Exploring Deep Features: Deeper Fully Convolutional Neural Network for Image Segmentation

BRAC
UNIVERSITY

Inspiring Excellence

A thesis submitted in partial fulfillment of the requirements for the degree of

Bachelor of Science in Computer Science & Engineering

Authors:

Sharif Amit Kamran - 13101176

Md. Asif Bin Khaled - 12201105

Sabit Bin Kabir - 13101194

Supervisor:

Moin Mostakim

Department of Computer Science and Engineering

School of Engineering and Computer Science

Wednesday 19th April, 2017

Disclaimer

The work that has been contributed to this paper was made possible with the CAFFE deep learning framework by BVLC [8] and per BSD clause-2 license `https://github.com/BVLC/caffe/blob/master/LICENSE` it is authorized to use for any research purpose such as ours. The architecture which was adopted in this regard was made with FCN Segmentation Architecture [15]. No post or pre-processing tools were used for this work. The training was done with Pascal VOC2012 [4] and SBD [6] datasets, so proper citation must be given if the following data are used. For any future work based on this paper proper citation must be given and no one should use this for any commercial purpose other than research purpose.

Declaration

We, hereby declare that this is an original report written by us with our findings, and has not been published or presented in parts or as a whole for any other previous degree. Resources and materials by other researchers used as guidelines for our research are carefully mentioned in reference citations.

Signature of the Authors:

Sharif Amit Kamran

Md. Asif Bin Khaled

Sabit Bin Kabir

Signature of Supervisor

Moin Mostakim
Lecturer
Dept. of Computer Science and Engineering
BRAC University

Acknowledgement

We, Sharif Amit Kamran, Md. Asif Bin Khaled and Sabit Bin Kabir would like to thank Department of Computer Science & Engineering of BRAC University for giving us all the support needed to continue our research. With all the library facility, computing support and many more other resources we were able to do our thesis work with nearly no difficulties. We are grateful to our family members and friends who have supported us along the way of our journey. We would also like to give our gratitude to all the staffs who also helped us in the process of doing our research works. In this thesis work, we have compiled our four years of knowledge that we have been taught in our university life for which we are really thankful to them. Lastly but most importantly we are grateful to Mr. Moin Mostakim Lecturer, Department of Computer Science and Engineering, BRAC University. Through our research work, we have faced a lot of problems which sometimes made us disappointed and we felt the lack of confidence in ourselves but with all the people we had around us and for all the trust they put on us, we went through all the hurdles and successfully achieved our desired result.

Outlines

The thesis outline consists of six chapters in all and is outlined below. Each chapter consists of at least one or more sections that describe a specific part of that individual chapter. A detailed description of each of these sections is also outlined below.

1. **Chapter 1**

   Details of the purpose, aim, and motivation for the development of this thesis. It has three sections - introduction, motivation, and objective.

   (a) The Introduction section describes the purpose and aims for the development.

   (b) The Motivation section describes the motivation behind the whole development.

   (c) The Objective section describes our objectives.

2. **Chapter 2**

   It has three sections - problem description and fully convolutional neural network

   (a) Problem description section describes the work.

   (b) Fully convolutional neural network section describes how fully convolutional neural network actually works.

3. **Chapter 3**

   Details of the classification network and how it is used in segmentation architecture. It has three sections - classification network as segmentation architecture, transfer learning, classifier and feature map and architecture for feature extraction.

   (a) The classification network as segmentation architecture section describes how we built our fully convolutional neural network, which additional layers are included and which layers are excluded.

   (b) The transfer learning section describes how we had set up the image to image learning settings. The process of how we adjusted the batch size and the learning rate.

   (c) The Classifier and feature map section describes how per-pixel softmax loss is calculated and how mIOU is validated with the background and mean of all classes ignoring pixels that are masked out in the ground truth.

   (d) The Architecture for feature extraction section describes the segmentation architecture. How the image is upsampled and how the images are fed into the neural network.

4. **Chapter 4**

   The procedure and the result of our work are described here. It has two sections, experiment process, and experimental results.

   (a) The experiment processes describe how we used to transfer learning for weight loading and performed fine tuning with additional data.

How training was performed is also explained.

(b) The experiment result describes the result with plotted graphs.

5. **Chapter 5**

   The result of our thesis is described here. It consists of four sections, they are Metrics and Evaluation, Validation results, Hyperparameter tuning, and test results.

   (a) The Metrics and Evaluation part describes the metrics used for our model to determine the scores and the four metrics used are shown.

   (b) The validation results section describes the result of our model's score against the score of other models.

   (c) The hyperparameter tuning shows the hyperparameter tuning results for the first stage of the training in tabular format as well as the result of the second stage of the training.

   (d) The test result section displays the list of test results of mean intersection over union of all models compared to our model in a tabular format.

6. **Chapter 6**

   This chapter describes the conclusion along with the problems faced, limitations and our future work.

   (a) The section limitations describe the problems we faced while training big class datasets due to memory limitations.

(b) The section future work describes the details about the features that are to be added in future with the system.

Abstract

Classification of images has been a widely regarded challenge for the past decade, but a new type of object recognition problem which deals with pixel-level segmentation is posing a more complex task for both computer vision enthusiasts and researcher alike. The convolutional neural network has become a staple for any recognition task, but a new type of ConvNet which is Fully convolutional in architecture has yielded more fine features and proponents. We propose a neural net where we take VGG19 [20], a well-known classification CNN, make it fully convolutional for extracting deeper features and lastly use skip-architectures[15] for getting finer output. This yields better result than the pre-existing FCN segmentation architecture [15, 25, 6]. Training was done on augmented VOC12 [4] with SBD [6]training data and validation set was used from reduced VOC12 validation dataset. The model scored mIOU of 68.1 percent in PASCAL VOC 2012 Segmentation challenge.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Problem Definition

Predicting dense object is one of the foremost challenges for any instance segmentation task. But to get finer features neural nets have to explore deeper and extract those features. For that, we have to classify every pixel of an image to a certain class [7, 19, 9, 10]. Fully Convolutional neural nets [15] have shown advances in pixel level segmentation for finer feature extractions and paved the way for even further exploration of deep features. Moreover, end-to-end supervised training [15] without any pre or post processing of images [5, 6] yielded better results. On the contrary, small convolutional neural networks without supervised training [5, 6, 17] has been used before but didn't achieve any state-of-the-art results in segmentation tasks. In recent works, many neural nets have strived to find global features over local features by using multi-scale contexts, more skip architectures and successive pooling [11, 20]. Moreover, we have also seen dilated convolution being used for getting wider receptive field [23, 14]. Furthermore, plugging higher order conditional random fields [1] and using CRF as recurrent neural networks [25], also tend to enhance the accuracy and give finer feature maps if applied to already existing segmentation architecture. The model was made using VGG-19, a well-known classification neural net [11, 20, 21] which got state-of-art results in classification task, and converted the architecture to a fully convolutional neural net and added skip architecture as done in [15]. The final output was upsampled to get the original size of the input image which serves as the output of the semantic classification.

