Semantic Instance Segmentation for Classroom Environments

Kobus Van der Walt
University Of The Witwatersrand
Johannesburg, South Africa

Richard Klein
University Of The Witwatersrand
Johannesburg, South Africa

***Abstract*—In this paper we show how semantic instance segmentation can be applied to classroom environments. We discuss the difficulties involved with the environment and introduce a new hand labeled dataset for validation of this task. A single Fully Convolutional Network is trained end to end, followed by a simple watershed-like post-processing step. This yields accurate segmentation masks on an instance level. We achieve a mean IOU of 53%, which is satisfactory given the complexity of the task.**

***Index Terms*—Semantic Instance Segmentation, Classroom Environment, Fully Convolutional Network, Computer Vision**

I. Introduction

Since the introduction of deep learning based computer vision methods, the field has advanced at an rapid pace. First came image classification, then object localization and detection, followed by dense pixel predictions from semantic segmentation, and more recently semantic instance segmentation (SIS) which is the focus of this work. This section explores the SIS task, and provides background on preceding work and the journey to SIS.

A. Image Classification

Image classification can be defined as the task of assigning a class label to a given input image. The output of the network is usually a vector with length equal to the number of classes. Each element of this vector contains a probability that the image belongs to that particular class. Between the input and output layers of the network are hidden layers that consist of non-linear activation functions. It is this non-linearity which allows the network to learn a hierarchical approximation of the function which maps the input pixels to the output class probabilities. Convolutional Neural Networks (CNNs) introduced the idea of learning which features to extract and pass on to the deeper layers of the network [1]. A convolutional layer has a number of benefits over a fully-connected layer when extracting features that will be used later in the network. Firstly, instead of mapping every input node to every output node, a convolutional layer uses a set of convolutions which greatly reduces the number of parameters that need to be trained. The second attractive attribute is the translation invariant nature of convolutions. Whether an object is at the top or bottom of the input image is irrelevant to the features being extracted. The response from the convolution will be the same regardless of the position. These properties have lead to CNNs being used in almost all modern machine learning models.

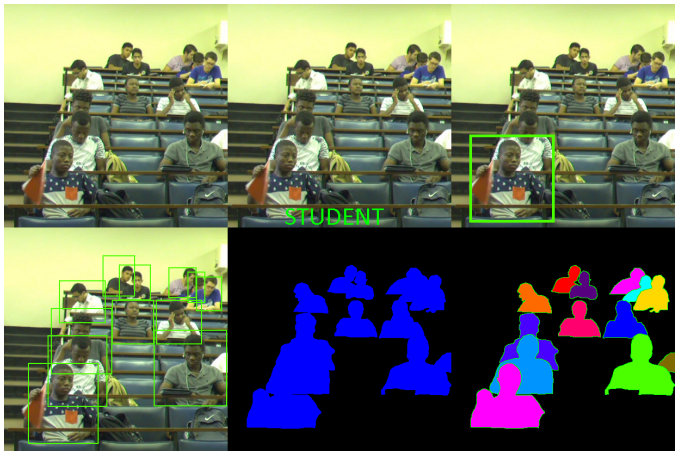


Fig. 1. The original input image at the top left, the result of image classification top center, object localization top right, object detection bottom left, semantic segmentation bottom center, and finally semantic segmentation bottom right.

Although CNNs have been around since they were introduced by [1], it wasn't until 2012 when Krizhevsky et al. [2] won the ILSVRC competition by a large margin that deep learning truly took off. In their work they show that large, deep convolutional neural networks are significantly better at solving the image classification task than any previous approaches. The increase in computing power available provided by graphics processing chips, and emergence of large annotated datasets, were key contributors to this discovery. To a lesser extent innovations such as the Relu activation function, and dropout normalization also aided in the feasibility of the solution. Alexnet (as the paper is now known) was could be considered the moment when the stars aligned for deep learning based computer vision.

