A comparison between fully connected and deconvolutional layers for road segmentation from satellite imagery

Dhruv Bhugwan
School of Computer Science and Applied Mathematics
University of the Witwatersrand
Johannesburg, South Africa

Pravesh Ranchod
School of Computer Science and Applied Mathematics
University of the Witwatersrand
Johannesburg, South Africa

Richard Klein
School of Computer Science and Applied Mathematics
University of the Witwatersrand
Johannesburg, South Africa

Benjamin Rosman
School of Computer Science and Applied Mathematics
University of the Witwatersrand
Johannesburg, South Africa
Mobile Intelligent Autonomous Systems,
Council for Scientific and Industrial Research
Pretoria, South Africa

***Abstract*—Semantic segmentation using fully convolutional networks has quickly become a popular solution as they provide very accurate per pixel classification. However, the implementation of deconvolutional layers and their mechanics differ greatly to those of patch based segmentation using convolutional neural networks. Both techniques have been used for road segmentation from satellite imagery but never compared. Thus we investigate the difference between fully connected and deconvolutional layers and provide an interpretation as to the correlation and differences between each methodology for road segmentation from satellite imagery.**

***Index Terms*—fully convolutional networks, segmentation, convolutional neural networks**

I. Introduction

Object detection in an image is the problem of identifying the location and size of the bounding box for the object of interest in the image. The bounding box does not always exclusively contain just the object but also the background. In contrast, image segmentation is the problem of partitioning an image into segments which have similar properties allowing it to be analyzed in a simpler manner [1]. Semantic segmentation is an extension of image segmentation in which pixels in an image are separated into a set of classes.

Although the concept of identifying shapes and outlines in images may seem simple, finding solutions requires large volumes of data and long periods of time to produce acceptable results due to differing configurations that these shapes may appear. A consequence of this problem is that some classifiers do not perform well on objects where the background can differ for the same object of interest [2].

Superpixels are a grouping of more than one pixel in an image. Superpixels are commonly used in region based classification, whereby superpixels are used as inputs to classification models and the output class is applied to each component pixel. convolutional neural networks (convNets, CNNs) are neural networks that implement convolutional operations, and are most commonly used to process image data and to isolate spatial relationships from images, allowing them to effectively classify image data.

Recent implementations of semantic segmentation models involve superpixels and convolutional neural networks [3] which requires multiple passes through a classification model. Some extensions to convolutional neural networks such as fully convolutional networks have become a popular segmentation model in which pixelwise classification is achieved by passing the image through the network in which fully connected layers are dropped in favor of deconvolutional layers [4]. Extensions to the fully convolutional network architecture [5] mitigate the need for very large datasets and improve spatial resolution of the segmented image, but they do disregard fully connected layers in favor of deconvolutional layers and thus all conventional convolutional neural network methods as possible solutions.

Patch based segmentation is the process of striding through an image with windows of a fixed size allowing each window to be passed through a convNet performing per pixel classification and stitching each predicted patch back together so as to obtain a segmented image. A number of studies have found patch based segmentation to be effective [6], but the computational cost of processing the individual patches is prohibitive.

The problem domain involves segmenting roads from satellite images as they have a wide variety of applications in many differing fields such as being used as a means of path planning for aerial drones, automated road map making, disaster relief etc. [7]. The goal is to identify certain characteristics of roads and making them apparent to the observer but we

implement two differing methods as a means of contrasting their effectiveness. We implement fully convolutional network and patch based convNet segmentation models as a means of finding all road pixels from input satellite imagery.

We implement the convNet model proposed in [7] as our base architecture, we then implement the method proposed by [4] to convert the base convNet to a fully convolutional network (FCN). From there we provide a comparison between performance of the two models, which allows us to understand more deeply the implications of implementing fully convolutional networks over other convNet methodologies. We compare the time taken for inference and accuracy between a convNet and adapted fully convolutional networks on the dataset provided in [6]. We also implement a pseudo random patch based selection in our training methodology as a means of mitigating the effects of minority classes in datasets.

