Nuclei Detection using Convolutional Neural Networks - Spot Nuclei, Speed Cures

Goutham Srivatsav Arra
Indiana University
Bloomington, IN 47408
garra@iu.edu

Priyadarshini Vijjigiri
Indiana University
Bloomington, IN 47408
pvijjigi@iu.edu

Pavan Kumar Madineni
Indiana University
Bloomington, IN 47408
pmadinen@iu.edu

ABSTRACT

We put forth a conceptually straightforward, flexible and a robust framework deep learning model for Object instance segmentation. Our approach efficiently identifies various nuclei in an image while simultaneously generating a high quality segmentation map that contains masks for all the nuclei, after a thorough pixel-level examination of the image. Dense image segmentation essentially involves dividing images into meaningful reagions which can be viewed as pixel level classification task and any such reliable image segmentation employs convolutional neural networks. We have implemented two of the complex CNN architectures, U-Net and Linknet, which are specefically designed CNNs suitable for the task at hand. We have pioneered a new model which is an aggregation of the results from the earlier built U-Net and Linknet models. We hope our simple and effective approach will serve as a solid baseline and help ease future research in instance level segmentation problems.

KEYWORDS

Deep Learning, Computer Vision, Image Segmentation, Medical Image Processing, U-Net, Linknet, Ensemble models, Pixel level classification

1 INTRODUCTION

Drug discovery is becoming increasingly difficult and producing new drugs is the need of the hour to develop faster and accurate cure for diseases ranging from cancer to metabolic malfunctions to psychiatric disorders. The number of new drugs produced every year after billions of dollars spent on research and development is depressingly decreasing. In 1950‘s, 30 new drugs could be discovered for a cost at which not a single new drug can be discovered today and this fact is corroborated by the Eroom‘s law. The process of inventing new and effective drugs involves testing thousands of chemicals on human cells. Analysis and interpretation of these strained section of cells is one of the major steps in the prognosis which is usually carried out manually by pathologists. The major bottleneck in expediting this process is the quest for a software that can analyze these complex image data and automatically segment nuclei from these microscopic images.

Since this process involves learning features from complex images comprising human cells which are homogeneous in nature and nucleus being the most distinguishable part of the cell, it is apposite to employ a neural network to learn minute attributes of nuclei which are otherwise inconspicuous to manual observation. We are looking to build a robust nuclei detector by juxtaposing the performances of different variants of convolutional neural network architectures.

1.1 Instance Segmentation

Our aim for this project is two fold. We need to identify all the nuclei in a given image while also precisely segmenting each nuclei by predicting a segmentation mask on the nuclei. The networks that we had built, seamelessly integrates these two tasks. This type of computer vision task is called an ***Instance Segmentation*** problem. Instance Segmentation is a combination of object detection where the goal is to classify individual objects and localize each object using bounding box and semantic segmentation, where the goal is to classify each pixel into a fixed set of categories without differentiating object instances [3].

1.2 Why Convolutional Network?

A Convolutional neural network is the coalition of convolution layers, pooling layers, fully connected layers and normalization layers that in confluence is generally used to predict the class of an image.

Before deep learning was introduced to computer vision, people used techniques like TextonForest and Random Forest based classifiers for semantic segmentation. One of the intial approaches using simple fully connected neural netwoks was patch classification, where each pixel was seperately classified into classes using a patch of image around it. Main reason to use patches was fully connected networks require fixed sized input images. Using fully connected neural networks for computer vision tasks, is generally not effective because once you flatten an image to a vector, you loose all the crucial spacial alingnment information of a 2d image. On contrary, a CNN does a great job in preserving the context of an image, aggregating all the important features [5].