B. Object Localization

Object Localization is a natural evolution of the image classification problem. Once we are able to assign a class label to an image, the next question becomes "where is the object?". Object Localization produces a bounding box around the classified image allowing the background to be disregarded.

This is useful in that a bounding box cut image is centered which results in less variation to later steps of a pipeline.

C. Object Detection

In most cases an image will contain multiple objects of different classes, and multiple instances of the same object class. Being able to localize and classify each object instance is a much more useful solution for higher level algorithms than merely knowing what is within an image. Object Detection or Object Recognition attempts to identify all objects of interest in an image, and subsequently classify them.

Before deep learning methods became popular, the focus of object detection was more towards determining better hand designed features which could be used for classification. One example of this is the Histograms of Oriented Gradients (HOG) detector [3]. The HOG detector performs localization through a sliding window sweep over the entire image, while downsizing the image multiple times to allow scale invariance. Another method to perform object detection is through a cascade of simple features. This was popularized by Viola and Jones [4], where the validity of the approach was demonstrated by building classifier to classify and localize faces in an image. The cascade approach by Viola and Jones [4] is distinct to sliding a window over multiple resolutions, in that it does not use pixel values directly. Instead a large number of simple rectangular features are computed at multiple resolutions and locations. This is feasible due to the once off construction of an integral image, allowing the features to be computed in constant time regardless of the image resolution.

An early approach to deep learning object detection involved dividing the input image into a grid of evenly spaced windows, and then performing classification for each resulting window [5]. This approach was superseded by networks which classify region proposals of variable dimensions that can better fit possible objects in the image. One such approach is described by [6] where a network is trained to output a fixed number of bounding boxes and a confidence score representing the probability of a bounding box containing an object. The classifier is then run on each bounding box proposal which meets some confidence threshold.

An evolution of the region-based classification method is presented by [11] as Fast-RCNN. Fast-RCNN introduces a single stage training process, which increases detection quality and execution speed of previous works. This was improved upon by Faster R-CNN, which utilized a region proposal network and direct feature classification, to increase the inference speed [12].

D. Semantic Segmentation

Being able to localize and classify objects in an image is already valuable. However, within a bounding box there are pixels that do not belong to the object that was detected. Semantic segmentation is the process by which a dense segmentation mask is produced which identifies each individual pixel within an image as belonging to a particular class of object. As with object detection there are sliding window methods to achieve this [7], but these methods have been surpassed by Fully Convolutional Networks.

A Fully Convolutional Network (FCN) is a network that consists solely of layers which are convolutional in nature. This results in an architecture that can receive an arbitrarily sized input and produce a dense per pixel prediction of the equivalent size. Image classification networks normally have fully-connected layers in the final layers which interpret the features extracted by earlier convolutional layers. This imposes a requirement for the network input to be of a specific size, since a fully connected layer has a weight for each input-output combination and this requirement is propagated forward to the earlier layers. The application of FCNs to semantic segmentation was first introduced by Long [8] where it was shown that FCNs are particularly well suited to dense prediction tasks.

It is with semantic segmentation that the core challenge of our work becomes evident. Classification and localization are at odds with each other. The nature of deep hierarchical networks, is the mapping from the local feature space to the global context space. The deeper you go within a network the more abstract the features become. This is why these networks excel at classification tasks, but it also introduces difficulties semantic segmentation domain. Consider for example a persons face. On a very low level a face is made up of edges and corners. A combination of particular edges and corners might represent the leftmost point for an eye. A combination of this feature along with features describing the top, bottom and right of an eye, would represent an eye. Similarly this chain follows for the mouth and nose of a person. The combination of eyes, a nose, mouth and ears describes a persons face. With this level of abstraction we are deep into the network structure and have no indication of what pixel belongs to the face or not. This characteristic of semantic segmentation is observed by Long et al. [8], and skip connections are presented as a way to address the absence of localization information.

