Semantic Segmenation
Lab Vision Systems

Manivannan, Iswariya, Ramesh, Sathiya

Hochschule Bonn Rhein Sieg
iswariya.manivannan@smail.inf.h-brs.de,
sathiya.ramesh@smail.inf.h-brs.de, Matrikelnummer: 9033254,9031938

Abstract. Semantic segmentation is a pixel-wise classification problem where a convolutional neural network (CNN) is trained to assign a class to every pixel in an input image. The output produced is a 2D segmentation map through which we can infer what objects are present in the image, their corresponding pixel locations, and object boundaries. The need to classify every pixel calls for dense annotation of every label image. Consequently, creating manual annotations for a new dataset is expensive. Nevertheless, this project creates a limited dataset consisting of 2000 images with manual annotations. We train the model suggested by [1], with a few modifications to the training methodology and record the segmentation results for the three classes(field, field lines, ball) in the dataset.

1 Introduction

In recent years, Convolutional Neural Network architecture has outperformed the state of the art in many computer vision tasks , e.g image classification [2] , object detection [3], etc. Their success was limited in the past due to the size of the available training sets and the size of the considered networks. The major breakthrough was Alexnet by Krizhevsky et al. [4] which trained a large network with 8 layers and millions of parameters on the ImageNet dataset with 1 million training images. Since then, even larger and deeper networks have been designed. A major part of this success comes from training larger and deeper networks with labeled samples in a supervised manner.

1.1 Motivation

Though the convolutional networks are used for classification tasks, where the output to an image is a single class label, many visual tasks, especially in robotics, the location of the object is also desired in the output. Semantic segmentation is a pixel-wise classification problem with a goal of assigning a class from a list of desired classes to every pixel in an image. Thus the segmented image has different meaningful regions. Unlike the task of image classification, where the location and boundaries of the desired object in an image are irrelevant, semantic segmentation (Fig 1 and Fig 2) requires the neural network to be able to spatially localize desired objects, delineate object boundaries and produce a single channel segmentation map (output image) which is of the same size as the input image.

1.2 Challenges and Difficulties

- Labeling cost: For Semantic segmentation every pixel in the ground truth image needs to be labeled. This labeling can be achieved by first performing accurate object boundary delineation and later annotating each region with the corresponding class manually. Object shape, object occlusion and cluttered image space with objects further add to this difficulty.
- Context knowledge: Different features could represent objects in general. In a local context, where only a small region inside an object is considered, different objects might have very similar features. It is difficult to the classify the pixels within the regions as in a local context the two objects are similar. However, when a global context is gathered, the three objects become distinct, and it is now possible to classify the pixels within these regions. Therefore, for the task of semantic segmentation, gathering global context is important.

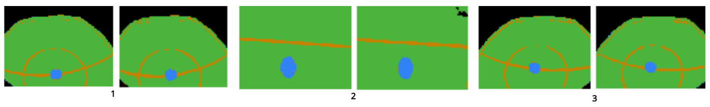

Fig. 1: Model predictions with high IoU's. The image on the left in each pair of images is the ground truth label and the right image is the network's prediction

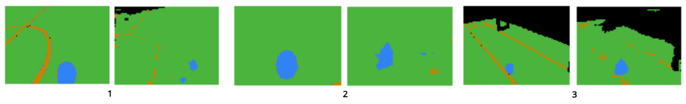

Fig. 2: Model predictions with low IoU's. The image on the left in each pair of images is the ground truth label and the right image is the network's prediction

1.3 Problem Statement

The goal of this research is to build a resource efficient framework to perform pixelwise classification on Robocup Soccer images.

- Dataset creation: The dataset should be created by collecting images and labeling them manually.
- Illumination: Illumination conditions will vary over images in the dataset which will be collected from various robocup matches. The goal is to achieve robustness to illumination variance.
- Resource efficiency: The semantic segmentation framework should to be resource efficient in terms of both memory and inference time along with considerable accuracy.