But one of the main problems with using CNNs for semantic segmentation tasks is with pooling layers [9]. Pooling layers increase the field of view by aggregating the context and important features while discarding the ‘where‘ information. But for our nuclei detection problem, we requre the model to predict both the context(object detection) as well as the location of the nuclei(semantic segmentation) which is achieved by examining an image at pixel level. This problem can be overcome by designing the convolutional networks in a clever manner so we dont lose the location information of the objects in an image. Encoder-Decoder type of architechtures of CNN such as U Net and Linknet, are specefically developed in recent years, for semantic segmentation problems, in which pixel alignment information is reconstructed by decoder path (more on this is discussed later) [1].

For our work, we implement a normal Fully Convolutional Network (FCN), then we experiment with complex arrangements of CNNs - U net and Link Net.

2 DATASET DESCRIPTION

This is also kaggle project, so there is certainity with regards to the viability of training data for our work. We have 670 training images and 65 test images. For each training image, there are many ground truth mask images for all the nuclei present in the image. Although, it may appear that the training set is less in number, when compared to usual machine learning problems, we have many nuclei per image and hence it is equivalent to having adequate training imformation to automate the nucleus detection function.

Some of the training images have 4 channels - red, green, blue and alpha channel, all of them with varying height and widths. The ground truth mask images are all of the same shape(256,256,3). The alpha channel is really a mask, specifying how the pixel's colours should be merged with another pixel when the two are overlaid on top of the other. So, while loading the data, I have removed the alpha channel for all the images.

Luckily, we also have with us, growth truth mask images for all the 65 test images, which we had used to quite a good use to improve our models. Next, we talk about the scoring metric.

3 SCORING METRIC

3.1 IoU

For any image segmentation problem, not necessarily at pixel level, Intersection over Union (IoU), is the common metric used to assess how well we have predicted the target area of the object, in our case, exact masks of the nuclei.

$$\text{IOU} = \frac{A \cap B}{A \cup B}$$

As it is obvious from the name, IoU is the ratio of Intersection area between predicted area and ground truth area to the combined area of the two. We have written a function that will calculate this metric, apart from being used as training metric for the convolutional network to see if the training converges with the number of epochs.

The final evaluation metric is calculated as follows. For each image, for several of the predicted masks we get an IoU score, so we have several IoU scores per image. Now, we compare these scores with a range of thresholds, starting from 0.5 till 0.95 in steps of 0.05. For each threshold, we determine whether it is a hit or miss by giving a class 1 or 0. With this for each image, we get values for True Positive, False Negative, False Positive values. We can easily calculate Precision value for each threshold. Now we calculate average precision value per image. Next we calculate average Precision score for all the images in the test set or validation set. In short, we look to improve average Precision value for test dataset.

3.2 Modified Binary Cross Entropy

For the loss function, we had modified the usual Binary Cross Entropy funciton, by subracting it with logarithm of dice coefficient, as we have observed, this is giving us better results for linknet.

The dice score is generally used to quantify the performance of image segmentation problems. The main essence of dice score is to measure the resemblance between two objects. In short, it is the ratio of size of the overlap of two segmentations to the total size of the two objects [4].

$$\text{Dice Score} = |A \cap B||A| + |\frac{B}{A}|$$

4 FULLY CONVOLUTIONAL NETWORKS (FCN)

The traditional segmentation splits an image into visually similar patches whereas semantic segmentation divides the image into semantically more meaningful patches through pixel wise classification. The preliminary idea of a Fully Convolutional Network (FCN) is to capture the global context of the scene in an image and also furnish a vague idea of the whereabouts of the things. The FCNs work on the underlying principle of interpreting the network as a single convolution filter for each output neuron over the complete image area on which the original network was trained. An FCN is a usual CNN, where the last fully connected layer is replaced by another convolution layer with a large receptive field. When the last fully connected layer is transformed to a convolution layer, some form of localization is obtained over the regions which have more activations and if this last convolution layer is big enough, then the localization effect can be scaled up to the size of the input image.