Another network which builds on the success of FCNs is U-Net [9]. The network architecture consists of an encoder-decoder like structure with the first half downsampling features to gain an understanding of 'what' is in the image. And the second half upsampling features to regain localization information. To enable precise edge reconstruction the network takes the idea of skip connections further and incorporates an upsampling layer for each corresponding downsampling layer, while providing skip connections between these dimension-altering layers.

E. Semantic Instance Segmentation

Semantic Instance Segmentation which attempts to produce a segmentation map while distinguishing between different instances of the same object class. There are primarily two approaches to semantic instance segmentation.

The first approach leans on established object detection methods mentioned earlier, to produce a bounding box of all detected objects in a scene. A segmentation mask is then produced for each of these bounding boxes.

The second approach is to use the FCN directly to produce a semantic segmentation mask for the entire image, performing detection, classification and segmentation in a single network step. The segmentation map contains a number of masks for each object type and then instances are determined through a post processing step which finds all the connected pixels of a mask. The result is a series of connected components which represent different instances of a class. With this approach the classification and localization steps are combined into a single step, instead of first performing localization, then classification, and finally a high resolution segmentation map.

II. Related Work

Intuitively the direct FCN approach is better suited for semantic instance segmentation, since it is not limited to a fixed number of potential instances. Every pixel in an image could be a potential instance assuming there is some way to separate instances. Region-based approaches produce a fixed number of regions where objects could possibly be and, as such, they are limited in the number of instances they can detect. A tiling strategy or larger output layer could be used to address this, but our resistance to this approach is that it seems redundant and to first detect bounding boxes of objects in a scene and then produce segmentation masks for each bounding box. If we know what and where a pixel is, we should not need an artificial bounding box to inform us of which object instance it belongs to.

Nevertheless the current state-of-the-art for instance segmentation is Mask-RCNN which is based on a region-based approach [13].

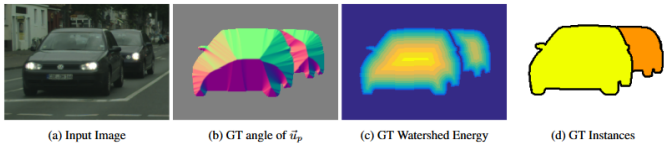
The direct FCN approach has a number of challenges when performing instance segmentation. The main difficulty is that instances of the same class are often close to each other, touching, or overlapping. The approach struggles in these scenarios since the pixels it is classifying are technically correct, but the result is not able to produce segmented instances if no borders are present. There are some approaches for dealing with this issue.

UNET introduces a weight map which calculates the distance from the current pixel to the nearest cells [9]. The lower the distance the higher penalty for misclassifying that particular pixel. This penalty incentives the network to pay close attention to pixels where cells might overlap with other cells.

Another popular idea has been to use an additional output channel to predict object boundaries. Iglovikov et al. [16] Used two output channels for satellite imagery instance segmentation. The first channel is the predicted output mask. And the second channel predicts touching instances. The second channel is then subtracted from the first channel and fed through the watershed transform to seperate instances.

The use of borders to separate instances was also used by Kirillov [17] where two concurrent branches within the network predicts the class and edge mask respectively.

A novel approach which is similar to edge prediction in spirit was proposed by Bai and Urtasun [18]. The approach



(a) Input Image (b) GT angle of $\vec{u}_p$ (c) GT Watershed Energy (d) GT Instances

Fig. 2. Deep watershed prediction ground truth. The predicted direction of the edge is shown by b) which is then used as input to the network predicting the watershed energy shown by c). Finally instances are cut to produce d)

requires the semantic segmentation mask, alongside the RGB image as input. he semantic segmentation mask is produced by any semantic segmentation network. As shown in Figure 2 a two stage neural network first determines the direction from the center to the edges, after which the watershed energy map is predicted.

Combinations of region-based, and edge detection methods are possible. Xu et al. [20] Proposed a network consisting of three branches predicting the segmentation mask, object boundaries and edges for biological gland segmentation. It is notable that Xu et al. [20] show that if any of the three branches are dropped the performances decreases. This supports our observation that classification and localization are working against each other within a network. With the three branch model each branch is can perform a single task without loss interference from a contradicting task.

