Computer Vision Final Project

Tracy Chen, Bayard Walsh

March 2024

1 Objective

Building on the success of Facebook's DETR model [1], we applied the open source version of the End-to-End Object Detection Transformer model to the Cityscapes Data set [2]. Our code is documented in the Google Collab notebook submitted, where we ran our model, as well as the following repository that we modified DETR code to fit the Cityscapes data set: `https://github.com/bkwalsh/CV-final-project/tree/main`.

2 Method

2.1 Dataset

The datasets used are sourced from the Cityscapes Dataset, accessible at `https://github.com/mcordts/cityscapesScripts`. The datasets comprise:

- **gtFine_trainvaltest.zip** (241MB) [md5]: Contains fine annotations for the training and validation sets (3475 annotated images) and dummy annotations (ignore regions) for the test set (1525 images).
- **leftImg8bit_trainvaltest.zip** (11GB) [md5]: Includes left 8-bit images for the training, validation, and test sets (5000 images).

In order to load our data in the Collab Notebook, we used a simple bash script adopted from `https://github.com/cemsaz/city-scapes-script`

2.2 Dataloader for Cityscrapes dataset

When building our Dataloader, we were heavily inspired by both the Facebook Detectron team [3] and the Scripts provided by the Cityscapes team [2]. First we loaded a list of files as Numpy arrays and processed the unique instance IDs and then found the contours of the instance mask using OpenCV's findContours function, converting the images to a dictionary. Next, we transformed our dictionary of polygons to image and annotation dictionaries that match the COCO file data structure for object detection. Conversion also included

converting polygons to bounding boxes for training. One of the key decisions we made in our data loader was to restrict the objects to the following classes: `['person','rider','car','truck','bus','train','motorcycle','bicycle']` which is a reduced subset in the original 31 classes in the Cityscapes dataset - given the smaller size of our fine tuning data set and because we were training with less compute power than the initial DETR script, we wanted to be certain we could demonstrate an improvement in precision through our training epochs so we reduced the number of classes to achieve this.

2.3 Data Visualization

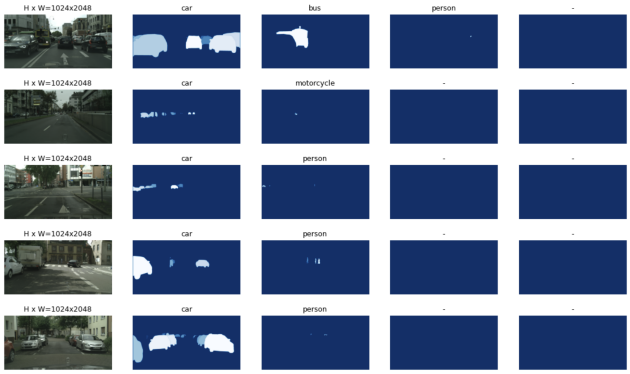

Figure 1: Example CityScapes Image Separated by Object Class

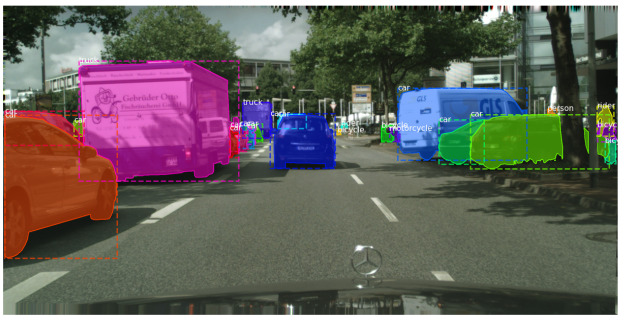

Figure 2: Example CityScapes Image with Bounding Boxes for polygons

Along with data loading, we added a visualization component to double check random data instances in the model to ensure the data loading processed correctly, and that the Cityscapes data was in the same format as the COCO dataset. This functionality is built on the COCO dataset class and visualization

`utils` [4] and involves showing the masks for each image (broken down by class instance) as well as the bounding boxes for each image. The visualization component of the script was built from the visualization `utils` in `https://github.com/matterport/Mask_RCNN/tree/master`

2.4 Adapt DETR structure

The original DETR model is trained on 80 classes for the COCO dataset, which we reduced to 9 for the subset of classes from the CityScapes dataset for our Fine-Tuned version of DETR (8 potential labels and 1 label for no object class). This was achieved through only loading the box annotations for the classes with the selected labels in our data loading phase, adding Cityscapes as a data loader and setting the classes to 9 in the model configuration.

2.5 Finetuning DETR

Run	Epochs	Learning Rate	Batch Size	Weight Decay
1	10	1×10^{-4}	2	1×10^{-4}
2	10	1×10^{-4}	3	1×10^{-4}
3	10	1×10^{-4}	2	1×10^{-5}
4	15	1×10^{-4}	2	1×10^{-4}
5	15	1×10^{-4}	3	1×10^{-4}
6	15	1×10^{-4}	2	1×10^{-5}

Table 1: Training parameters for DETR model fine-tuning on Cityscapes dataset

To fine-tune the DETR model using the Cityscapes dataset, we acquired the pre-trained DETR-resnet50 weights and removed the class-specific weights from the model to adapt it for our dataset. Our dataset was structured to follow the COCO format by the custom data loader. We modified the output class to be 9 in the finetuning process. The general workflow of finetuning and the model loading part of the finetuning code were adapted from `https://gist.github.com/m-klasen/651297e28199b4bb7907fc413c49f58f`. The code for finetuning is modified from the codes to train original DETR model to take in a new parameter of the number of classes and remove the original warnings in the DETR code that rejects datasets that are not COCO `https://github.com/facebookresearch/detr/blob/main/main.py`. In order to run the training script to fine-tune the model, we pulled from `https://github.com/bkwalsh/CV-final-project/tree/main`, which is a Github built on the original DETR script and modified to train datasets outsided COCO.
We tested six distinct configurations of hyperparameters, specifically varying epochs, batch sizes, and weight decay values, to assess their impact on model performance.