1.2 Motivation

The task of recognition has always been easy for human beings where for the machine it is one of the sophisticated things to do. To us, vision appears to be simple, yet actually, we are processing around 60 images in every second with millions of pixels in each image. The truth is, a big part of our brain is busy doing this processing which makes it clear that this processing is a very hard thing to do for our brain. Moreover, teaching a machine to see like we do is an extremely difficult errand, not just because it is difficult to make a machine understand all the technical stuff, but since we do not know actually how it happens as the entire process is done by the central nervous system. When we see an object reflection of light from that object enters our retina and after doing some elementary analysis it is passed to the brain where the visual cortex analyses the image in a more detailed manner. This process happens in a fraction of a tiny second almost subconsciously. Though all the difficulties we human being have managed to developed a lot in the field of computer vision and we have been able to teach machines the way to seeing a thing. Nowadays machines can classify objects near to human level which can solve many problems we face in our day to day life. With the help of computer vision, it is possible to do image search via search engines like google, we can now do facial recognition and can recognize humans which are vastly used on facebook, through gesture recognition we can detect robbery and so on. Moreover, we can now make autonomous cars, intelligent robots and we can also do sophisticated operations with the help of machines so accurately which was previously impossible for humans to do. The more we can increase the accuracy of object recognition in the field of computer vision the more we will be able to accurately solve the above-mentioned problems and so on and it will be a huge improvement in the field of artificial intelligence and human beings. Our thesis is dedicated to increasing the accuracy in segmentation task where machines reach above human level as done in classification within real time. Moreover, as classification task is saturated in its entirety, the only valid option for deep learning enthusiasts to pursue segmentation and object detection with supervised learning. So, we chose segmentation with supervised learning as our primary task.

1.3 Objective

In computer vision, image segmentation is the way of segmenting a digital image into multiple sets of pixels called super pixel. The main objective of image segmentation is to simplify the image into something which is meaningful and easier to analyze. Objects and boundaries such as line, curve etc are located by image segmentation. A label is assigned to each pixel so that the pixels having the same label share certain characteristics. The result of semantic segmentation is set of segments which cover the image entirely. With respect to same characteristics or computed property such as color, intensity or texture each of the pixel in a region are similar. The region which is adjacent is different with respect to same characteristics. Our research is based on pixel level image segmentation. The convolutional neural network has become popular for recognition tasks. We created a model where we took the VGG-19 neural network which is a popular classification convolutional neural network. We turned it into a fully convolutional neural network with more fine features for extracting deep features and we used skip architecture to get a better output. Our training was performed in pascal VOC2012 dataset with SBD training data and validation set was used from reduced pascal VOC validation dataset.

Chapter 2

Literature Review

2.1 Early Works

Convolutional networks can learn from an extensive amount of image and video [11, 20] data. Large public image repositories like ImageNet, SBD dataset with increasing amount computing power, especially GPUs are making this learning process high-yielding [20]. With the help of convolutional network above many other methods, it is possible to reach human-like accuracy in visual ability [11]. Zeiler et al. [24] enhanced the design of Krizhevsky et al. [11] and use relatively smaller receptive window size and stride for the starting convolutional layer [20].

2.2 Fully Convolutional Neural Network

With the advancement of transfer learning [3], it is very convenient for us to use pre-trained ConvNet without using datasets of adequate size while saving a huge amount of time. Starting from a couple of visual recognition tasks [15, 3, 24] the recent advances in this field lets us create such nets and fine-tune them so that they can dense prediction of semantic segmentation [15].

Fully convolutional networks can be redesigned specially to learn for image data discarding irrelevant parameters and making the network more productive. In fully convolutional networks thoughts of developing the convnets to take variable sized inputs was probably was first seen [15] in Matan et al. [16] which used the LeNet [12] to recognize strings of digits but it could only handle one-dimensional input string [16, 15]. Later another revolutionary attempt by Wolf and Platt modified the convolutional network outputs to two-dimensional maps of detection scored for the four corners of postal address blocks [22].

Ronneberger et al. adjusted the design of the fully convolutional network so it could work with very few training images yet yielding more exact segmentation [18]. They made it happen by shrinking the network by successive layers while substituting the polling operator by upsampling operators.

Chapter 3

Architecture

3.1 Classification network as Segmentation Architecture

We built our network especially based on the VGG architecture which performed very well in the ILSVRC14 [15]. This network was the first to utilize significantly smaller 3×3 filters in each convolutional layers and furthermore joined them as a sequence of convolutions. In any case, the immense preferred standpoint of VGG is the understanding that various 3×3 convolutions in succession can imitate the impact of larger receptive fields like 5×5 or 7×7. We followed the VGG 19-layer network where along with other changes we removed the final classifier network and turned all the fully connected layers to convolutions also done by Evan Shelhamer et al. [15].

Recognition networks like LeNet [12], AlexNet [11] apparently could take the input of fixed size and could produce non-spatial outputs [15]. The fully connected layers that these have fixed sizes and they discards away the spatial coordinates [15].

3.2 Transfer Learning

If we want to tune the FCN network properly we need to give proper attention to an image to image learning setting. This setting includes setting a good batch size. We skipped normalizing the loss so that each and every pixel has the same weight paying a little attention to the dimension of the image and the batch [15]. We encounter that it is very hard to do segmentation task if we keep the batch size at a high dimension and for that, we had to decrease the learning rate according to the batch size. Keeping batch size minimal was not the only thing we did for optimization, we also used a higher momentum which added an extra weight on recent gradients as mentioned in Evan Shelhamer et al. [15].

3.3 Classifier and Feature Map

In our training process, we will be calculating per-pixel softmax loss and will be validating mean pixel intersection over union with the background and mean of all classes ignoring pixels that are masked out in the ground truth [15, 4].