The rest of this paper is structured as follows. Section II outlines related works that use convNet for aerial imagery segmentation. We discuss fully convolutional networks and how they can be constructed from adapting existing convNet. In section III we outline our proposed method which consists of our base architecture, the method for adaptation and our learning criterion. Section IV details our dataset and comparisons made between our convNet and adapted FCN. In section V we summarize our findings.

II. Related Works

Methods of applying artificial neural networks to aerial imagery as a means of pixelwise classification have already been proposed and have achieved acceptable results [6]–[8]. Noise is a common problem in most image processing and vision related tasks and can arise from lens distortion, faulty photo receptors or even in file transfer. This used to be a problem especially due to computational restrictions. However, due to the large number of neurons and feature maps in convolutional neural networks , noise omission in post processing is unneeded as the features learned allow the network to represent labels in such a way that noise in output labels are in the minority and have little effect on the prediction results of such systems [6].

Both [6] and [7] approached the problem by training on 64x64 input patches with outputs of each network being a vector of length 256 reshaped into 16x16 patches. These output patches are the segmented 16x16 center patch of the input image. [6] reasoned that this would improve prediction on patches as the network will have been trained to use features in a larger spatial vicinity to that of the output patch. A constraint of the above mentioned architecture is that the input image will have to be divided into patches determined by the architecture of the network implemented.

An approach to accuracy improvement and prediction smoothing was implemented by [7] called model averaging with spatial displacement. Their implementation made use of eight models with the same architecture trained for the same number of epochs with the only difference being that each model was initialized with different weights. Prediction was implemented in the same manner as training where the output of the network represents the center patch of the input image, but each model receives the same input patch but with a spatial displacement d ($0 \leq d \leq 7$) in both the vertical and horizontal direction. The predicted label patches are then tiled and divided by eight for averaging.

Popular convNets for segmentation such as [9], [6] and [7] implement patchwise training methods which as mentioned earlier can be expensive. The method proposed by [4] allows for prediction of any image on any scale without the need for partitioning. They suggest a method for adapting existing convNet classifiers and a technique of fusing learned features across layers to improve segmentation detail. The techniques suggested involve removing all fully connected layers and replacing them with a convolutional layer with a 1x1 kernel effectively linearizing all inputs from the previous layer. Subsequently they introduce deconvolutional layers as a means of upsampling the outputs from previous layer. But [4] found that this caused a great loss in segmentation detail and that predictions would look similar to region based classification. As a means of addressing this issue they implemented a skip architecture which combines outputs from earlier convolutional layers with that of the output from a later deconvolutional layer. This allows for the features detected in the image to be integrated with the later layers before all spatial context from the original scene was lost.

Data augmentation is the process of applying label-preserving transformations, wherein spatial context of the image is not lost after applying the transformation to an image [2]. If such a transform is applied to an image with a low probability, features will not be learned over the same set of images repeatedly. Thus it prevents overfitting if smaller datasets are to be used. This concept may be thought of as equivalent to increasing the dataset size. Elastic distortions have also been used for data augmentation [10]. The method involves applying filters to an image which produce feasible scenes such as stretching an image and thus elongating features, or applying a matrix to the image which will simulate distortions at random points in the image.

III. Proposed Method

In an effort to compare fully connected and deconvolutional layers we first construct a base convNet and train it using the same parameters as [7] to allow for fair comparison. We then follow [4]'s methodology to produce a FCN. Thereafter we train the adapted model using the same parameters and methodology as the initial model. The goal of both models is to perform pixelwise classification on an RGB input image where the output is a single channel image predicting if a pixel was a road or background. We outline the base architecture and techniques used to perform our experiment and comparisons below.

A. Base Architecture

The base architecture of the base covnet [7] consists of three convolutional layers followed by two fully connected layers.