2 Related Work

The detection of the ball and other objects on the field of play of RoboCup Soccer competitions have become more and more challenging since the introduction of multi-colored balls, white goal posts, etc. Therefore, achieving reliable results with traditional algorithms based on color segmentation and edge detection within captured images becomes increasingly difficult. The relentless success of Convolutional Neural Networks (CNNs) in various tasks such as image classification or object detection motivated researchers to explore the capabilities of such networks for semantic segmentation.

In this section, Fully Convolutional Networks(FCNs) is introduced first. Section 2.2 describes the decoder variant of FCNs and the final section 2.3 discusses the decoder along with skip connection.

2.1 Fully Convolutional Networks

Long et al [5] introduced FCNs for image segmentation in 2015. FCNs was designed and trained to produce segmented output taking advantage of existing CNNs to learn hierarchies of features.FCNs were designed and trained, replacing the fully connected layers with convolutional ones which enable the generation of spatial maps. Those maps are upsampled using deconvolutions to produce dense per-pixel labeled outputs. Recent work show results on segmentation using CNN for making dense predictions, however these methods are limited,e.g. need of post processing.

2.2 Decoder Variants

Though the FCNs based architectures are more popular and successful, the original FCNs have a drawback of generating lower resolution output predictions, due to pooling layers and striding in convolutional layers. Therefore other variants were developed which modified a classification network suitable for segmentation. In general terms, all of them take a network for classification, such as VGG-16, and remove its fully connected layers. This part of the new segmentation network is called an encoder and produces a low-resolution feature maps. The decoder learning to decode those low resolution images to pixel-wise predictions for segmentation. SegNet [6] was the first with this type of architecture(fig 3). The decoder stage of SegNet is composed by a set of upsampling and convolution layers which are followed by a softmax classifier to produce pixel-wise labels for an output which has the same resolution as the input image. Max pooling indices are used to upsample the image in the decoder which improves object boundary delineation and can be adapted in any encoder-decoder architecture. The upsampled maps are then convolved with a set of trainable filter banks are to refine predictions.

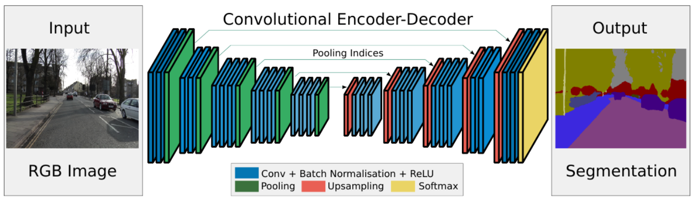

Fig. 3: SegNet architecture with an encoder and a decoder followed by a softmax classifier for pixel-wise classification. Figure extracted from [6].

2.3 Skip Connetions

Ronneberger et al [7] have designed a network called U-Net which won the ISBI cell tracking challenge in 2015. The network consists of an encoder path followed by a symmetric decoder path (fig 4). The main idea behind this architecture is to combine higher resolution feature maps from the encoder to the corresponding decoder output through skip connections. This in turn will assist the model in better localization, since the decoder might not always be able to reconstruct all the information that was lost during downsampling. The network is designed so that it is able to obtain accurate segmentation using only a few training images.

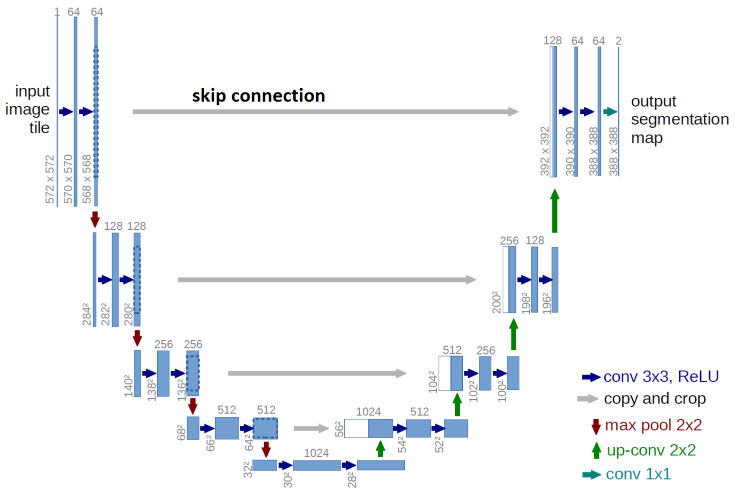

Fig. 4: U-Net architecture with skip connections. Figure extracted from [6].