The softmax function with loss [2] is a used which crushes a N-dimensional vector x of random real values to an N-dimensional vector (x) of real values in the range from 0 to 1 that will sum up to 1. The function is as follows:

$$\sigma(x)_j = \frac{e^{z_j}}{\sum_{k=1}^{K} e^{z_k}} \text{ here j=1,2,...,k}$$

For our model which is made for testing on VOC2012 data [4], the output feature map will be for 21 classes (including background). In the 3D output feature map the pixels belonging to the predicted class will be 1 and for that same pixel, other classes will contain 0. As the output feature map is the same size as the input feature map the 0.

3.4 Architecture for Feature Extraction

Here the image is fed sequentially into (Conv1_1, Conv2_2) to (Conv2_1, Conv2_2). In CONV1 and CONV2 we have 3x3 kernels so CONV1 have 2 convolutions. Therefore the first one has receptive field 3 and the second one has receptive field 5. Then passed to (Conv3_1, Conv3_2, Conv3_3) to (Conv_1, Conv4_2, Conv4_3, Conv4_4) to (Conv5_1, Conv5_2, Conv5_3, Conv5_4). For these three we have 4 Conv so we have 3x3,5x5,7x7,9x9 receptive field. Then finally we pass it through FC6 and FC7. The last layer we have is the score layer which is first upsampled to make it equal to the size of Conv4 and then concatenated. The concatenated result is upsampled to make it equal to the size of Conv3 and it is again concatenated. Finally, the last score layer map is upsampled and it is made equal to the size of the input image.

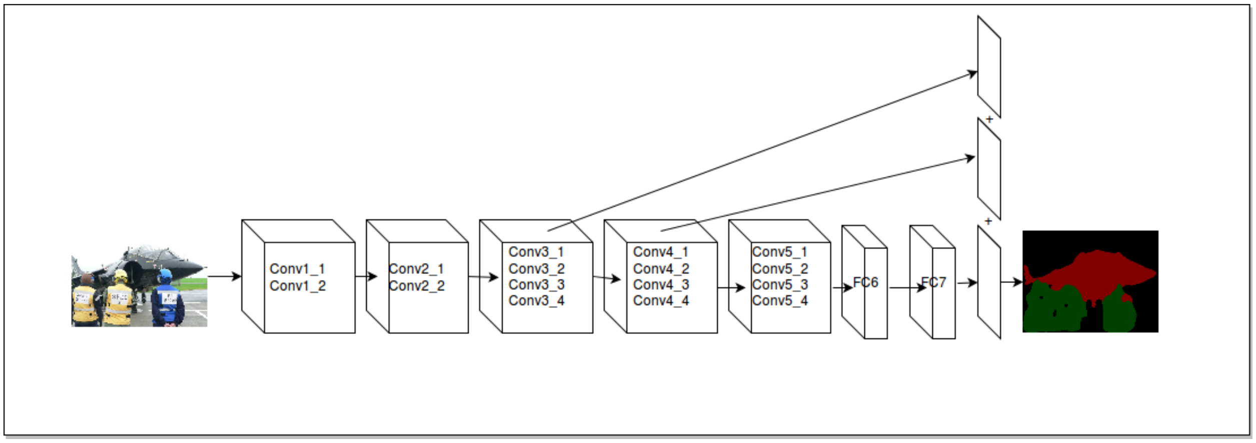

Figure 3.1: Our Deep Convnet Architecture

In our network, we are using a padding of hundred at the beginning while we are taking an image as an input as mentioned by Evan Shelhamer et al. [15] keeping hundred pixels as an input padding guarantees balanced alignment to the output to the input for any size of input from the given datasets.

Chapter 4

Experiments

4.1 Experimental Process

We used transfer learning to load weights from vgg-19 and then fine-tuned with additional data. Backpropagation [12] was used to fine tune all the layers end-to-end. As we follow the method described in [15] and adopt the 3 layer skip architecture in fcn-8s-all-at-once, the training time was slashed less than half by the scale layers usage. It took us 10 hours to train the whole network to get the best mIOU using a single GPU solution. For the first stage of training, we used Pascal VOC 2012 training images which sum up to 1464 images. After validating on the reduced VOC2012 validation set of 346 images as described in [25] we get 58.5 mIOU.

Additional data were used to enhance the accuracy and mIOU of the model for which we trained on SBD datasets [6] which consist of 8498 training images and 2857 validation images. We select all the images for training which sums up to 11355 images. If we take out the common images in the validation set of Pascal VOC 2012, we find the reduced set of 346 images of the total 1449 images. The optimum mIOU we find is 66.2 percent.

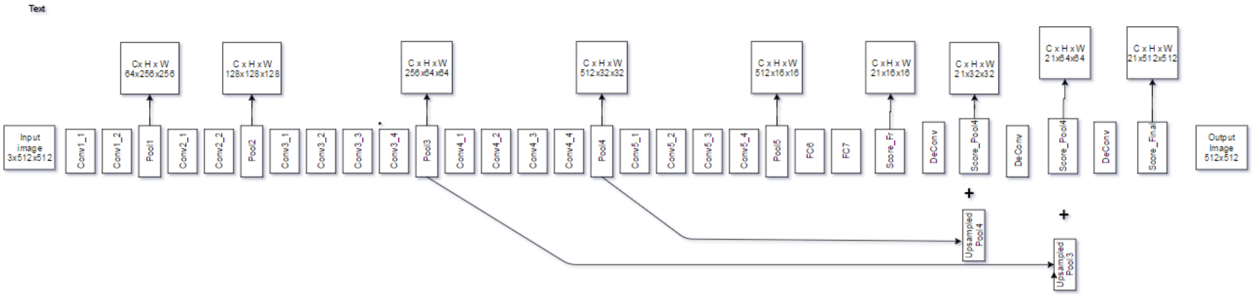

Figure 4.1: Tensors Changes with FCN.

4.2 Experimental Results

The experimental result is for validation data of 346 images. We have tried to visualize the loss against different metrics across a range of tests. As our training was done in stages, the loss map would go down and then spike again after the start of the 2nd stage. This is because we use data of [4] and then in the 2nd stage we use data of [6]. The size of data increases drastically 10 times and so does the loss. But due to VOC and SBD data are quite similar the prediction becomes much accurate for the similarity.