III. Dataset

The classroom environment is a challenging instance segmentation task. There is a lot of variation between each student and the clothes they wear. Further complicating matters is the variation in classrooms. There is also a high level of overlap between students which also increases the difficulty of instance extraction.

Raw video recordings we're obtained from previous work involving a system to measure student engagement [21]. We hand labeled 50 of these images, annotating the student, separation border and student face. Each image takes around hour to label making this a very time consuming process. There are 10 different classroom environments of which some had multiple view points and classes. The result is 20 videos. We labeled one image of every video, except the first video which has 30 labeled images, and then rectified the class imbalance by oversampling from the underrepresented videos until the distribution is equal.

IV. Network Architecture

Our network structure is primarily based on UNET while being updated with some improvements from recent work. A common problem with FCN architectures and convolutional networks in general is the lack of scale invariance. FCNs are remarkably good at learning scale invariance but the network does not specifically cater for scale invariance. Max pooling layers are common in CNNs an help to increase the receptive field of a network but again it produces only one level of scale as it's output.

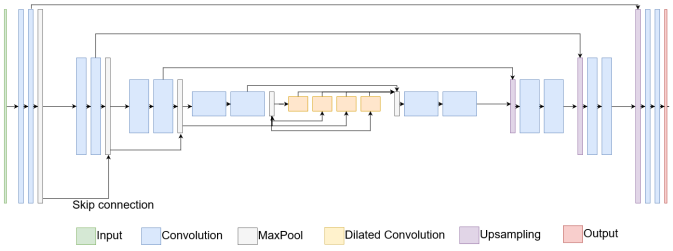


Fig. 3. The architecture of our network. The dilated convolutions are of stride 1, 2, 4, and 8. Before and after the dilated convolutions we have a dropout layer as well which is not indicated on the figure.

Dilated convolutions introduce a stride within the convolutional kernel, effectively increasing the receptive field without increasing the number of weights or computation required. We follow the work by [25] and introduce multiple dilated convolutions at the core of the network. This results in a set of output features at 4 different levels of scale, which are concatenated and propagated forward to the decoder network.

The other modification from UNET was to introduce residual connections from [23]. Residual connections were shown to have an increase in accuracy for FCNs by [25]. We add 3 skip connections to the encoder part of UNET.

We use multiple output channels which produce different semantic information that is used in the post processing step to recover individual instances. The first output channel is the probability of a pixel belonging to a student. The second output channel is used to predict where students overlap. And the final output channel describes the entire edge of a student. We use two channels for edge detection since it is a critical step to recovering instances. By forcing the network to focus on where students overlap we double down on the importance of student edges in a semantically distinct way.

The activation function for the hidden layers is the now standard ReLU [22]. But we we're concerned about dying neurons so we opted for the negative gradient of 0.1 instead of 0, also commonly known as a "leaky-ReLU".

For the final activation function we used the sigmoid activation function which produces the probability distribution across each channel, as opposed the commonly used softmax which produces a probability distribution across all output channels. We specifically chose sigmoid over softmax activation as we view our output channels as separate output branches. In [14], two distinct branches were used. One branch was responsible for semantic edge detection, much like our final 2 output channels, and the other branch predicted the segmentation mask of the human. Our task is much more focused, thus we can avoid a multiple branch network.

A. Training

FCN architectures often uses the categorical cross-entropy loss function and a final softmax activation layer to predict the segmentation class for each mask. The softmax activation function outputs a probability of the pixel belonging to a specific class and thus the loss function encourages the network to select only one class for each pixel. Normally this is a desirable property, but the nature of our output channels is that a pixel may belong to more than one class. A pixel might be part of a student and the edge of a student as well as be at a position where it is close to other students. The term multi label output might be a better description of our output. In this case the potential missclassification of the pixel as belonging to a student instead of a border, would make it impossible to separate the student into it's own instance; since the student and the bordering student are now one connected component.