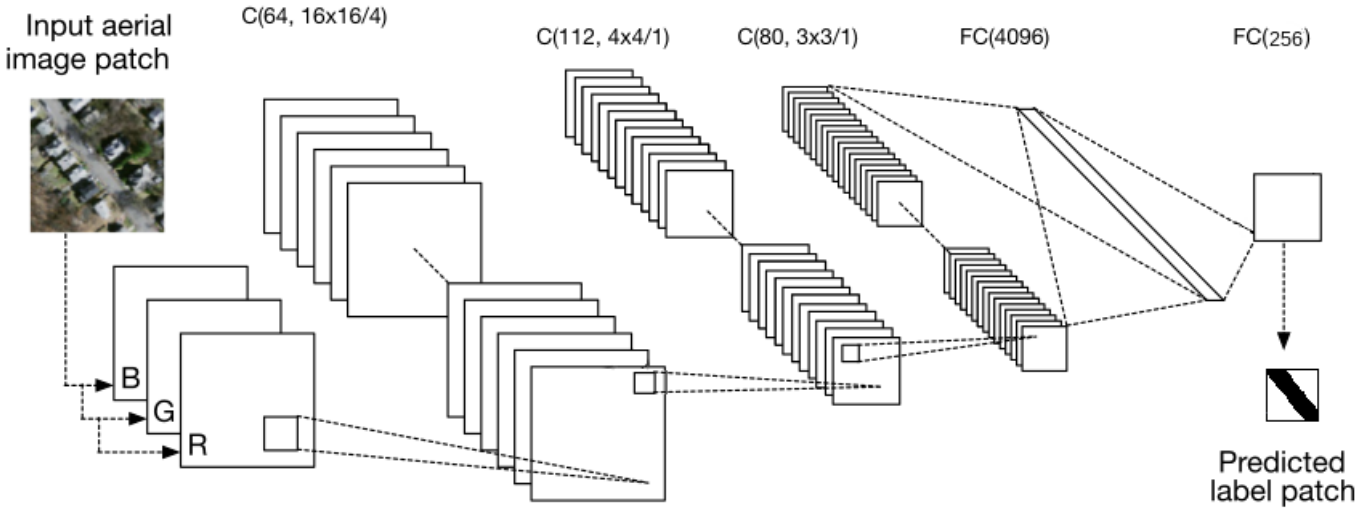

Fig. 1. The base CNN architecture used in this article (adapted from [7])

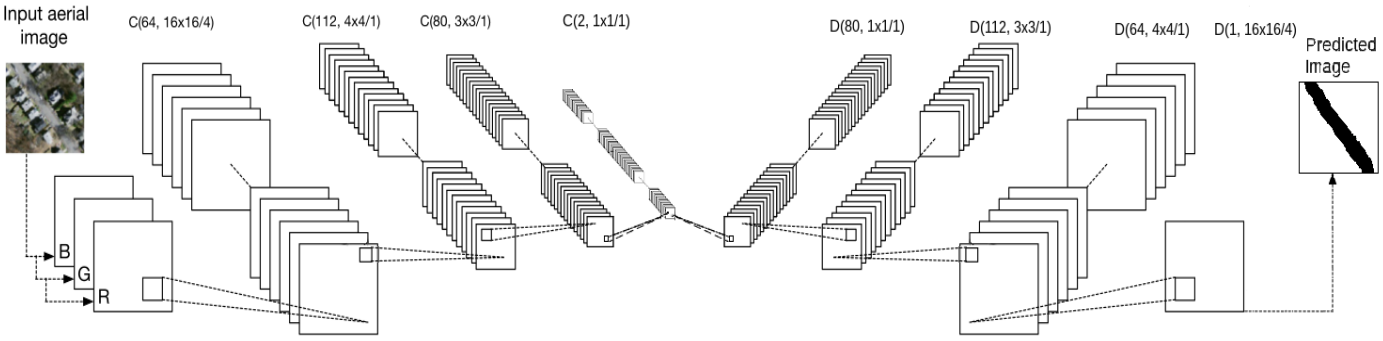

Fig. 2. Our FCN model.

All five of the layers consist of trainable parameters. The input is a 64x64-sized RGB patch sampled from the dataset used in [6] and [7]. The output of the network is a 256x1 vector which is reshaped to a 16x16 patch representing the segmented road and background center pixels of the input image.