If we plot a graph for a number of iteration vs. mean accuracy then we can see that with every iteration the mean accuracy increase rapidly for the first 100,000 iterations. And afterward, it increases steadily and quite slowly. The final mean accuracy reaches 78.6 percent after 400,000 iterations.

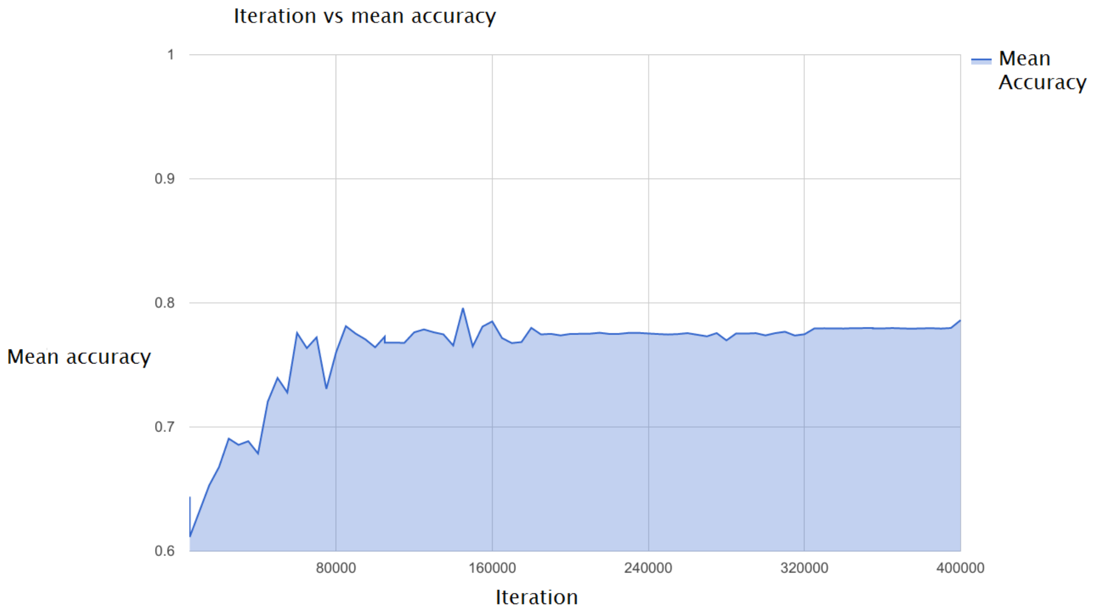

Figure 4.2: A graph to show the increase in mean accuracy per iterations

If we plot a graph of iteration vs loss then plot seems to have many upward and downward slopes and it gradually drops down after 320,000 iterations. We also see that after 60,000 iterations the loss seems to go down quite low, but our 2nd stage starts from 100,000 iterations and loss spikes pretty high afterward.

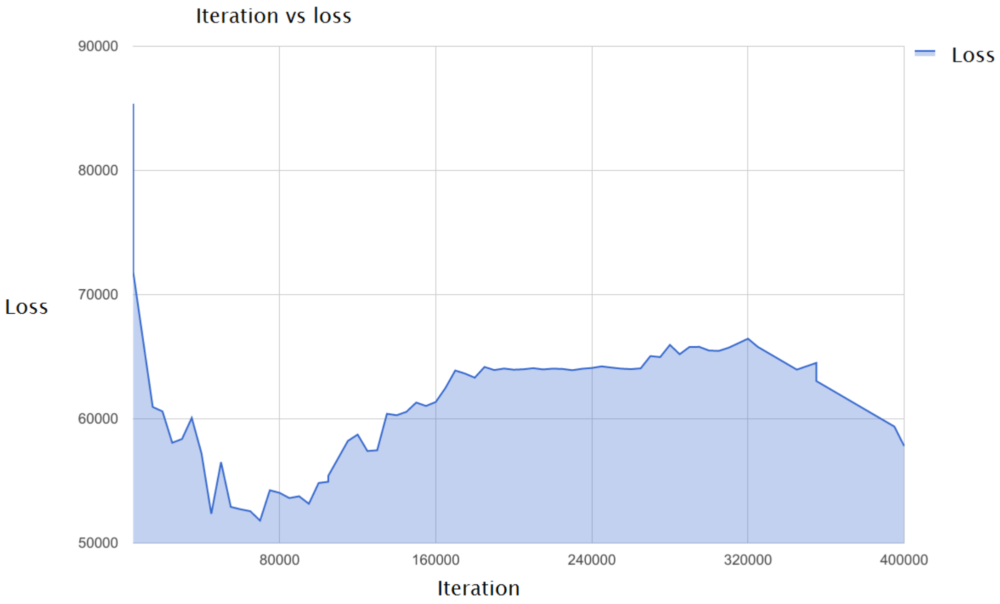

Figure 4.3: A graph showing the decrease in loss after each iteration.

Notice after 100,000 iterations the loss increases significantly and then goes down. This is happening because of the two stage training, where we first train on VOC12 training data then after 100,000 iterations we train on VOC augmented data.

As for PASCAL VOC 2012 Segmentation Challenge [4], our main objective was to find the best mean intersection over union. And the following graph portrays how it was achieved over 400,000 iterations scoring 66.2 mean IOU for VOC12 validation dataset. Though mean IOU is preferred over pixel accuracy, due to most pixel containing background pixels (only 0), sometimes mean IOU doesn't reflect the obtained result in test cases. For more finer features local context seems to be important over global contexts.