3 Network Architecture

The visual perception architecture has the following components.

3.1 Encoder

Feature extraction can be performed effectively, in a time saving manner by utilizing pre-trained networks such as VGG net or ResNet. The encoder part of the network uses a pre-trained ResNet 18 since it has a lower model complexity as compared to the VGG net. A crucial aspect of choosing this architecture is that it also facilitates the gradient to be propagated effortlessly all the way until the lowest layers by means of skip connections. We remove the last GAP and fully connected layers to make the encoder fully convolutional, so that it can operate on an input image of any size and becomes easier to train with less number of parameters. Since the ResNet does not use any max-pooling, we do not store the pooling indices and reuse them while upsampling the images as in the case of SegNet.

3.2 Decoder

The decoder part consists of a combination of four transposed convolutions, batch normalization and relu non-linearity at each upsampling stage. The output from the encoder is first passed through the non-linear transformation and then upsampled using transposed convolutions. The upsampling is followed by a concatenation with an output from the corresponding stacked convolutional block from the encoder ResNet 18. This type of lateral connection similar to UNet helps the decoder recover spatial information that was lost during the downsampling process. It is to be noted that we maintain the number of channels in the lateral convolutional layers and in the transposed convolutional layer of the decoder to be 128 with the only exception being the last layer which has the number of layers equal to the number of classes. (4 - 3 classes for each for ball, field line, field plus one background class) The network outputs a downsampled heat map of the image. The smaller size of the decoder and output helps in achieving real time computation and inference by minimizing the number of parameters in the network according to [1].

4 Training

4.1 Dataset

The dataset consists of around 2011 images of robocup soccer playing bots in the soccer field. The three classes ball, field lines and field were manually annotated using MATLAB image labeller tool in the Computer Vision toolbox. Capturing all the images at one stretch often introduces a bias in the dataset as most images end up being taken under the same real-world conditions. As a result, we ended up creating the dataset progressively over many days, and training the model simultaneously so as to get further insights on diversity of the dataset. Among the 2011, we had to discard a few images during training because we found them irrevelant and have a degrading effect on the model's performance (For example, a long shot top view of the stadium, which will never be seen by the robot). Hence, 1994 images were used for training and 499 were set aside as the validation data.

4.2 Supervised pre-training

The encoder part of the network is built using a pre-trained ResNet 18 available in pytorch torchvision model zoo. We first freeze the weights of this pre-trained network and train only the decoder part with the lateral connections with low resolution images of size 320 $\times$ 160 for the first 30 epochs.

4.3 Domain specific fine-tuning

After 25 epochs, we then unfreeze the weights of the pre-trained ResNet in the encoder and train the whole network with the low resolution images for another 15 epochs. After this we train the network until 80 epochs using full resolution images 640 $\times$ 480 and a L2 weight decay of 0.0005. The number of epochs were chosen based on the improvement of model's performance through repetitive training trails. The number of training epochs can be increased with the increase in training data.

This method of training the network proves to faster and effective and we complete the whole process in on 1 Nvidia GTX 1070 gpus. We use the Adam optimizer and the learning rate is chosen according to the approach followed by Smith [8]. The final output of the network is fed to a pixel-wise classification softmax layer and the whole model is optimized using a pixel wise cross-entropy loss.

5 Results

The major contribution of this work is the implementing and training the model suggested by [1] with minor modifications so that it performs semantic segmentation and creation of the dataset. The experiments are focused on validating the effectiveness of the model and the dataset. In the following sections we discuss the results of the evaluating the model using the Robocup Soccer dataset and the metrics used for evaluation.

5.1 Metrics

We use the Jaccard Index (JI) to measure the Intersection over Union (IoU) and evaluate our model in contrast to overall pixel accuracy because of the presence of highly imbalanced classes in our dataset. The metrics used for evaluation are mean Intersection over Union (mIoU) and inference time.