We have discussed in the beginning, why using only pure Convolutional Network is not a good idea for segmentation problems since we need to preserve location information too. So, as expected, we did not achieve good results with FCNs. We implemented 3 convolutional layers, having 16,32 and 64 filters respectively each with filter size of (3,3) with non linear Relu activation. The final image size at the end of convolutional layers is only (16,16,1). We achieved a very low average Precision score of 0.0072. Next model, that we implemented is U Net.

5 U NET

U-Net is an encoder-decoder architecture designed specifically for biomedical image processing or cell segmentation. The intrinsic intuition behind U-Net is how extensively can the features of an image be extracted. Augmenting this intuition from identifying objects using bounding boxes to exactly locating pixels of an object led to the evolution of U-Net architecture. U-Net is basically a shallow network without any fully-connected layers. Data augmentation through elastic deformations is essential since deformation is a variation that innately exists in tissues. The encoder downsamples image features using a bank of trainable convolutional filters, followed by ReLUs and max-pooling. Contrastingly, the decoders employ upsampling to increase the resolution of the feature maps for pixel classification. Max-pooling destroys spatial information which is indispensable for precise localization. In order to

reconstruct the preserved contextual information, the max-pooling indices from the encoder are availed. The U-Net architecture introduces a weight map to compensate for the class imbalance of pixels and compel the network to learn borders between touching cells [6].

As we can see in the figure below, we have a lot of choices to build the structure of the U-Net. After a lot of experimentation, our U Net has 2 copies of 16 convolution layers at the input with (3,3) filter, with Relu non linear acitvatation, Drop Out layer and a Max Pool layer with stride 2 and kernel size (2,2) [7]. Similar network repeats in the contractive path with 32, 64, 128 filters respectively. Then the expansive path of the U Net begins with a Convolution Transpose layer of 128 filters of (3,3) size, followed by concatenation with output branch of normal convolutional layer from the contractive path. Similar network repeats in the expansive layer with 64, 32, 16 filters layer [8].

Finally at the output, we have a convolutional layer with a single filter of size (1,1). The activation function used at the output layer is sigmoid, because the output is grey scale image of 0s and 1s and we are trying to predict the class of each pixel by first doing regression to 1, and then assigning class based on a threshold of 0.5.

Our first objective while optimizing our model is to fix the architechture of the U net. We have experimented with number of filters in all the levels of expansive and contractive sides. With a 128x128 size input image, and a batch size of 32, model scored 0.214 mean precision value on the test data, when we doubled the number of filters at every level. Whereas the final network that we selected (and described above) scored 0.225 on the test data.

We found batch size to have considerable impact on the performance of our model. We experimented with several values for batch size like 8,16,32 etc. For U net, batch size of 32 outperformed others with a mean precision value of 0.225 on the test data. Also, we have implemented data augmentation of the train data by generating additional train images, which are minor modifications of the original images.

Next, we explore another encoder-decoder type of architecture called Linknet.

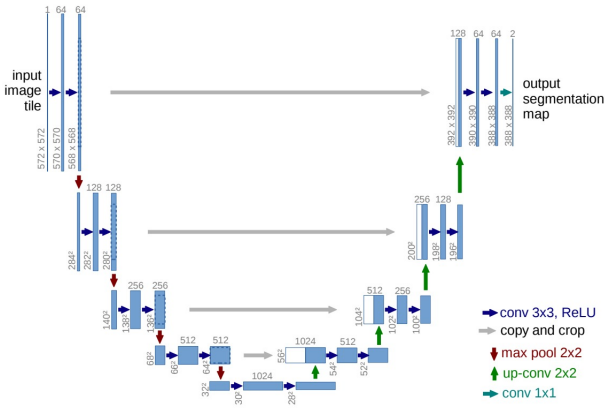

Figure 1: U Net Architecture

6 LINKNET

Pixel wise semantic segmentation of an image for better visual discernment of a scene demands not only accuracy but also efficiency in order to be viable for use in practical applications. LinkNet is a light deep neural network architecture designed to perform semantic segmentation for intricate scrutiny of images for highly complex tasks such as self-driving vehicles, augmented reality, genome analysis etc. This architecture though elementary in its basic structure is well-known for its ability to perform at rapid pace by maneuvering the enormous number of parameters innately present in a deep neural network. This network is basically an encoder-decoder type architecture where the encoder and decoder exist in pairs. This deep neural network efficiently shares the information learnt by the encoder with the decoder after each downsampling block. This technique is more effective than mere application of pooling indices or fully convolutional networks in decoder and the ramifications of using this technique are not just limited to better accuracy but a deep plummet in the number of parameters in the decoder [2].