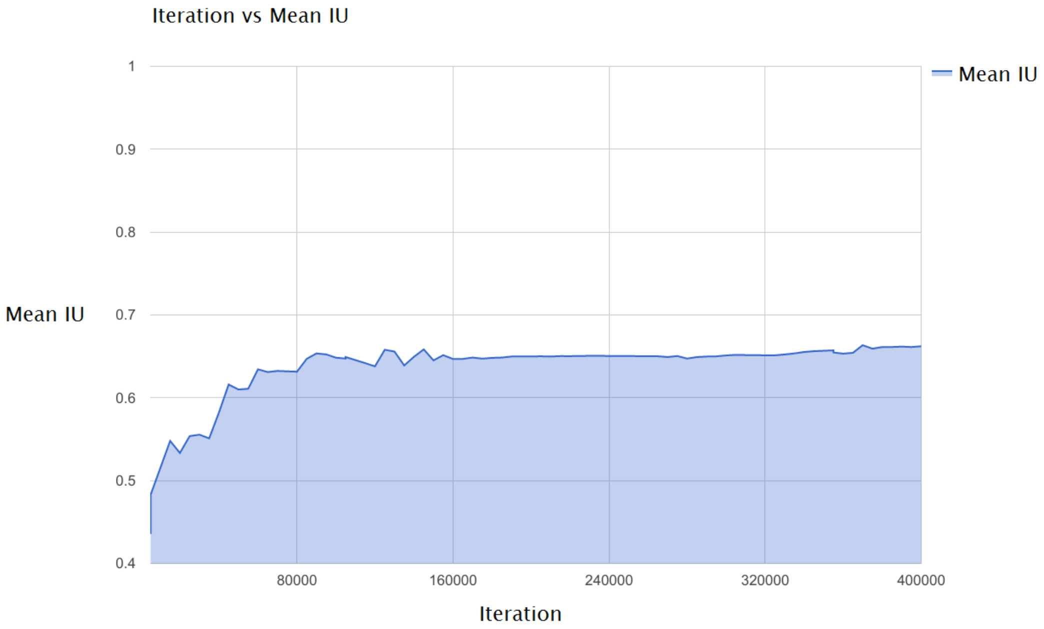

Figure 4.4: A graph showing the increase in mean intersection over union after each iteration.

Overall accuracy is used for predicting each pixel and the corresponding class. For our validation test, the overall accuracy reaches nearly 91.5 percent. But as mentioned before the overall accuracy is not a preferred method for semantic segmentation task but tried to portray over each iteration.

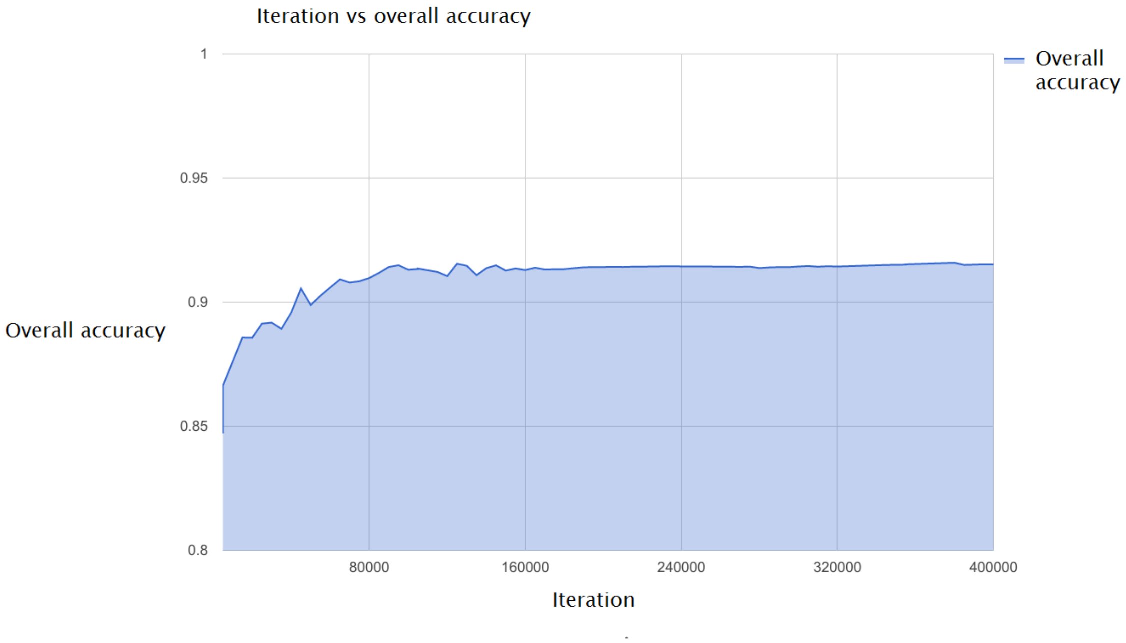

Figure 4.5: A graph showing the increase in pixel accuracy after each iteration.

Frequency weighted accuracy is another metric we show our iterations against. This metric is much reliable instead of pixel accuracy to portray how much the pixels belonging to the corresponding class more precisely. Out of 4 metrics that has been mentioned, the mean IOU and frequency weighted accuracy seems to be more self-explanatory and precise for measuring the experiments with the validation sets that we have carried out.

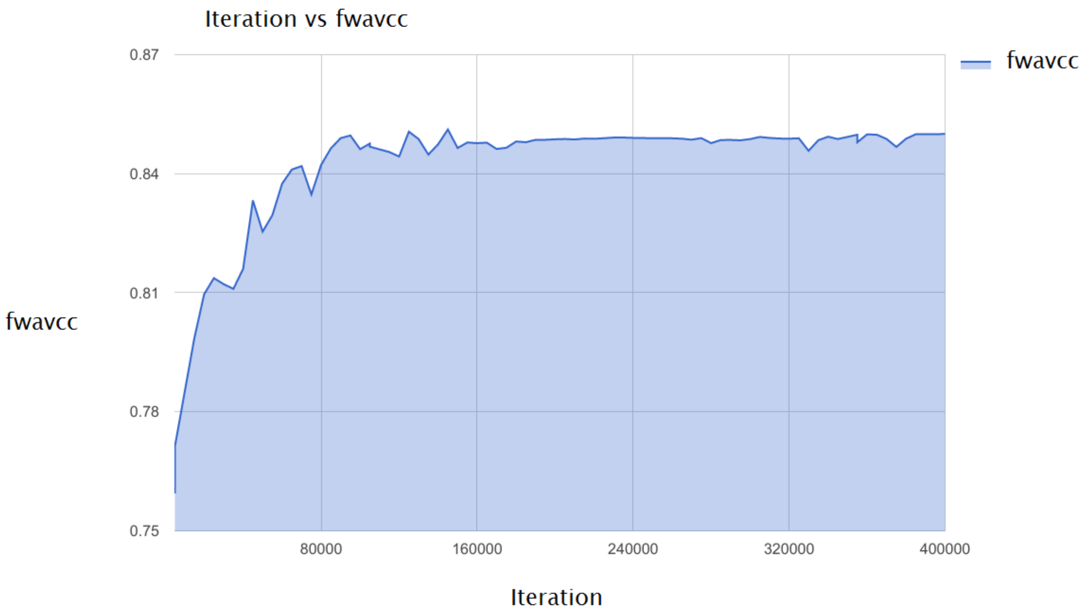

Figure 4.6: Showing the frequency weighted accuracy increasing after each iteration.

