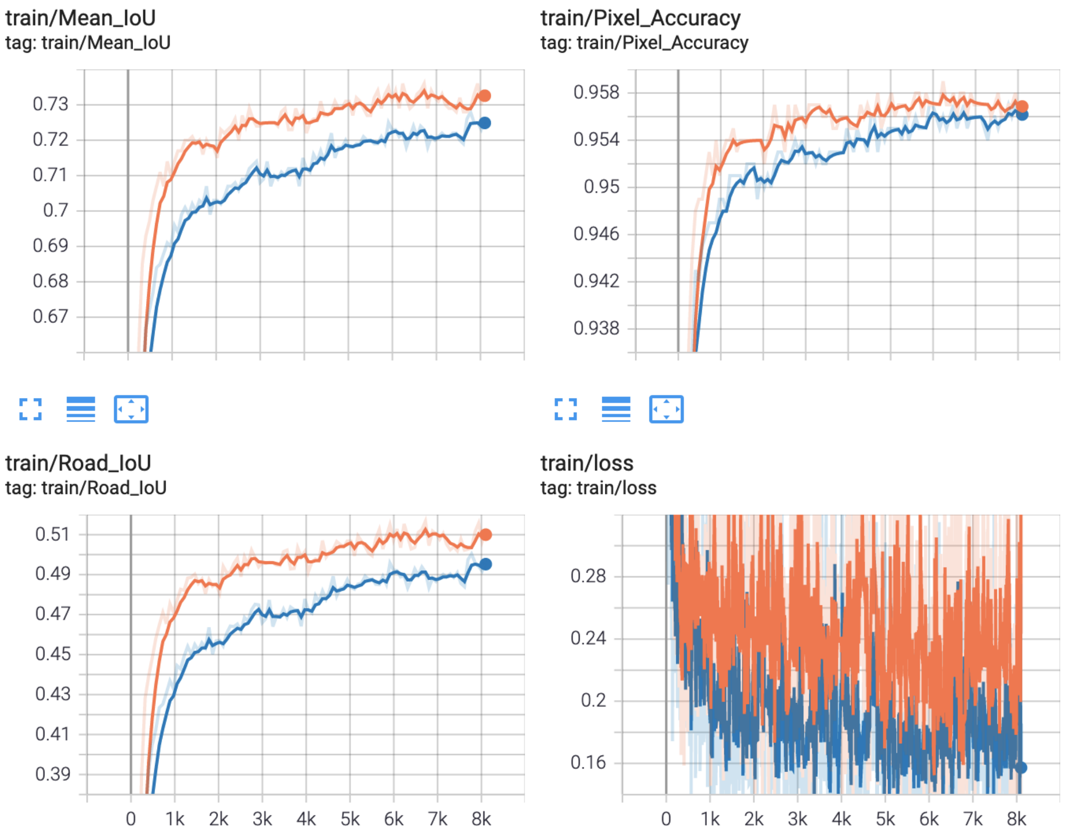


Exp 5 : Pink - SGD with Warm Restarts, Exp 6: Grey - SGD with One Cycle LR and Exp 7 : Grey : Adam optimizer - Validation Plots

There are few interesting observations that could be made from the above training and validation plots. First, Warm Restarts Scheduler (Exp 5; Pink) converges much faster than One Cycle LR policy. Second, with careful selection of hyperparameters, both the schedulers seem to converge to, approximately, similar values. Finally, contrary to expectations, the performance of Adam pales in comparison to the above experiment (probably due to high max LR).

C. Choice of Architecture

Using SGD with Warm Restarts Scheduler and Weighed CE Loss (1:5), performance of UNet, PSPNet and Deeplabv3+ (all Resnet-50 backbones) were compared during the Experiments 5, 8 and 9. We observe that performance of Deeplabv3+ far exceeds UNet and PSPNet on class-wise IOU calculated.