Generally, spatial information is lost in the encoder due to pooling or strided convolution which is recovered by using the pooling indices or by full convolution. The first encoder block does not perform strided convolution and instead bypasses spatial information directly from the encoder to the corresponding decoder which results in an improved accuracy coupled with drastic decrease in processing time. Every convolution layer is followed by batch-normalization and employs ReLU activation function to induce nonlinearity into the network. In this way, contextual information which prone to being lost at each level of encoder is preserved thereby preventing the use of additional parameters and operations in retrieving this lost information.

Here is a brief description of our Linknet architecture :

First encoder block consists of two networks of Convolutional layers, but one is strided with value of 2. Both have (3,3) size kernel with 64 filters, followed by Average2D pooling. It is followed by two more convolutional layers (stride 1 and 64 filters). Second encoder block is similar to first, except it has 128 filters everywhere.

First decoder block starts with a simple convolutional layer with 16 (1,1) size filters, followed by Deconvolution layer with (3,3) filters and a stride of 2 with Dropout layer in the end. Second decoder block is similar to first, except it has 32 filters.

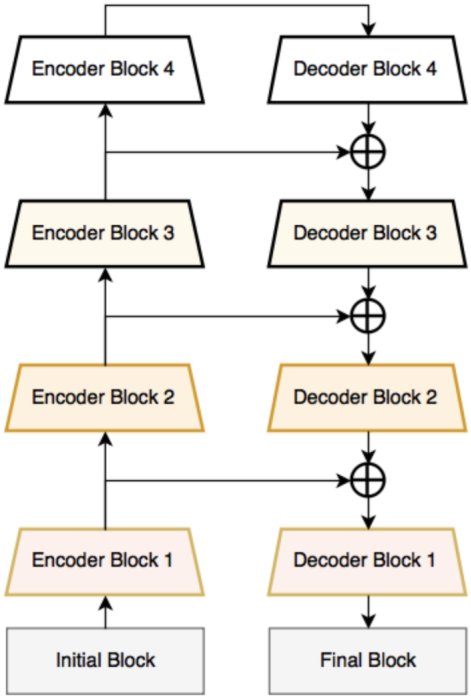

Figure 2: Link Net Architecture

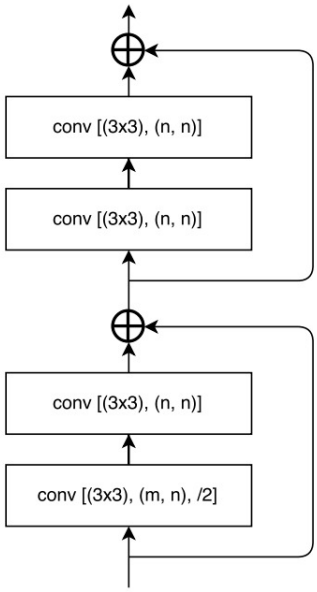

Figure 3: Encoder block

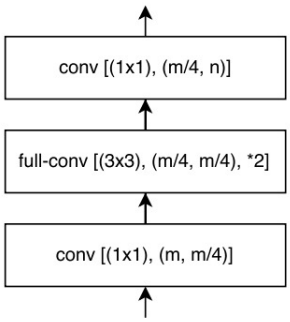

Figure 4: Decoder block

We have experimented with two designs of linknet, 2 encoder-2 decoder type and 4 encoder-4 decoder type of architecture with number of filters doubling for the next two encoder/decoders.