Table 4.1: Below is the chart of results we found after validating on the reduced set of VOC2012 data.

Models	MeanIOU Percentage (Trained on augmented VOC2012 images and Validated on reduced VOC2012)
FCN-8s	63.9
FCN-8s-all-at-once	63.8
FCN-8s and CRFDisconnected [14]	63.7
Our model	66.2

Chapter 5

Results

5.1 Metrics and Evaluation

The metrics used for our model to determine the scores were done through four metrics given below. Pixel Accuracy is not preferred in segmentation tasks, as background counts as the majority pixels. Instead, mean intersection over union is preferred for this scene parsing and semantic segmentation tasks.

Pixel Accuracy:

$$\sum_i P_{ii} / \sum_i \sum_j P_{ij}$$

Mean Accuracy:

$$(1/P_{class}) \sum_i P_{ii} / \sum_j P_{ij}$$

Mean IU:

$$(1/P_{class})\sum_i P_{ii}/(\sum_j P_{ij} + \sum_j P_{ji} - P_{ii})$$

Frequency Weighted IU:

$$(\sum_k \sum_j P_{kj})^{-1} \sum_i \sum_j P_{ij} P_{ii} (\sum_j P_{ij} + \sum_j P_{ji} - P_{ii})$$

P_{ij} = the number of pixels of class i predicted to belong to class j

$$P_{class} = \text{different classes}$$

$$\sum_j P_{ij} = \text{total number of pixels of class i.}$$

For our model we didn't use any post processing or augmentation of the data before or after the training. All the images and supporting labels as it was provided by [4] and [6].

5.2 Validation Results

The test results show our model's scores against other similar models. FCN-8-at-once and FCN-8s were proposed in [15], which uses vgg-16 classification net [4] and uses upsampled layers to get the feature map. The difference is quite visible as we tried to take weights from VGG-19 neural net which has more convolutional layers and use similar data from [6] and [4] and we used the skip architectures similar to [15] for getting finer features from bottom pool layer, pool4, and pool3.

Table 5.1: Below is our Pixel Accuracy, Mean Accuracy, MeanIOU and FW Accuracy compared with other models. As seen, most of the scores generate better outcome than the previous models for each metric.

Models	Pixel Accuracy	Mean Accuracy	Mean Intersection Over Union	Frequency Weighted Accuracy
FCN-8s-at-once	90.8	77.4	63.8	84
FCN-8s	90.9	76.6	63.9	84
Our model	91.5	78.6	66.2	85

The validation scores for different metrics shows a slope against different test and validation cases, (i.e data) and the loss goes downward with iterations. So all the metrics can be defined by proportional to iteration numbers and the loss can be defined by inverse proportional to iteration numbers.

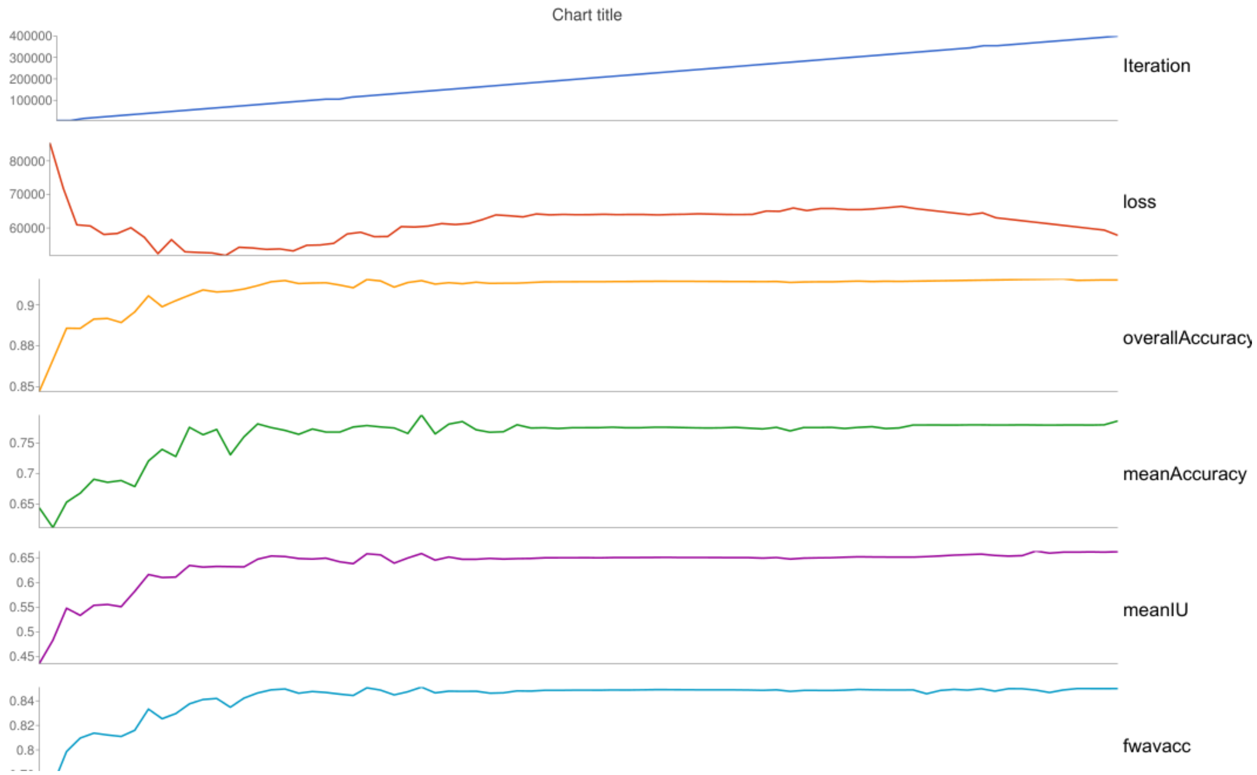

Figure 5.1: Comparative metrics and loss graph against iterations of the total validation process.