The convolution and deconvolutional layers are described as $C/D(n, f\text{x}f/s)$ where n denotes the number of filters, f describes their width and height and s describes the stride. The fully connected layers are described by $FC(k)$, where k are the number of neurons in the fully connected layer. From figure 1, we have 256 output neurons and we reshape those into a 16x16 matrix which represents the segmented center patch of the input image.

B. Adaptation for FCN Segmentation

Considering the above architecture mentioned in figure 1 we implement [4]'s methodology for converting convNet to FCNs. From the trained model convNet in figure 1, we remove the fully connected layers, then place a 1x1 convolutional layer at the end as per the first step of conversion from [4]. Thereafter we upsample by stacking deconvolutional layers and implementing skip connections. Skip connections are used as we are working with data which is spatially small, in order to preserve features in upsampling [4] we implemented skip connections at every level. This adapted network is shown in figure 2.

The adaptation implements 4 deconvolutional layers after the 1x1 convolution. The first deconvolutional layer upsamples its input, the output from that layer will be of the same dimension to that of the third convolutional layer allowing us to implement a skip connection between those two layers. The second deconvolutional layer again upsamples its input. The output will be of the same dimensionality to that of the second convolutional layer and thus we implement another skip connection to recover spatial context. The third deconvolutional layer used before the final prediction layer is implemented so that we may upsample its input from the second deconvolutoinal layer as means of constructing a skip connection to the first convolutional layer. This allows us to integrate all learned features from all of our convolutional layers. Finally our final layer upsamples its input to a patch of the original width and height.

C. Learning

1) Biased Patch Based Sampling: [4], [6], [8] trained using sampled patches from images in their dataset in an

effort to emphasize common groupings of features in close proximity. From our findings this was effective at emphasizing common groupings, but led to the potential disappearance of minority classes as it is possible for these to never occur in randomly sampled patches. In order to avoid this problem, we biased the sampling process to ensure that minority classes occurred in each sampled patch. In particular we notice that from our training masks that the road pixels only represent approximately 4% of our training data, making it unlikely that they will appear in a majority of the sampled patches. Thus in training both networks we implement a sampling methodology in which every patch selected for our minibatch must consist of a minimum of one road pixel.

2) Data Augmentation: We applied horizontal and vertical flipping along with rotation by random angles before our patches are sampled from the larger images.

3) Hyperparameters: Both implemented models were trained using the following hyperparameters adapted from [6]:

- Number of patches sampled per epoch: 131072
- Batch Size: 128
- Epochs: 100
- Learning Rate: 0.001
- Dropout Rate: 0.5
- Momentum: 0.9
- Weight Decay: 0.001

Each sampled patch was of size 64x64 and they were sampled randomly from our dataset which consists of images with a 1500x1500 resolution. Thus the large number was chosen as a means of spanning all images in our training dataset.

4) Weighted Binary Cross-Entropy Loss: Due to the imbalance between our two classes background and road we implement a weighted binary cross entropy loss function where the weights are chosen based on the ratio between the amount of road and background pixels per minibatch. The formulas used are:

$$ratio = \frac{n_r}{n_t}$$

$$loss = \Sigma_i \Sigma_j [g_{i,j} log(p_{i,j}) + ratio * (1 - g_{i,j}) log(1 - p_{i,j})]$$

Where n_r is the number of road pixels sampled and n_t is the total number of pixels in a specific minibatch. $g_{i,j}$ represents the ground truth and $p_{i,j}$ represents the predicted value. Thus the punishment of incorrectly predicting a road is discounted based on the number of road pixels sampled. We optimize this loss by means of stochastic gradient descent and backpropagation.

We implemented our models with PyTorch.

IV. Experiments

To provide a basis of comparison between fully connected and deconvolutional layers we train the convNet mentioned in subsection III-A using all of the parameters and techniques from section III. Thereafter we dropped both fully connected layers and added the layers mentioned in subsection III-B. We train the new model in the same manner as the base model. The first three convolutional layers that are transferred from the base model are frozen as this will provide us with a means of comparing fully connected and deconvolutional layers.