We have benefitted with a batch size of 8 for almost all of the runs of our model on the test data. When we have 2 encoder- 2 decoder model, with a batch size of 8, the test average precision was 0.218 When we have 4 encoder- 4 decoder model, with a batch size of 16, the test average precision was highest with a value of 0.202.

We trained our models with adam optimizer and with a learning rate of 0.001. After seeing our results, we could see that 4 encoder-4 decoder model was overfitting the data a little and has a lot of capacity than needed for our problem as it is not producing great results on the test data. So, we dropped the idea and continued our optimization with 2 encoder-2 decoder model.

Next, to further improve our score we increased the size of the input image. Earlier we downsampled input image to (128,128,3) and then upsampled the predicted test image to (256,256,1) to evaluate the model. After increasing the input size of the image to (256,256,3) with a batch size of 8 we get lower average precision value (0.19), and due to large size of input images, takes about 30 mins to run on the GPU clusters. So, we decided it is not a feasible option for us.

The most significant improvement came when we implemented data augmentation to the input data with ImageDataGenerator package from Keras Image preprocessing.

We now, further improve our model by building an ensemble model out of U-net and Linknet architectures.

7 FINAL ENSEMBLE MODEL

Ensembling models always produces better result but we have to be careful of overfitting. Thankfully, that was not the case here. Before we get into that, lets recap about the models that we have built so far. FCN model that we first built has performed very poorly and this was expected. Next, U net and Linknet both performed well on the test data with U net slightly better than the former. Our experiment on ensembling these models, worked in our favour because the ensemble model outperformed all the models built so far. We have done unweighted average of the predicition layer before thresholding and prediction of class. This allows for the final score in the pixel to improve more than 0.5.

7.1 Summary of Ensembled Model- Training and Performance

- Minibatch size : 8
- Loss Function. : Softmax Cross Entropy
- Optimizer : Adam Optimizer
- No of Epochs : 20
- U Net Performance : 0.225 average precision value
- Link Net Performance : 0.218 average precision value
- Ensemble Model Performance : 0.228 average precision value

To confirm our results, we have visualized our work by comparing original image with nuclei, ground truth mask image and predicted masks image.