5.3 Hyper-parameter Tuning

For hyperparameter tuning, we choose a learning rate of 10e-10and a weight decay of 5e-4. With a momentum as high as 0.99 the training starts with a good chance of high oscillation and then it becomes stable. For our first stage, the training was done for 1464 images and the step size is 100,000 iteration. But we find a good mIOU within 80,000 iterations. We can check the mIOU for every 5,000 iterations and compare it with the next one to see if the learning rate is appropriate or not. As weights were transferred along with hyperparameters from VGG19 [20], the weight initializers are not needed, but for any custom convolutional layers, Gaussian or Xavier Initializers can be used to initialize weights. And for deconvolutional layers, we use bilinear interpolation as described in [15].

Table 5.2: Comparative metrics and loss graph against iterations of the total validation process.

Hyperparameters	Values
Step size (iterations)	100,000
Learning Rate	10e-10
Test size	346
Weight decay	0.0005
Momentum	0.99
Weight initializers	Gaussian/Xavier (Convolutional layers), Bilinear interpolation (Deconvolutional layers)
Training dataset	1464 images (VOC2012 training data)

For the second stage of training, we choose a learning rate of 10e-13and a step size of 300,000 while the weight decay stays the same. No weight initializers are used and we use 11,355 images from [6] which consists of both validation and training data. Moreover, the momentum remains the same as 0.99. We can also check the mIOU spikes with the help of snapshot of every 5,000 iterations.

Table 5.3: Hyperparameter tuning for 2nd stage of training

Hyperparameters	Values
Step size (iterations)	300,000
Learning Rate	10e-13
Test size	346
Weight decay	0.0005
Momentum	0.99
Weight initializers	Not needed
Training dataset	11355 images (Union of SBD training and validation data)

5.4 Test Results

After testing on Pascal VOC 2012 in the evaluation server we scored 68.1 percent mIOU, scoring better than many other models. FCN8s and FCN8s heavy both used VGG-16 [20] as for primary weights whereas Deeplab used modified VGG-16 [2] whereas CRF_RNN used VGG-16 same as FCN8s. All the models used similar data for training the net. Below is a list of test results of MeanIOU for all these models compared with ours.

Table 5.4: A chart showing different model's test result in VOC2012 segmentation challenge.

Models	MeanIOU (VOC2012 test results trained on only VOC2012 training or VOC2012 augmentated data)
FCN-8s-heavy	67.2
FCN-8s	62.2
CRF_RNN	65.2
DeepLab-CRF	66.4
DeepLab-CRF-MSc	67.1
VGG19_FCN (our)	68.1

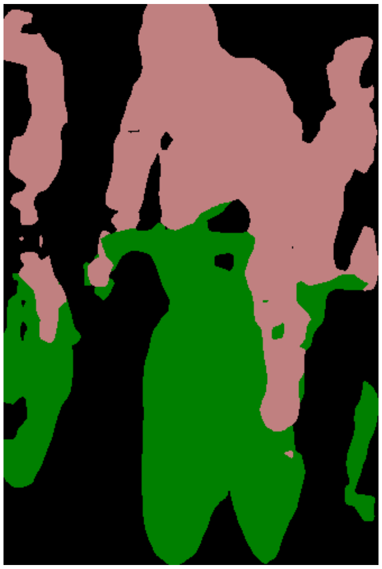

Figure 5.2: FCN-8 Heavy

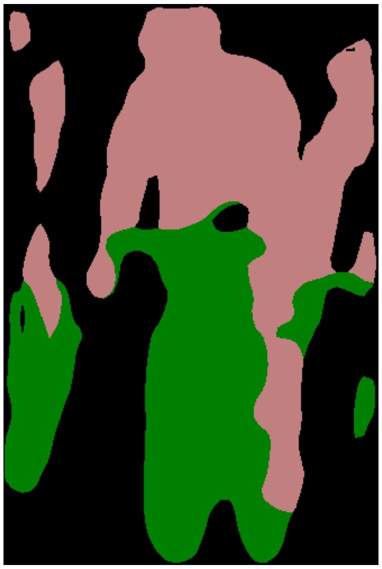

Figure 5.3: FCN-8 At Once

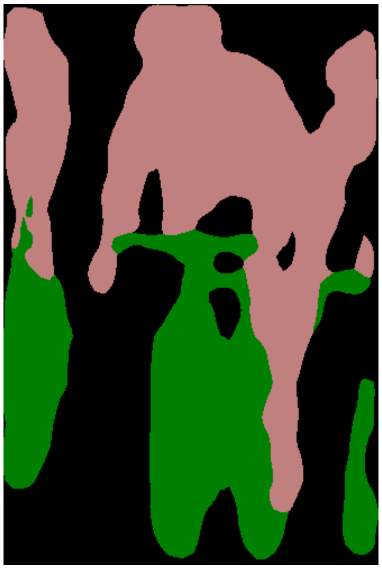

Figure 5.4: VGG19 FCN

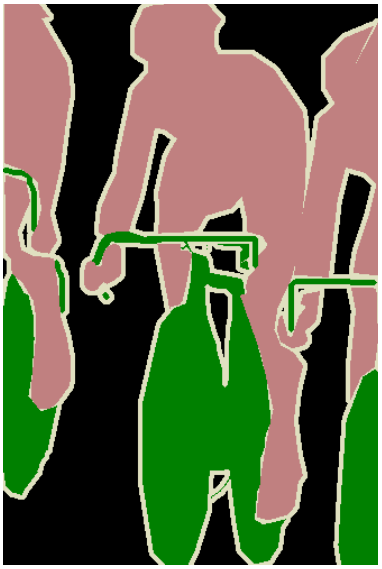

Figure 5.5: Ground Truth

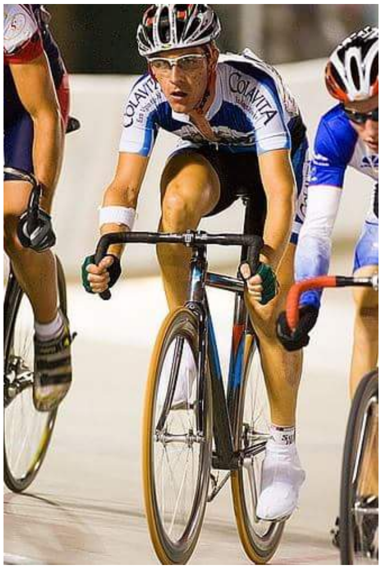

Figure 5.6: Original Image

Chapter 6

Discussion

6.1 Conclusion

Convolutional neural networks is a cornerstone for any recognition task nowadays ranging from classification, object-detection to segmentation. Moreover, Fully convolutional neural networks are better feature-extractors than their fully connected layer consisting counterparts. Furthermore, the deeper the architecture the better the feature extraction process although not always. Our novel idea was to show that with a deeper model than a traditional FCN, with similar training data and hyperparameter the results obtained can beat other base models with similar settings.