As mentioned earlier we instead opted for a sigmoid activation over each output channel with the loss being a weighted sum of the binary cross-entropy loss function for each output channel. When using vanilla binary cross-entropy loss, we observe that the network first learns to accurately predict the student pixels, before starting to produce the border pixels. This is reasonable since the error from the student pixels dominates the final loss and only later does the network have to perform accurate border prediction to improve. To balance the progress across all output features we perform weighted binary crossentropy on the border layers, favouring false positives over false negatives. Additionally to encourage crisp edges and force a more even loss distribution we add a DICE loss to every channel. The full loss function is shown in equation 3.

$$loss1 = -(\alpha_c \cdot y \cdot log(\hat{y}) + (1-y) \cdot log(1-\hat{y}) \quad (1)$$

$$loss2 = 1 - \frac{2\sum y \cdot \hat{y}}{\sum y + \sum \hat{y}} \quad (2)$$

$$finalLoss = \frac{-\sum_{c=1}^{M} \beta \cdot loss1_c + (1-\beta) \cdot loss2_c}{M} \quad (3)$$

Where M is the number of output channels, y the ground truth label, $\hat{y}$ the prediction and c the respective channel, α is a constant unique to each channel, beta is a global constant.

For our tests we set α to 1.5 for channel 2 which predicts the instance overlap, and 1.5 for channel 3 which predicts the instance border. We set β to 0.8.

To speed up the training process we had to somewhat standardize the images to have the same resolution. We do this by randomly cropping a square tile of between 128 and 640 pixels from the original image and resizing it to 512 pixel. Our dataset is also exceptionally small for deep-learning purposes, so we performed a number of data augmentation techniques including random cropping, zooming, sheering, rotation, random brightness and contrast shifting, optical distortion and finally elastic deformations described by [10].

We train on tiles of 512x512 pixels and predict on the full image at inference time, rounding down to the nearest number divisible by 32.

The the ADAM optimizer was used with learning rate 0.0001 and trained for 70 epochs before satisfactory results

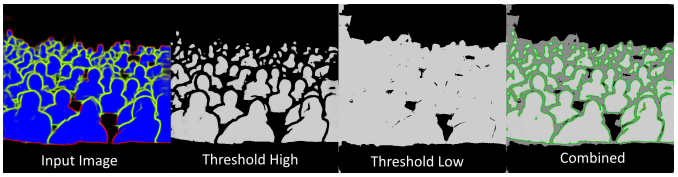


Fig. 4. An illistration of the post processing step applied.

were obtained. An epoch consisted of 200 random samples from the equally represented dataset.

B. Post Processing

We fundamentally have two problems when dealing with the output of the network. The first is determine student instances. We can do this effectively by using a high threshold on the student channel, and a low threshold on the border channels. But this results in poor segmentation masks. If however we lower the threshold it becomes increasingly difficult to determine instances. To address this we produce two masks from the output. The first mask is created with a high threshold and the second mask with a low threshold. Once we have determined instances, we dilate the high threshold mask to expand significantly and use this as a mask for the lower threshold mask. This is similar to the watershed segmentation method.

V. Experiment

A. Quantitative Results

To the measure the performance of our network on the dataset we split our data 70:20:10 to get training, validation and testing sets respectively. The precision metric is an indicator showing how classification accuracy of the selected pixels. Recall shows how accurate the selection of pixels are.

$$Precision = \frac{TP}{TP + FP} \tag{4}$$

$$Recall = \frac{TP}{TP + FN} \tag{5}$$

Where TP, FP, FN is the true postives, false positives and false negatives respectively.