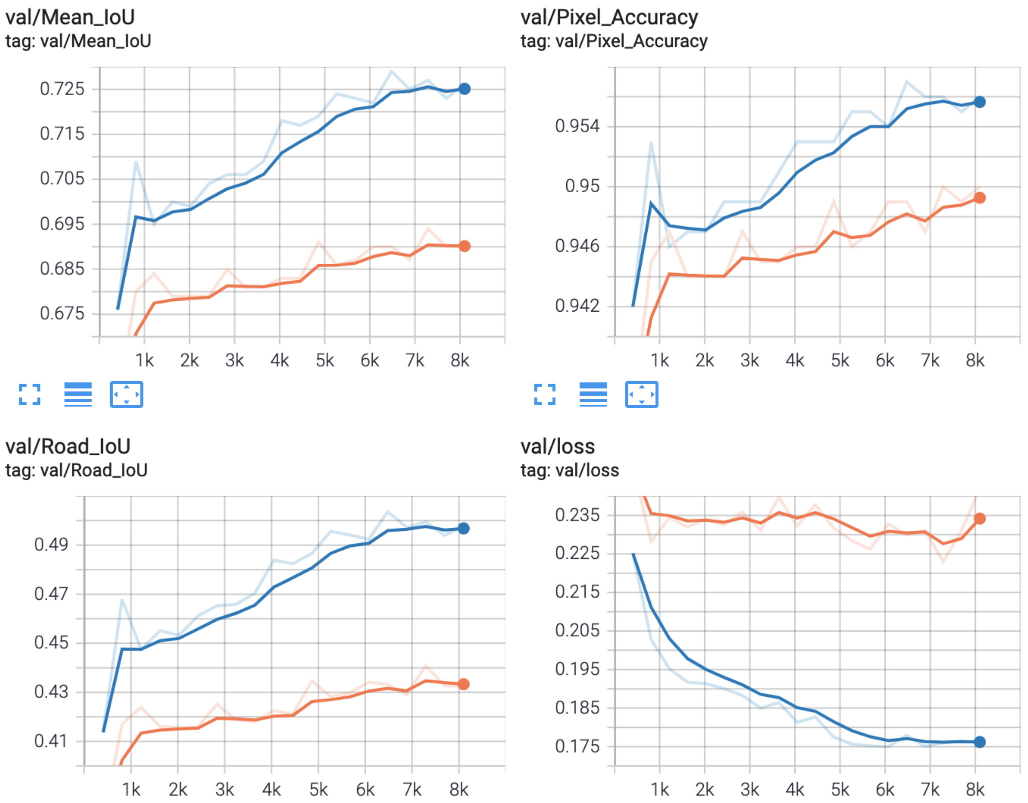

train/Mean_IoU
tag: train/Mean_IoU

train/Pixel_Accuracy
tag: train/Pixel_Accuracy

train/Road_IoU
tag: train/Road_IoU

train/loss
tag: train/loss

Exp 8 : Orange - PSPNet, Exp 9 : Blue - Deeplabv3+ - Training Plots


val/Mean_IoU
tag: val/Mean_IoU