6.2 Limitations

The limitation that was faced for training was due to having smaller GPU memory. As for our training and simultaneous testing, the memory required was nearly 5 GB of GPU memory. For using any other proponent or using deeper net the memory needed is much more. The slack can be cut off if only training is conducted and testing is conducted separately. If training was done with CPU the time required would be 10 fold. For our training, we needed 72 hours of continuous training but if conducted with CPU it would increase more. For using data of different origin the labeling need to be done image by image basis. New scripts needed to be written to label the images according to VOC2012 labeling. Training for bigger class datasets like Siftflow or Pascal Context dataset was not possible due to memory issue. As it requires quite a large GPU memory, which we didn't possess. The snapshot of models was nearly 500mb, so HDD memory is another big issue if every 5000 iteration snapshot is being saved for 400,000 iterations then it stacks up to 40GB. So we have to carry out simultaneous testing with training while saving the info in the log file.

6.3 Future Works

As for future works, we plan to use much more larger dataset from Microsoft COCO challenge [13] which consists of nearly 60,000 + images. Though the labeling is quite sparse and less fine than pascal dataset we hope to work with this in our future for getting better mean IOU and segmentation.

We have also experimented with dilated convolution [23] in our model, which seems to save up to 20 percent of memory usage but the segmentation result was poorer than our model without dilation. We plan to use dilated model in a way which would help us to get better results.

Conditional random fields can be used as recurrent neural nets as proposed in [25], which we plan to incorporate in our model to get a more finer feature. As tested in various cases it has increased the mean IOU significantly. We wanted to use it in our current model but due to memory shortage, it was not possible for to incorporate.

References

[1] Anurag Arnab et al. “Higher order conditional random fields in deep neural networks”. In: *European Conference on Computer Vision*. Springer. 2016, pp. 524–540.

[2] Liang-Chieh Chen et al. “Semantic image segmentation with deep convolutional nets and fully connected crfs”. In: *arXiv preprint arXiv:1412.7062* (2014).

[3] Jeff Donahue et al. “DeCAF: A Deep Convolutional Activation Feature for Generic Visual Recognition.” In: *Icml*. Vol. 32. 2014, pp. 647–655.

[4] Mark Everingham et al. “The pascal visual object classes (voc) challenge”. In: *International journal of computer vision* 88.2 (2010), pp. 303–338.

[5] Clement Farabet et al. “Learning hierarchical features for scene labeling”. In: *IEEE transactions on pattern analysis and machine intelligence* 35.8 (2013), pp. 1915–1929.

[6] Bharath Hariharan et al. “Simultaneous detection and segmentation”. In: *European Conference on Computer Vision*. Springer. 2014, pp. 297–312.

[7] Xuming He, Richard S Zemel, and Miguel Á Carreira-Perpiñán. “Multiscale conditional random fields for image labeling”. In: *Computer vision and pattern recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE computer society conference on*. Vol. 2. IEEE. 2004, pp. II–II.

[8] Yangqing Jia et al. “Caffe: Convolutional architecture for fast feature embedding”. In: *Proceedings of the 22nd ACM international conference on Multimedia*. ACM. 2014, pp. 675–678.

[9] Pushmeet Kohli, Philip HS Torr, et al. “Robust higher order potentials for enforcing label consistency”. In: *International Journal of Computer Vision* 82.3 (2009), pp. 302–324.

[10] Vladlen Koltun. “Efficient inference in fully connected crfs with gaussian edge potentials”. In: *Adv. Neural Inf. Process. Syst* 2.3 (2011), p. 4.

[11] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. “Imagenet classification with deep convolutional neural networks”. In: *Advances in neural information processing systems*. 2012, pp. 1097–1105.

[12] Yann LeCun et al. “Backpropagation applied to handwritten zip code recognition”. In: *Neural computation* 1.4 (1989), pp. 541–551.

[13] Tsung-Yi Lin et al. “Microsoft coco: Common objects in context”. In: *European Conference on Computer Vision*. Springer. 2014, pp. 740–755.

[14] Wei Liu, Andrew Rabinovich, and Alexander C Berg. “Parsenet: Looking wider to see better”. In: *arXiv preprint arXiv:1506.04579* (2015).

[15] Jonathan Long, Evan Shelhamer, and Trevor Darrell. “Fully convolutional networks for semantic segmentation”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2015, pp. 3431–3440.

[16] Ofer Matan et al. “Multi-digit recognition using a space displacement neural network”. In: *NIPS*. 1991, pp. 488–495.

[17] Feng Ning et al. “Toward automatic phenotyping of developing embryos from videos”. In: *IEEE Transactions on Image Processing* 14.9 (2005), pp. 1360–1371.

[18] Olaf Ronneberger, Philipp Fischer, and Thomas 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.

[19] Jamie Shotton et al. “Textonboost for image understanding: Multi-class object recognition and segmentation by jointly modeling texture, layout, and context”. In: *International Journal of Computer Vision* 81.1 (2009), pp. 2–23.

[20] Karen Simonyan and Andrew Zisserman. “Very deep convolutional networks for large-scale image recognition”. In: *arXiv preprint arXiv:1409.1556* (2014).

[21] Christian Szegedy et al. “Going deeper with convolutions”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2015, pp. 1–9.

[22] Ralph Wolf and John C Platt. “Postal address block location using a convolutional locator network”. In: *Advances in Neural Information Processing Systems* (1994), pp. 745–745.

[23] Fisher Yu and Vladlen Koltun. “Multi-scale context aggregation by dilated convolutions”. In: *arXiv preprint arXiv:1511.07122* (2015).

[24] Matthew D Zeiler and Rob Fergus. “Visualizing and understanding convolutional networks”. In: *European conference on computer vision*. Springer. 2014, pp. 818–833.

[25] Shuai Zheng et al. “Conditional random fields as recurrent neural networks”. In: *Proceedings of the IEEE International Conference on Computer Vision*. 2015, pp. 1529–1537.