A common metric used to measure the performance of semantic segmentation is the Intersection Over Union or Jaccard index.

$$Jaccard(A, B) = \frac{|A \cap B|}{|A \cup B|} \tag{6}$$

TABLE I
Final evaluation metrics on test set

Metric	Score
Loss	0.2123
Precision	0.8462
Recall	0.9030
Mean IOU	53.39%

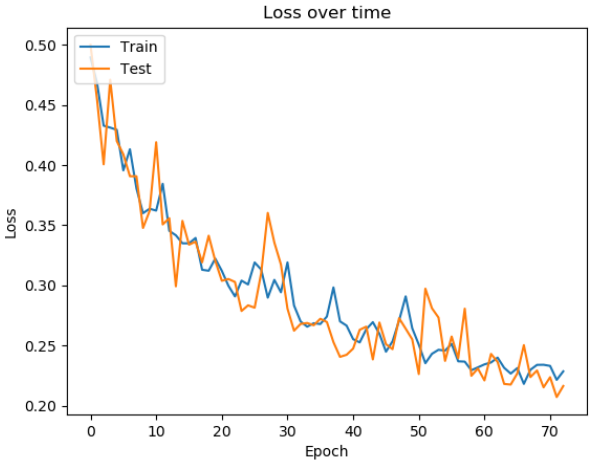


Fig. 5. Loss over 70 epochs.

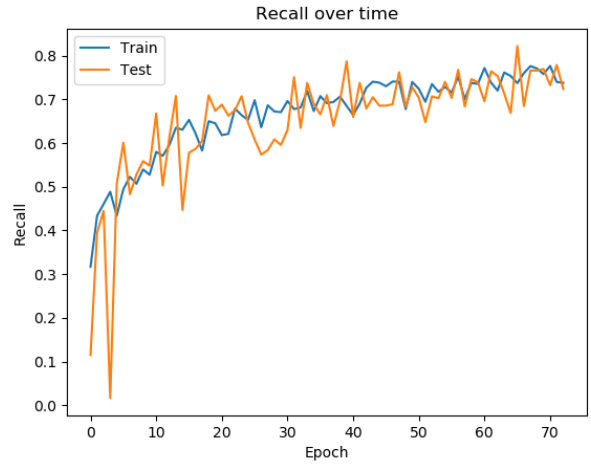


Fig. 6. Recall metric over 70 epochs.

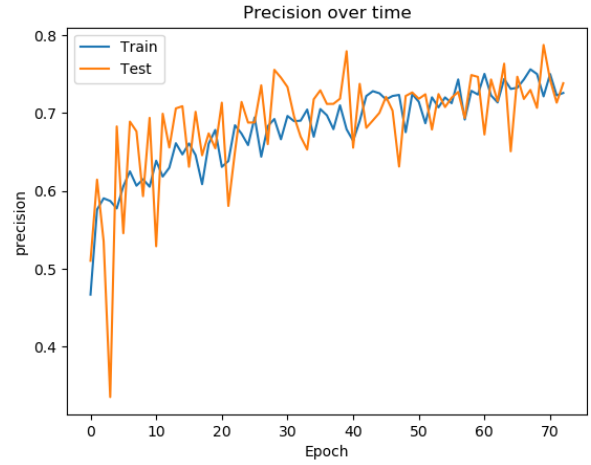


Fig. 7. Precision metric over 70 epochs.

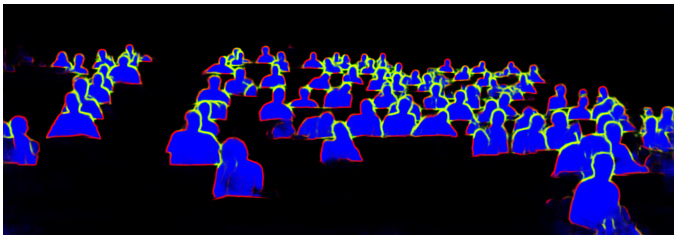


Fig. 8. Network output on an unseen frame from seen environment.

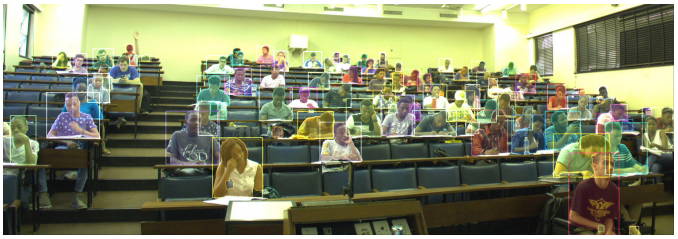


Fig. 9. Final result on an unseen frame from seen environment.