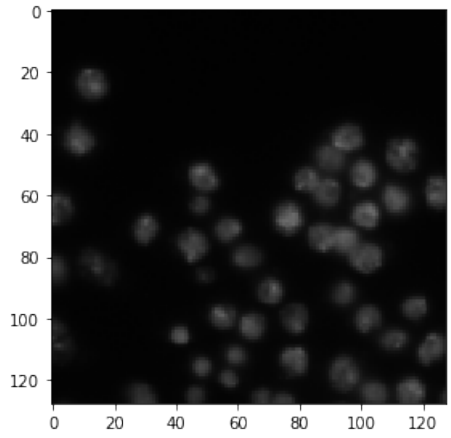

Figure: Image with Nuclei

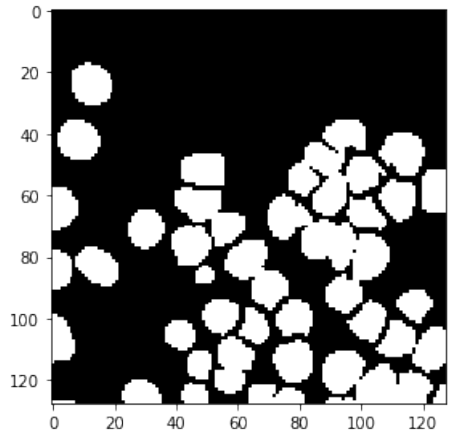

Figure: Ground truth Mask of image

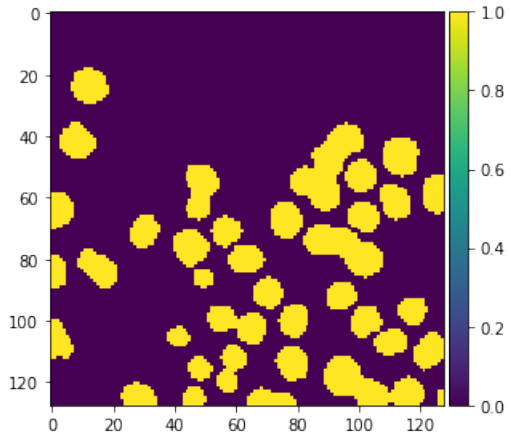

Figure: Predicted mask of Image

8 CONCLUSION

In summary, we had succesfully built an aggregate model of U-Net and Linknet CNN architechtures which achieves relatively good performance on diverse biomedical segmentation applications. In addition to the fusion architecture that we pioneered to build, the model offers enormous scope for further refinement owing to the large number of parameters in each of these networks (close to 2 lakh parameters) implying high model capacity to approximate even more complex functions.

9 WORK BREAKDOWN

Initial data preprocessing and building Fully convolution model is contributed by Priyadarshini Vijjigiri. Architecture U-Net is built and optimized by Pavan Kumar Madineni. Link-Net Architecture is built and optimized by Goutham Srivatsav Arra. All the models are analysed, optimized and ensembled together by all the team members. All the team members have equally contributed to the sections of project paper, and also have put equal team efforts to resolve the issues faced at each step.

ACKNOWLEDGMENTS

The authors would like to thank Prof. Minje Kim for his support and suggestions to write this paper. Also, course (ENGR E533) materials and lectures were extremely helpful in completing this project.

REFERENCES

[1] [n. d.]. *Multibox Single Shot Detector.* https://leonardoaraujosantos.gitbooks.io/artificial-inteligence/content/image_segmentation.html
[2] Eugenio Culurciello Abhishek Chaurasia. 2016. . https://arxiv.org/pdf/1707.03718.pdf
[3] Sasank Chilamkurthy. 2017. . http://blog.qure.ai/notes/semantic-segmentation-deep-learning-review
[4] Gumeo. 2016. . https://stats.stackexchange.com/questions/195006/is-the-dice-coefficient-the-same-as-accuracy/253992
[5] Trevor Darrell Jonathan Long, Evan Shelhamer. [n. d.]. *Fully convolutional networks.* https://people.eecs.berkeley.edu/~jonlong/long_shelhamer_fcn.pdf
[6] University of Freiburg. 2018. *Convolutional Networks for Biomedical Image Segmentation.* https://lmb.informatik.uni-freiburg.de/people/ronneber/u-net/
[7] Thomas Brox Olaf Ronneberger, Philipp Fischer. [n. d.]. *Network Architecture.* https://arxiv.org/pdf/1505.04597.pdf
[8] Thomas Brox Olaf Ronneberger, Philipp Fischer. 2018. *U-Net Specifications.* http://people.ee.duke.edu/~lcarin/Greg1.12.2018.pdf

[9] Alexey Shvets Vladimir Iglovikov. [n. d.]. *Network Architecture.* https://arxiv.org/pdf/1801.05746.pdf