A. Dataset

We used the dataset proposed by [6] called the Massachusetts Roads dataset. The dataset consists of 1171 image-mask pairs and are separated into 1108 training images, 14 validation images and 49 test images. Each image has a resolution of 1500x1500 with a scale of 1 m^2/px. We reshaped the images to a resolution of 1440x1440 for convenience as seen in figure 4.

B. Results and Discussion

We test both our fully convolutional network and convNet on the dataset mentioned in subsection IV-A. This dataset is large enough that overfitting is not a concern when using randomly sampled patches [6] as there are simply too many possible patches.

Evaluation

We report using a common segmentation evaluation metric [4], that is pixel accuracy given by:

$$\frac{\Sigma_i \Sigma_j (p_{i,j} = g_{i,j})}{||g||}$$

where *p* represents the predicted pixel value and *g* represents the ground truth value. The first terms value will only increment if both the prediction and ground truth match. The denominator is simply the number of pixels in the ground truth image.

We provide further analysis by comparing road pixel classification histograms and confusion matrices for each model.

Classification distributions

The plots in figure 3 provide us with the classification value of each pixel in both our fully convolutional network and convNet. Comparing the two models we can see from the right histogram that the convNet implementation is fairly definite in its evaluation of road pixels as most predictions lie well above the median of predicted intensities. Observing the results from the left histogram we can see that there is no definite value that me may use as a threshold in the valid range [0,255] where by we may segregate road and background pixels. But in the case of our convNet implementation we find that most road classifications have an output of greater than 128, thus the convNet implementation we may threshold our outputs and any value greater than 128 are classified as road pixels. We explore this further in subsection IV-B.

The reason that the fully convolutional network implementation has no specific value in which we can threshold our classifier outputs is attributed to the fact that our dataset is fairly large with regards to the manner in which it is used for training. We can also attribute this to the fact that deconvolution in simple terms can be considered a learning of multiple parameters from multiple outputs as a means of obtaining the original image after convolution remove spatial context.

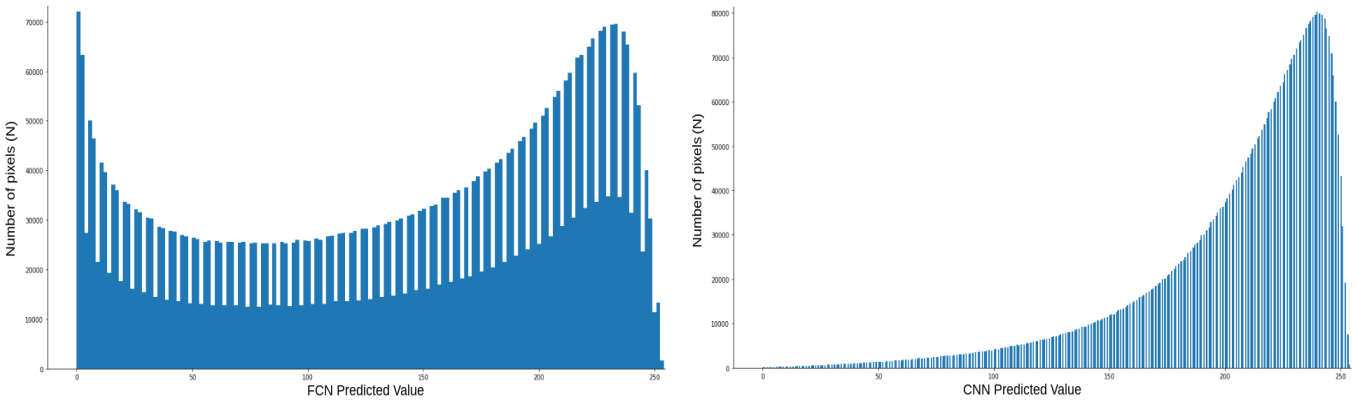

Fig. 3. The above figures represent the histogram of classification output bins for all road pixels in our testing dataset. The figure on the left represents the output bins for our fully convolutional network model, and the figure on the right corresponds to the adapted convNet model output. A potential problem is that the road predictions are much harder to visually identify in the fully convolutional network model.