val/Pixel_Accuracy
tag: val/Pixel_Accuracy

val/Road_IoU
tag: val/Road_IoU

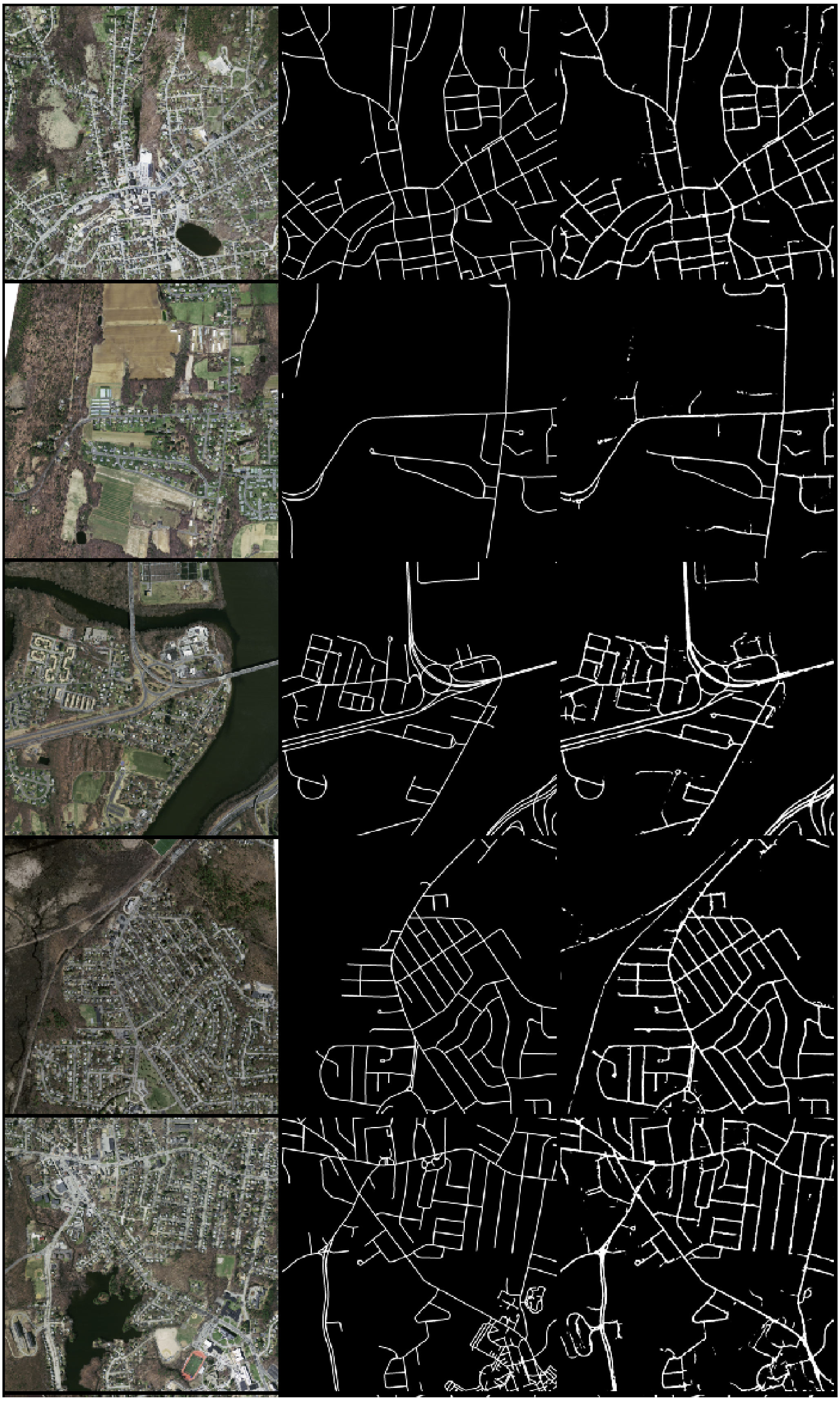
val/loss
tag: val/loss

Exp 8 : Orange - PSPNet, Exp 9 : Blue - Deeplabv3+ - Validation Plots

An overall summary of the experiments has been documented in the following table.