A CHKTEX

```
cd ../../hid-sample; git pull
Already up to date.
cp ../../hid-sample/paper/Makefile .
cp ../../hid-sample/paper/report.tex .
WARNING: line longer than 80 characters
('    ', '89:', '\\title{Nuclei Detection using Convolutional Neural Networks - Spot Nuclei, Speed Cures}\r\n')
WARNING: line longer than 80 characters
('    ', '150:', '\\keywords{Deep Learning, Computer Vision, Image Segmentation, Medical Image Processing, U-Net, Linknet, Ensemble models, Pixel level classification}\r\n')
WARNING: line longer than 80 characters
('    ', '84:', 'important features ~\\cite{Fully-Convolutional-Networks-for-Semantic-Segmentation}.\r\n')
WARNING: line longer than 80 characters
('    ', '101:', 'reconstructed by decoder path (more on this is discussed later)~\\cite{Fully-Convolutional-network}.\r\n')
WARNING: line longer than 80 characters
('    ', '93:', 'overlap of two segmentations to the total size of the two objects ~\\cite{Dice-Coefficient}.\r\n')
WARNING: line longer than 80 characters
('    ', '84:', '\\includegraphics[scale=0.5]{images/link-net.png}\\label{fig: Link Net Architecture}\r\n')
Warning 1 in content.tex line 87: Command terminated with space.
\subsection {Instance Segmentation}
           ^

Warning 39 in content.tex line 97: Double space found.
without differentiating object instances ~\cite{Semantic-segmentation}.
                                         ^

Warning 39 in content.tex line 117: Double space found.
important features ~\cite{Fully-Convolutional-Networks-for-Semantic-Segmentation}.
                   ^

Warning 39 in content.tex line 120: Double space found.
is with pooling layers ~\cite{Semantic-segmentation2}. Pooling layers increase
                       ^

Warning 33 in content.tex line 122: Use ' to end quotation, not `.
discarding the `where` information. But for our nuclei detection problem, we
                  ^

Warning 36 in content.tex line 123: You should put a space in front of parenthesis.
requre the model to predict both the context(object detection) as well as the
                                           ^

Warning 36 in content.tex line 124: You should put a space in front of parenthesis.
location of the nuclei(semantic segmentation) which is achieved by examining
                      ^

Warning 36 in content.tex line 149: You should put a space in front of parenthesis.
images are all of the same shape(256,256,3). The alpha channel is really a
                                ^

Warning 39 in content.tex line 199: Double space found.
overlap of two segmentations to the total size of the two objects ~\cite{Dice-Coefficient}.
                                                                  ^

Warning 39 in content.tex line 250: Double space found.
learn borders between touching cells ~\cite{convolutional-neural-network}.
                                     ^

Warning 29 in content.tex line 271: $\times$ may look prettier here.
expansive and contractive sides.  With a 128x128 size input image, and a batch
                                  ^

Warning 24 in content.tex line 287: Delete this space to maintain correct pagereferences.
\label{fig:U-Net Architecture}
^
Warning 39 in content.tex line 310: Double space found.
accuracy but a deep plummet in the number of parameters in the decoder ~\cite
                                                                       ^

Warning 1 in content.tex line 310: Command terminated with space.
accuracy but a deep plummet in the number of parameters in the decoder ~\cite
                                                                           ^

Warning 26 in content.tex line 324: You ought to remove spaces in front of punctuation.
Here is a brief description of our Linknet architecture :
                                                        ^

Warning 26 in content.tex line 405: You ought to remove spaces in front of punctuation.
 \item Minibatch size               : 8
                                   ^

Warning 26 in content.tex line 406: You ought to remove spaces in front of punctuation.
 \item Loss Function.              : Softmax Cross Entropy
                                   ^

Warning 26 in content.tex line 407: You ought to remove spaces in front of punctuation.
 \item Optimizer                   : Adam Optimizer
                                   ^

Warning 26 in content.tex line 408: You ought to remove spaces in front of punctuation.
 \item No of Epochs                : 20
                                   ^

Warning 26 in content.tex line 409: You ought to remove spaces in front of punctuation.
 \item U Net Performance           : 0.225 average precision value
                                   ^

Warning 26 in content.tex line 410: You ought to remove spaces in front of punctuation.
 \item Link Net Performance        : 0.218 average precision value
                                   ^

Warning 26 in content.tex line 411: You ought to remove spaces in front of punctuation.
 \item Ensemble Model Performance  : 0.228 average precision value
                                   ^

Warning 24 in content.tex line 420: Delete this space to maintain correct pagereferences.
\label{fig:Input Image}
^
Warning 24 in content.tex line 426: Delete this space to maintain correct pagereferences.
\label{fig:Mask }
^
Warning 24 in content.tex line 433: Delete this space to maintain correct pagereferences.
\label{fig:Predicted Mask}
^
```