Fig. 4. Original Image

Fig. 5. Ground Truth

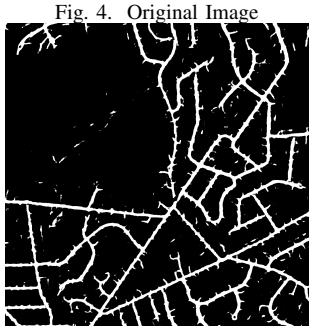

Fig. 6. CNN Prediction

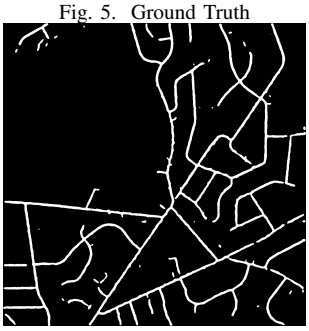

Fig. 7. FCN Prediction

The problem with this is that there are many orientations in which roads can follow, as such the parameters learned in the deconvolutional layers are very slow to converge as we need to upsample from data which has lost all spatial context and since we randomly sample with augmentation instead of train on an entire image, the features detected from one epoch may not occur in the next.

Time taken for inference

The time taken per model for inference over a single image in our test set can be considered as a distinguishing point of the fully convolutional network implementation. In order to perform inference on the convNet implementation we passed 64x64 patches from the top left corner of the image with strides of 16 as our network yielded the center 16x16 segmented patch of an input image, thereafter we combined the predicted images into a final segmented image as seen in figure 6. The our fully convolutional network model can take in an image of any size by design [4] thus inference was performed without any need for pre or post processing, the results can be seen in figure 7. This timing comparison was implemented as passing arbitrary sized inputs to a fully convolutional network classifier it is a distinguishing characteristic compared to the patch based approach. Table I gives the times taken for inference per image, observing the time taken for the fully convolutional network implementation we see that it is approximately *50* times faster than that of the patch based convNet approach. This is attributed to the partitioning and stitching of images so that they may be passed to the convNet processed and merged to form a prediction on an image of non-input size. This is the largest limitation to the patch based approach as the fully connected layers have a fixed size input.

CNN	**FCN**
0.201322s	0.00406776s

TABLE I
INFERENCE TIME PER IMAGE

Accuracy Comparisons

In figure 3 we compared how well each model could classify a road pixel. We found that the fully convolutional network model had no minimum value at which it would begin to classify road pixels with any certainty, whereas the patch based convNet approach yields a neat relationship between the output bin and the number of road pixels.

If we observe figure 7 the fully convolutional network prediction is comparable to that of the ground truth and has less noise than that of the convNet patch based prediction in figure 6. Table II gives us the confusion matrix and pixel accuracy

	Predicted Background	Predicted Road
Background	93.96%	1.35%
Road	1.90%	2.79%
Pixel Accuracy	96%	

TABLE II
CONFUSION MATRIX AND ACCURACY ON THE TEST SET (FCN)

	Predicted Background	Predicted Road
Background	84.78%	10.53%
Road	0.30%	4.39%
Pixel Accuracy	89%	

TABLE III
CONFUSION MATRIX AND ACCURACY ON THE TEST SET (CNN)

of our fully convolutional network model over the entire test set. The fairly accurate prediction of the fully convolutional network model can be seen when observing the minimal number of false positive predictions. This can be attributed to the use of skip connections from each convolutional layer. While the fully convolutional network model does not provide a mapping that is as easily identifiable as the convNet model for roads, it does perform really well at classifying background as these pixels are only found in a small number of bins.

Figure 6 gives us the prediction for the patch based convNet implementation. This model has a significant amount of noise and the road predictions are much thicker than that of the ground truth figure 5 and the fully convolutional network prediction figure 7. This was noticed by [7] and on analysis of this we found that the model predicted roads correctly as seen by the minimal amount of false negatives in table III, but the large number of false positives and the thicker roads clearly demonstrate that the pavements are also predicted as roads. The correlation between the true positives and the almost double false positives reinforce this fact.