Experiment Name	*Architecture*	*Loss*	*Optimizer Scheduler*	*Road IOU*	*Pixel Accuracy*	*Dice Score*	*Precision*	*Recall*
Exp 1	Unet-Resnet50	Weighted CE (1:20)	SGD with Poly LR scheduler	0.283	0.847	0.440	0.296	0.859
Exp 2	Unet-Resnet50	Weighted CE (1:5)	SGD with Poly LR scheduler	0.396	0.927	0.565	0.488	0.678
Exp 3	Unet-Resnet50	Dice Loss	SGD with Poly LR scheduler	8.4e-13	0.929	8.4e-13	1.00	8.45
Exp 4	Unet-Resnet50	Focal Loss	SGD with Poly LR scheduler	0.2098	0.934	0.336	0.600	0.248
Exp 5	Unet-Resnet50	Weighted CE (1:5)	SGD with Cosine Annealing Warm Restarts	0.502	0.945	0.667	0.583	0.783
Exp 6	Unet-Resnet50	Weighted CE (1:5)	SGD with OneCycle LR	0.497	0.932	0.657	0.579	0.781
Exp 7	Unet-Resnet50	Weighted CE (1:5)	Adam	0.104	0.839	0.187	0.144	0.2761
Exp 8	PSPNet	Weighted CE (1:5)	SGD with Cosine Annealing Warm Restarts	0.4528	0.9380	0.6222	0.5423	0.737
Exp 9	Deeplabv3+	Weighted CE (1:5)	SGD with Cosine Annealing Warm Restarts	0.5101	0.9450	0.6742	0.574	0.8201

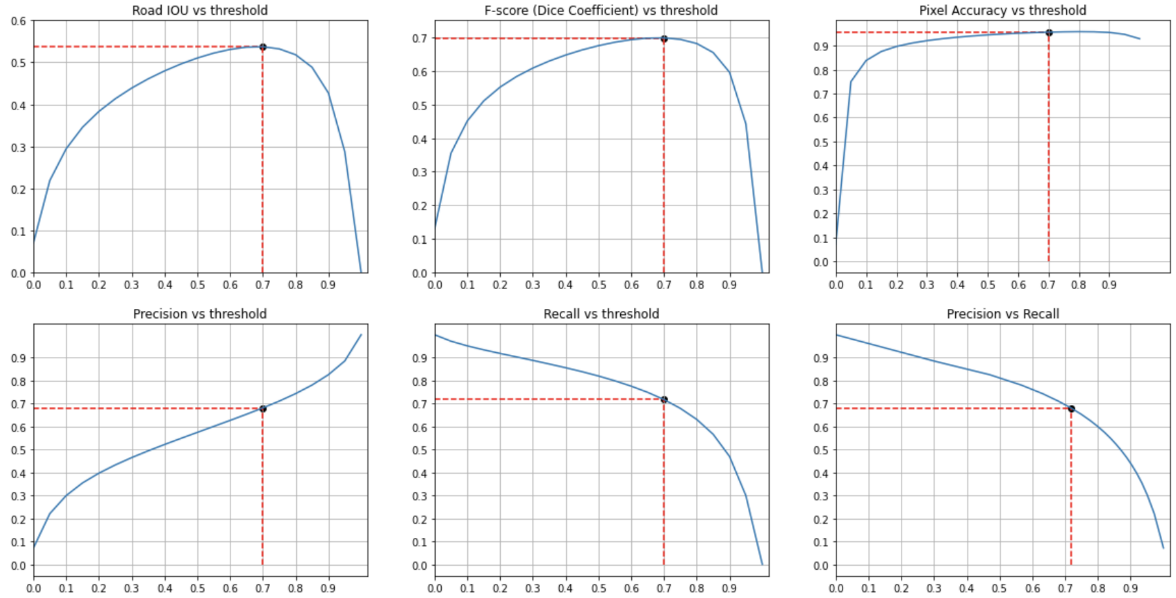
**Metrics reported on the test set by softmaxing the logits and thresholding probabilities at 0.5*



Visualization of Predictions and Comparison. Image (left), Ground Truth (center) and Prediction (right)

D. Threshold Optimization

Threshold optimization is an important exercise in classification based predictive modelling, especially in cases of imbalanced data. Generally, using a default threshold (0.5) results in a sub-optimal performance for imbalanced datasets. A simple approach of doing so is to tune the threshold used to map probabilities to class labels.



