CS230

Kaggle Image Segmentation Challenge

Yokila Arora
ICME
Stanford University
yarora@stanford.edu

Hubert Lu
Department of Mechanical Engineering
Stanford University
hubertlu@stanford.edu

Eley Ng
Department of Mechanical Engineering
Stanford University
eleyng@stanford.edu

Abstract

One of the most important areas of research in autonomous driving is real-time and precise object detection, which is the topic of the 2018 CVPR Kaggle Challenge. Perception in autonomous vehicles can benefit from image segmentation algorithms that can not only quickly detect where objects such as pedestrians, traffic signs, and other vehicles are, but also their geometries such that the car can interact with the other objects and tell them apart from one another. Therefore, we apply the Mask R-CNN model to achieve pixel-level segmentation as well as detection of instances of objects. Assisted by transfer learning using COCO Dataset pre-trained weights, the Mask R-CNN model trained with ResNet50 layers as the network heads produced the most favorable results, giving a Kaggle submission score of 0.03607 and placement of 56th on the leaderboard.

1 Introduction

In light of recent advancements in autonomous car research, and particularly in the field of computer vision, the 2018 CVPR workshop on autonomous driving is hosting a Kaggle competition in image segmentation using images taken from video streams collected by autonomous vehicles [1]. We explored several models before selecting our model of choice, the Mask R-CNN.

2 Related work

Object detection has made massive strides with the use of convolutional neural networks (CNNs). CNNs have been further advanced by using region proposal methods in which candidate bounding boxes project the probability of an object inside the box. However, CNNs cannot localize objects well within an image; the sliding window detection scheme does not fare well deeper in the network. To solve this, a two-stage concept was introduced into the CNN framework; first, the region proposal stage, and then a detection stage. The R-CNN model uses the region proposal concept by generating a number of regions and resizing each region before passing them into a CNN to be classified using a softmax layer[2]. Faster methods have been proposed with little changes (such as keeping the region proposals unwarped), yet highly improved results. These models include the Fast R-CNN and the Faster R-CNN [3],[4].

CS230: Deep Learning, Winter 2018, Stanford University, CA. (LateX template borrowed from NIPS 2017.)

Other developments in CNNs focus on single stage detection methods that do not use region proposals work. YOLO (You Only Look Once) detection uses features from an entire image to predict bounding boxes and is faster in speed and training than several R-CNN models [5], [6]. However, the accuracy of YOLO has been known to be a work-in-progress, and it does not provide the pixel-wise segmentation that is critical for autonomous vehicles to interact with its environment with instance-level knowledge of the environment.

3 Method

We performed transfer learning for the task of object detection and segmentation using the Mask R-CNN model [7]. We utilized the open-source code provided by matterport [8]. We use pre-trained weights from COCO dataset and only train the last layers (ResNet50 and ResNet101). Since COCO weights have been trained on a larger number of objects, it helped our model to take advantage of the features learned using COCO dataset.

3.1 Dataset and Features

The Kaggle competition supplies datasets provided by Baidu Inc. consisting of color and labeled images. The color images were extracted from video collected by an autonomous car. There are 39,222 RGB images with their corresponding labels (i.e. 39,222 segmented images). We use a smaller subsets of this data for training, and 1% of it as validation set. Along with this, there are 1,917 test images provided which are used for evaluation.

From the labeled images, the information of object classes (labels) and segmentation masks can be obtained, as shown by the equations 1 & 2 below. For example, the pixel value of 33001, indicates the first instance of a car which has label 33, and 33002 indicates second instance. Though the dataset consists of multiple labels, for the challenge only seven instance-level annotations, which include car, motorcycle, bicycle, pedestrian, truck, bus and tricycle, are evaluated.

$$int(PixelValue/1000) = label \tag{1}$$

$$PixelValue\%1000 = instance\_id \tag{2}$$

3.2 Model architecture

The backbone of the Mask R-CNN is ResNet and Feature pyramid networks (FPN)[9]. The Mask R-CNN model inherits Faster R-CNN and modifies the RoI pooling layers by the Realigning RoIPool (RoIAlign) layers for the first stage of the feature map to achieve pixel-level segmentation and added another branch for mask generation. The basic architecture of the model is shown in Figure 1.