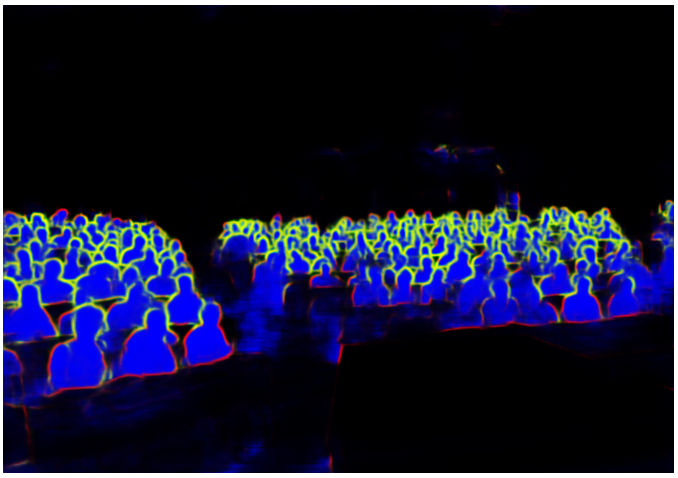


Fig. 10. Network output on an unseen frame from unseen environment.

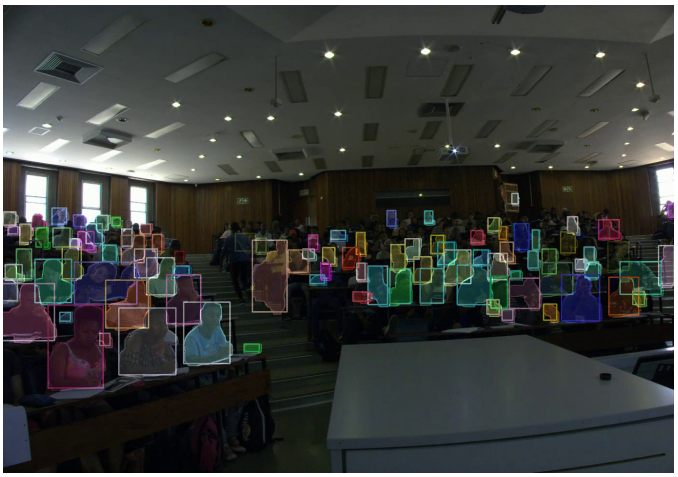


Fig. 11. Final result on an unseen frame from unseen environment.

B. Qualitative Results

VI. Conclusion

We introduced the classroom instance segmentation task and provided a labeled dataset which can be used for performance evaluation. Then we described a Fully Convolutional Network including state of the art techniques including residual connections and dilated convolutions, showing that this architecture is well suited to solving the instance segmentation task. Following the output of the network, a simple post processing step which decreases the risk of erroneous instance combinations, while keeping high resolution borders, is applied.

VII. Acknowledgement

The authors would like to thank Martin Fourie and Zanro De Beer for assistting in the data labeling process.

References

[1] Sermanet, P., Chintala, S. and LeCun, Y., 2012. Convolutional neural networks applied to house numbers digit classification. arXiv preprint arXiv:1204.3968.

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

[3] Dalal, N. and Triggs, B., 2005, June. Histograms of oriented gradients for human detection.

[4] Viola, P. and Jones, M., 2001. Rapid object detection using a boosted cascade of simple features. CVPR (1), 1(511-518), p.3.

[5] Sermanet, P., Eigen, D., Zhang, X., Mathieu, M., Fergus, R. and LeCun, Y., 2013. Overfeat: Integrated recognition, localization and detection using convolutional networks. arXiv preprint arXiv:1312.6229.

[6] Erhan, D., Szegedy, C., Toshev, A. and Anguelov, D., 2014. Scalable object detection using deep neural networks. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 2147-2154).