Precision-Recall curve comparison

Figure 8 shows the precision-recall curves for the two implemented architectures. Observing Figure 6 we have a noisy prediction, this characteristic remains regardless of the threshold used due to the lower precision and the large class imbalance which leads to a large number of false positives as demonstrated in Table III. Conversely Figure 7 presents a cleaner prediction, attributed to the higher precision. But the FCN implementation does exhibit a more gradual decrease in precision as the recall increases which is the expected result when observing the somewhat even distribution of predicted values for the FCN implementation in Figure 3.

V. CONCLUSION

In this paper we provided a comparison between fully connected and deconvolutional layers for pixelwise classification. We implemented a biased random patch based sampling methodology as a means of mitigating the class imbalance between road and background pixels. We showed how fully convolutional networks classification values differ from that of convNet, finding that while fully convolutional networks are effective at categorizing background pixels due

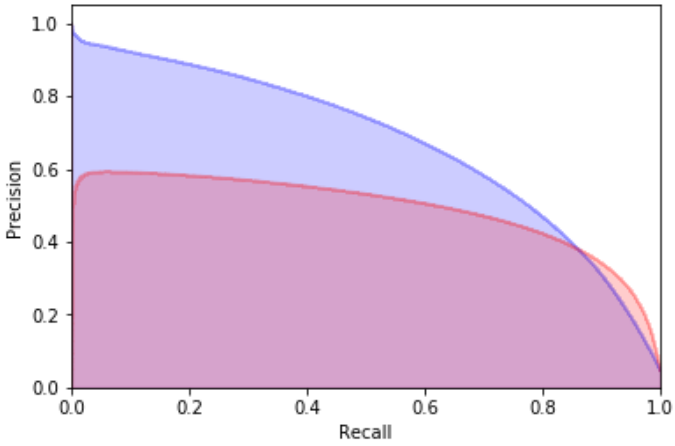

Fig. 8. Precision-recall curves for CNN (red) and FCN (Blue)

to the implementation of skip connections but they lack any deterministic value when classifying foreground pixels. We compared the inference time for the two implementations used and found that our fully convolutional network model performed inference significantly faster than that of the patch based convNet model. From this we can conclude that while both types of models have advantages in their own right, fully convolutional networks are the superior option due to their faster inference time and low false positive prediction percentages. We expect that these results will generalize to other domains if those networks are trained in a patch based fashion. The code and experiments are available at https://github.com/Alterith/Satellite_Road_Segmentation.

REFERENCES

[1] E. Anjna and E. R. Kaur, "Review of image segmentation technique," *International Journal*, vol. 8, no. 4, 2017.

[2] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097–1105.

[3] A. B. Cambra, A. Muñoz, and A. C. Murillo, "How to transfer a semantic segmentation model from autonomous driving to other domains?" in *Iberian Robotics conference*. Springer, 2017, pp. 652–665.

[4] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 3431–3440.

[5] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *International Conference on Medical image computing and computer-assisted intervention*. Springer, 2015, pp. 234–241.

[6] V. Mnih, "Machine learning for aerial image labeling," Ph.D. dissertation, University of Toronto, 2013.

[7] S. Saito, T. Yamashita, and Y. Aoki, "Multiple object extraction from aerial imagery with convolutional neural networks," *Electronic Imaging*, vol. 2016, no. 10, pp. 1–9, 2016.

[8] E. Maggiori, Y. Tarabalka, G. Charpiat, and P. Alliez, "Can semantic labeling methods generalize to any city? the inria aerial image labeling benchmark," in *IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*. IEEE, 2017.

[9] F. Ning, D. Delhomme, Y. LeCun, F. Piano, L. Bottou, and P. E. Barbano, "Toward automatic phenotyping of developing embryos from videos," *IEEE Transactions on Image Processing*, vol. 14, no. 9, pp. 1360–1371, 2005.

[10] P. Y. Simard, D. Steinkraus, J. C. Platt *et al.*, "Best practices for convolutional neural networks applied to visual document analysis." in *ICDAR*, vol. 3, 2003, pp. 958–962.