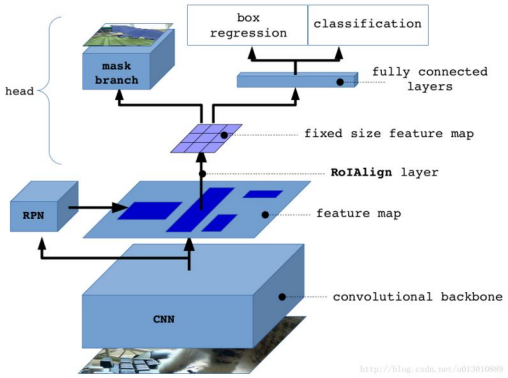


Figure 1: Architecture of Mask R-CNN [10]

In general, Mask R-CNN can be characterized by the following:

- Feature Extraction
  Mask R-CNN utilizes the convolutional layers and the region proposal network (RPN) to construct feature map. Since the RoI pooling (used in Faster R-CNN) may cause the location alignment for the use of mask segmentation (integer pixel representation), the RoIAlign adopting bilinear interpolation is used to circumvent the problems and precisely maps a region of interest from the original image onto the the corresponding feature map (can be floating pixel).
- Box Regression and Classification
  First, Mask R-CNN adds a Full Convolution Network (FCN) in order to address the possible problems that occur when some objects are overlapped and have overlapped bounding boxes. In the meantime, it keeps the same architecture as Faster R-CNN (for classification and detection). The outputs at the first step are the bounding boxes and the classes of objects.
- Mask Generation
  With the RoIAlign, He et al. [7] added another branch to generate the mask in parallel with the existing branches for object classification and localization (box regression). The binary outputs show a pixel whether belongs to a object ('1') or not ('0').

3.3 Loss function

The loss function is a multi-task function given as:

$$L = L_{cls} + L_{box} + L_{mask} \tag{3}$$

where L_{cls} is the classification loss, L_{box} is the bounding-box loss, and L_{mask} is the mask loss. L_{cls} and L_{box} are the same as those used in Faster R-CNN obtained by multi-class cross entropy and the smoothed L1 loss, respectively. L_{mask} is obtained by the average binary cross-entropy loss:

4 Experiments, Results, and Discussion

4.1 Kaggle Evaluation Details

We submit results in a csv file, where each submission line represents one object instance with the following columns: ImageId, LabelId, Confidence, PixelCount and EncodedPixels. Here, the ImageId denotes the image file name, the LabelId denotes the label of the object, pixelcount denotes the number of pixels of that object and EncodedPixels denotes the run length encoding of the object pixels. For each object, we get the binary masks of objects as output and we convert them into RLE format for submission.

In the Kaggle competition, the metric of object segmentation is the mean Average Precision (mAP) at different intersection-over-union (IoU) thresholds.

$$IoU(A, B) = \frac{A \cap B}{A \cup B} \tag{4}$$

where A and B represent a predicted instance and a ground truth instance B, respectively.

4.2 Experiments and Results

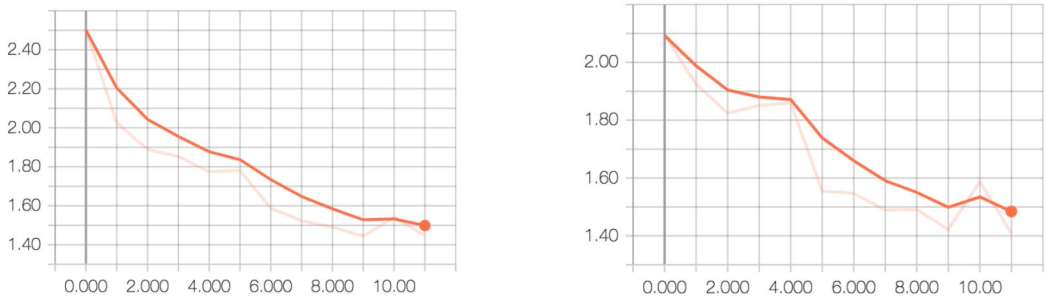
We first completed data preprocessing to convert our dataset into a format which can be fed into this model. The input training images are resized from 3384 $\times$ 2710 to 1024 $\times$ 1024 pixels before training. For each image, we generated a boolean mask with shape (height, weight, number of objects), and created a list of object labels of each object in the same order.

For the baseline model, we generated output with 1,917 test images using pre-trained COCO weights. Since the classes output from COCO are different, we mapped COCO class labels to our labels and filtered out only the seven required labels. The original model does not have the label "tricycle", so we omitted it. This submission gave us a score of 0.02538 and a ranking of 30 on the kaggle leaderboard.

Next, we trained the ResNet50 model (denoted as Model 1) with 100 images for first 5 epochs and 1000 images for the next 8 epochs. We used a learning rate of 0.001. The training and val loss values are described in Table 2, and are plotted in Figure 1. Model 1 gave the best kaggle score.