[7] Ciresan, D., Giusti, A., Gambardella, L.M. and Schmidhuber, J., 2012. Deep neural networks segment neuronal membranes in electron microscopy images. In Advances in neural information processing systems (pp. 2843-2851).

[8] Long, J., Shelhamer, E. and Darrell, T., 2015. Fully convolutional networks for semantic segmentation. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 3431-3440).

[9] Ronneberger, O., Fischer, P. and Brox, T., 2015, October. U-net: Convolutional networks for biomedical image segmentation. In International Conference on Medical image computing and computer-assisted intervention (pp. 234-241). Springer, Cham.

[10] Simard, P.Y., Steinkraus, D. and Platt, J.C., 2003, August. Best practices for convolutional neural networks applied to visual document analysis. In Icdar (Vol. 3, No. 2003).

[11] Girshick, R., 2015. Fast r-cnn. In Proceedings of the IEEE international conference on computer vision (pp. 1440-1448).

[12] Ren, S., He, K., Girshick, R. and Sun, J., 2015. Faster r-cnn: Towards real-time object detection with region proposal networks. In Advances in neural information processing systems (pp. 91-99).

[13] He, K., Gkioxari, G., Dollár, P. and Girshick, R., 2017. Mask r-cnn. In Proceedings of the IEEE international conference on computer vision (pp. 2961-2969).

[14] Gong, K., Liang, X., Li, Y., Chen, Y., Yang, M. and Lin, L., 2018. Instance-level human parsing via part grouping network. In Proceedings of the European Conference on Computer Vision (ECCV) (pp. 770-785).

[15] Zhao, J., Li, J., Cheng, Y., Sim, T., Yan, S. and Feng, J., 2018, October. Understanding humans in crowded scenes: Deep nested adversarial learning and a new benchmark for multi-human parsing. In 2018 ACM Multimedia Conference on Multimedia Conference (pp. 792-800). ACM.

[16] Iglovikov, V., Seferbekov, S.S., Buslaev, A. and Shvets, A., 2018, June. TernausNetV2: Fully Convolutional Network for Instance Segmentation. In CVPR Workshops (pp. 233-237).

[17] Kirillov, A., Levinkov, E., Andres, B., Savchynskyy, B. and Rother, C., 2017. Instancecut: from edges to instances with multicut. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (pp. 5008-5017).

[18] Bai, M. and Urtasun, R., 2017. Deep watershed transform for instance segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (pp. 5221-5229).
[19] Hayder, Z., He, X. and Salzmann, M., 2017. Boundary-aware instance segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (pp. 5696-5704).
[20] Xu, Y., Li, Y., Liu, M., Wang, Y., Fan, Y., Lai, M. and Chang, E.I., 2016. Gland instance segmentation by deep multichannel neural networks. arXiv preprint arXiv:1607.04889.
[21] Klein, R. and Celik, T., 2017, September. The Wits Intelligent Teaching System: Detecting student engagement during lectures using convolutional neural networks. In 2017 IEEE International Conference on Image Processing (ICIP) (pp. 2856-2860). IEEE.
[22] Abien Fred, M.A., 2018. Deep Learning using Rectified Linear Units (ReLU). Neural and Evolutionary Computing, 1.
[23] He, K., Zhang, X., Ren, S. and Sun, J., 2016. Deep residual learning for image recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 770-778).
[24] Jégou, S., Drozdzal, M., Vazquez, D., Romero, A. and Bengio, Y., 2017. The one hundred layers tiramisu: Fully convolutional densenets for semantic segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops (pp. 11-19).
[25] Yamashita, T., Furukawa, H., Yamauchi, Y. and Fujiyoshi, H., Multiple Skip Connections and Dilated Convolutions for Semantic Segmentation.
[26] Yu, F. and Koltun, V., 2015. Multi-scale context aggregation by dilated convolutions. arXiv preprint arXiv:1511.07122.