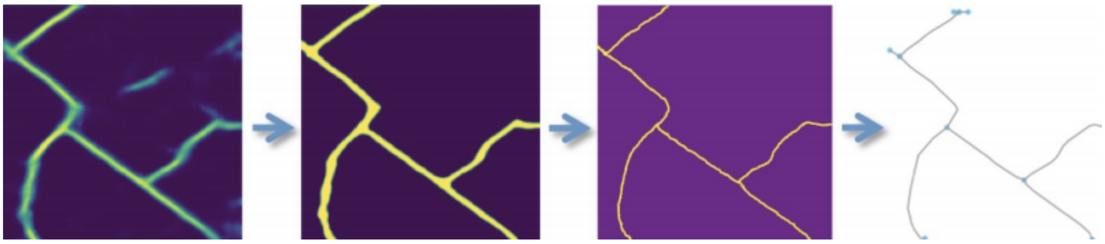
VI. Note on Post-processing

We can observe in the visualisation of predictions, there are certain “open holes”and breaks in a continuous stretch of road as documented in the above predictions. To obtain a road network, it could be an important exercise to fill in the blind spots in the mask extracted and thus obtain the masks.

As noted in the paper, *City-Scale Road Extraction from Satellite Imagery v2*, the authors use morphological transformations (erosion and dilation) in image processing for post-processing. The process of erosion involves removal of foreground object and process of dilation involves adding foreground object around the boundary.While, erosion is used for diminish the features, dilation is used for accentuate features.

a) Morphological Opening - The opening operation erodes an image and then dilates the eroded image. This could be useful for noise removal

b) Morphological Closing - The closing operation dilates an image and then erodes the dilates image.. It could be useful for closing small holes in the pixels of the road.



Approach as discussed in paper (fig. From the same paper) for post-processing

Further, a **practical** extension of the above work is the extraction road network (graph of roads as vertices and junctions as nodes). Authors in the same paper, use skeletonization as a process of reducing road pixels or *thinning* to skeletal shape that preserves the extent and connectivity of the original region. Authors further process the above skeletons to obtain graphs structure.

VII. Conclusion

Satellite imagery analysis is crucial for intelligent remote monitoring and building system to generate efficient routes. The use cases for the technology spans across humanitarian to military.

In the document, methods to road segmentation maps from satellite images has been discussed. It was observed that using carefully tuned weights in cross entropy loss, hyperparameters in the optimiser and scheduler (SGD with warm restarts) with Deeplabv3+ yields a class wise IOU of 0.51 on the testing dataset. Further, the document discusses post-processing techniques - thresholding, closing, opening and skeletonization to obtain road network graphs.

VIII. References

[1] *V. Mnih, "Machine learning for aerial image labeling*," Ph.D. disser-tation, University of Toronto, 2013.
[2] *Alexander Buslaev, Alex Parinov, Eugene Khvedchenya,Vladimir I Iglovikov, and Alexandr A Kalinin.* Albumenta-tions: fast and flexible image augmentations. arXiv preprintarXiv:1809.06839, 201
[3] *Ronneberger, O., Fischer, P., Brox, T.*: U-net: Convolutional networks for biomedical image segmentation. In: MICCAI. (2015)
[4] *Zhao, H., Shi, J., Qi, X., Wang, X., Jia, J.*: Pyramid scene parsing network. In: CVPR. (2017)
[5] *Liang-Chieh Chen and Yukun Zhu and George Papandreou and Florian Schroff and Hartwig Adam* : Encoder-Decoder with Atrous Separable Convolution for Semantic Image Segmentation, ECCV (2018)

[6] *L. Smith* : Cyclical Learning Rates for Training Neural Networks (2017)
[7] *I. Loshchilov, F. Hutter* : SGDR: Stochastic Gradient Descent with Warm Restarts (2017)
[8] *L. Smith, N. Topin* : Super-Convergence: Very Fast Training of Neural Networks Using Large Learning Rates (2018)
[9] *A. V. Etten* : City-Scale Road Extraction from Satellite Imagery v2: Road Speeds and Travel Times (2019)