	Training Loss	Validation Loss	Submission score
Baseline	-	-	0.02538
Model 1	1.4182	1.4644	**0.03607**
Model 2	1.5459	1.4794	0.03098
Model 3	1.9589	2.1058	0.02763

Table 1: Training loss, Validation loss and Kaggle submission score



(a) Training loss

(b) Validation Loss

Figure 2: Loss plots for Model 1

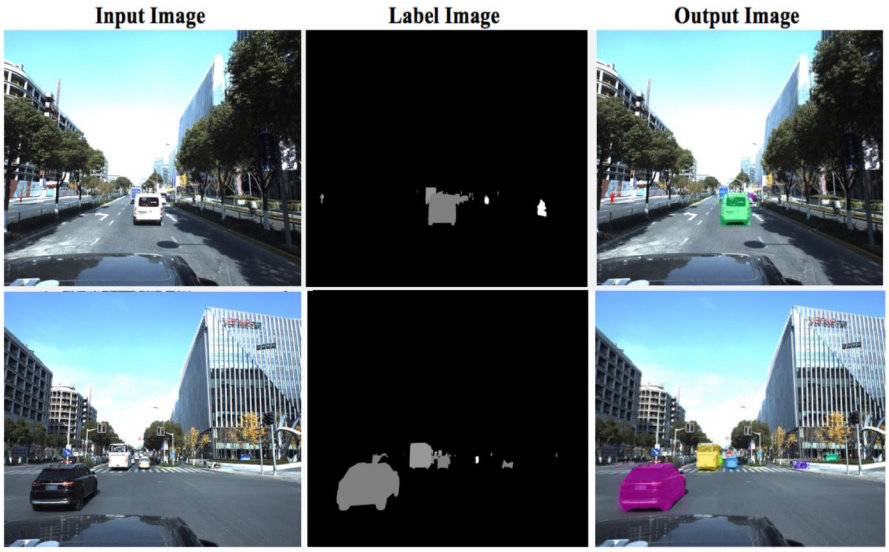


Figure 3: ResNet50 results on test set (top), ResNet101 results (bottom)

Then, we trained the ResNet101 model with the weights from Model 1. We used 1000 images and trained for 4 more epochs (total 4000 steps). We trained with learning rate 0.001 (denoted as Model 2) and 0.005 (denoted as Model 3). Figure 3 shows some training images along with corresponding labeled and output images.

4.3 Discussion

Although the results as shown in Figure 2 seem good, due to the size of the training data (approx. 91 GB), the model was trained with less images. Training with more images can lead to improvement in performance but also requires more time. Also, we noticed that the model doesn't detect smaller objects sometimes, as shown in Figure 4 the bicycle on the right side is not detected. This may be because the confidence threshold, which is set to 0.7, is too high, or because of the mask threshold parameter, which denotes the size of mask output, needs to be tuned. Tuning these parameters might

Figure 4: Missing "bicycle" classification example

improve the model. Since the loss has not stopped decreasing, training for more epochs can also lead in improvement. Due to time and cost constraints, we could not train more.

5 Conclusion and Future Work

We did transfer learning using Mask-RCNN model for object detection and segmentation. The output image show bounding boxes of all instances of objects for 7 classes, along with confidence scores. Although the model performs well, the performance can be improved by training with more images and tuning some parameters like mask size and confidence threshold, as described in Section 5.3 above.

We planned to add LSTM layers at the end of the model to take into account that the data is sequential. Using LSTM would help predict the next frame with the previous frame prediction as a feature. This way, the model could perform in real-time.

6 Contributions

Hubert Lu and Yokila Arora worked on training the model and modified and adapted the code for perform transfer learning for Kaggle competition. Eley Ng worked on data preprocessing, adapting code in the model, and model training. Most of the time every member in this team figured out the solutions of the problems faced together.

References

[1] Cvpr 2018 wad video segmentation challenge. `https://www.kaggle.com/c/cvpr-2018-autonomous-driving`. Accessed: 2018-05-12.

[2] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014.

[3] Ross Girshick. Fast R-CNN. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2015.

[4] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. In *Neural Information Processing Systems (NIPS)*, 2015.

[5] Joseph Redmon, Santosh Kumar Divvala, Ross B. Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 779–788, 2016.

[6] J. Redmon and A. Farhadi. Yolo9000: Better, faster, stronger. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6517–6525, July 2017.

[7] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Computer Vision (ICCV), 2017 IEEE International Conference on*, pages 2980–2988. IEEE, 2017.

[8] Mask r-cnn for object detection and instance segmentation on keras and tensorflow. `https://github.com/matterport/Mask_RCNN`. Accessed: 2018-05-12.

[9] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *CVPR*, volume 1, page 4, 2017.

[10] Deep object detectors. `https://www.slideshare.net/IldooKim/deep-object-detectors-1-20166`. Accessed: 2018-06-10.