– **mIoU**: Mean Intersection over Union is an accuracy metric used for semantic segmentation. IoU of a particular class is the ratio between pixels shared between the ground truth and the prediction to the total pixels of the corresponding class belonging to the ground truth and prediction.

$$IoU = \frac{ground\_truth \cap prediction}{ground\_truth \cup prediction}. \tag{1}$$

mIoU is the mean of IoUs of all classes in the dataset. IoU could also be interpreted as shown in equation (6.2) on the pixel level where, TP denotes True Positives which is the total number of pixels correctly predicted, FP denotes False Positives which is the total number of pixels which does not belong to a particular class according to the ground truth but is predicted as belonging to the class and FN denotes False Negatives which is the total number of pixels belonging to a class according to the ground truth but is predicted as not belonging to the class.

$$IoU = \frac{TP}{TP + FP + FN} \tag{2}$$

When the ground truth and predictions have no intersecting pixels for a class, the corresponding class IoU will be 0 percent. Similarly, if there is a perfect union between the ground truth and the prediction, the corresponding class IoU will be 100 percent.

5.2 Comparing Individual Classes

The relationship between the individual class IOUs and the percentage of pixels occupied by each class in the dataset can be inferred from this experiment. With the increase in the percentage of pixels, the class IOU is expected to increase. This expectation is based on the notion that the segmentation model gives preference to objects which dominate the dataset. For each class, the mean class IOU over 80 epochs is considered as its class IoU. Both the percentage of pixels and the class IOU is normalized to lie in the range of [0, 1]. The classes are arranged in increasing order of percentage of pixels and values are given by Table 1.

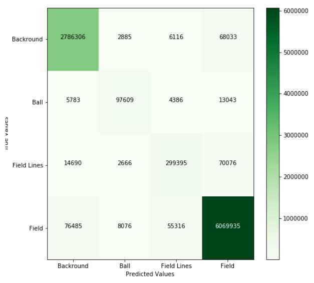

Fig. 5: Confusion matrix

Classes	Train IoU	Val IoU
Field	95.9%	95.4%
Field Lines	62.7 %	64.7%
Ball	73.2%	73%
Mean IoU - 81.9%		

Table 1: Class IoU scores

5.3 Confusion matrix

The confusion matrix depicts the model's inability to distinguish between different classes. The leading diagonal elements of the confusion matrix have the highest values in each row (fig 5). This result is expected and suggests that the model correctly classifies a majority of pixels in each class. The model is able to correctly identify the field and background classes as they occupy majority of the region in the image.

6 Conclusion

We progressively created a dataset and introduced diversity into it by inference through model training and then adding and discarding images from the dataset. We used a pretrained network for the encoder and a smaller sized decoder with tranposed convolutions is used to upsample the feature maps. Training is done stage-wise by freezing and unfreezing the weights and using low resolution and later high resolution images. Finally the performance of the segmentation model is measured in terms of mean Intersection over Union for each class. We attribute the lower performance of the model to the field line and ball classes because they occupy a smaller percentage of the image compared to the other classes. Further, the diversity of images introduced in our limited dataset makes it difficult for the model to learn the with imbalanced classes. We believe that our model would provide a better performance with more training data.

References

1. G. Ficht, H. Farazi, A. Brandenburger, D. Rodriguez, D. Pavlichenko, P. Allgeuer, M. Hosseini, and S. Behnke, "Nimbro-op2x: Adult-sized open-source 3d printed humanoid robot,"
2. K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.
3. J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 779–788, 2016.
4. A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, pp. 1097–1105, 2012.
5. 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*, pp. 3431–3440, 2015.
6. V. Badrinarayanan, A. Kendall, and R. Cipolla, "Segnet: A deep convolutional encoder-decoder architecture for image segmentation," *IEEE transactions on pattern analysis and machine intelligence*, vol. 39, no. 12, pp. 2481–2495, 2017.
7. 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*, pp. 234–241, Springer, 2015.

8. L. N. Smith, "Cyclical learning rates for training neural networks," in *Applications of Computer Vision (WACV), 2017 IEEE Winter Conference on*, pp. 464–472, IEEE, 2017.