3 Results

All training runs were completed on a T4 GPU on Google Collab. After our

Figure 3: Example Detection Output on *The Busy City* (image outside CityScapes dataset) After Fine-Tuning (using the Fine-Tuned model from run 6)

6 training runs, we found that run 5 had the best precision, suggesting that increasing the batch size and increasing the number of epochs will result in better performance. As our precision epoch graphs indicate a direct correlation between number of epochs and model performance, given more time and data, we believe we could achieve higher accuracy by fine-tuning DETR further.

3.1 Attentional Behavior

In order to visualize the Transformer component's attentional behavior for detected objects within one of our images, we applied the functionality within `https://colab.research.google.com/github/facebookresearch/detr/blob/`

`colab/notebooks/detr_attention.ipynb`, which provides a heat map corresponding to the attention score for every object in an image.

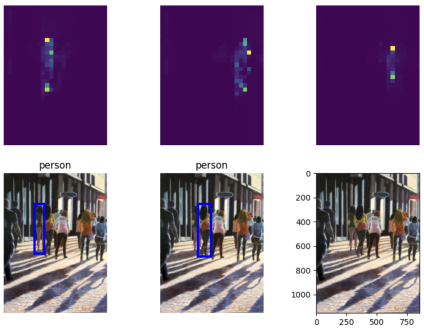

Figure 4: Example Attention Detection Output on *The Busy City* (image outside CityScapes dataset) After Fine-Tuning (using the Fine-Tuned model from run 6)

3.2 Benchmarking

For Instance-Level Semantic Labeling for this dataset, a high score of 44.5 precision was achieved using a HRI transformer instance segmentation model. In our best training iteration we achieved a score of 40.9 precision. However, as mentioned earlier, because of the scope of the project and the limits on compute resources, we reduced the number of classes in the dataset from 31 to 9, meaning our model had a far fewer potential labels to chose from, which reduces mislabeling error. Therefore our precision is not exactly comparable to the benchmark model instances, however a similar precision score to the benchmark suggests our model was able to achieve quality object detection performance after fine-tuning.

3.3 Training Runs

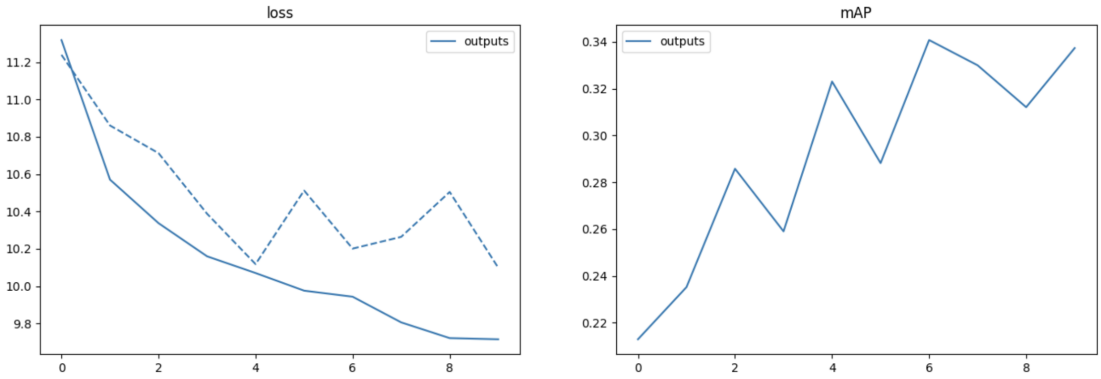

Figure 5: *Run 1:* Training time: 1:50:24. Precision 35.9 at Epoch 6

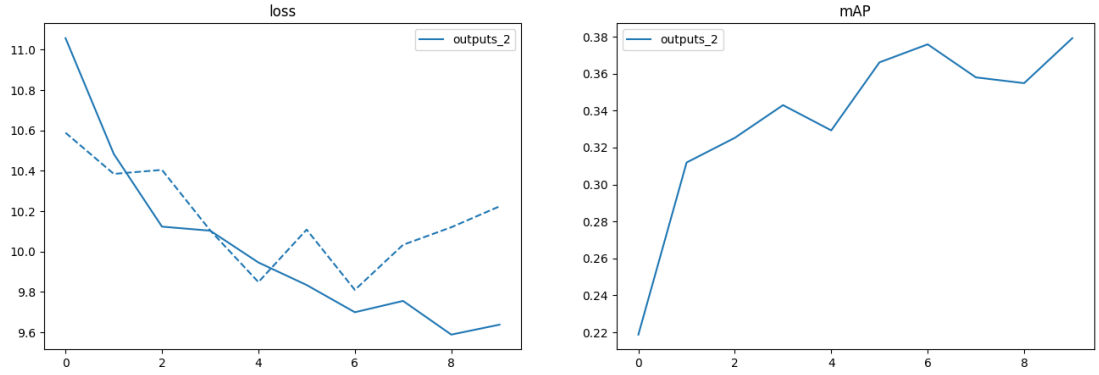

Figure 6: *Run 2:* Training time: 1:16:28. Precision 37.9 at Epoch 9

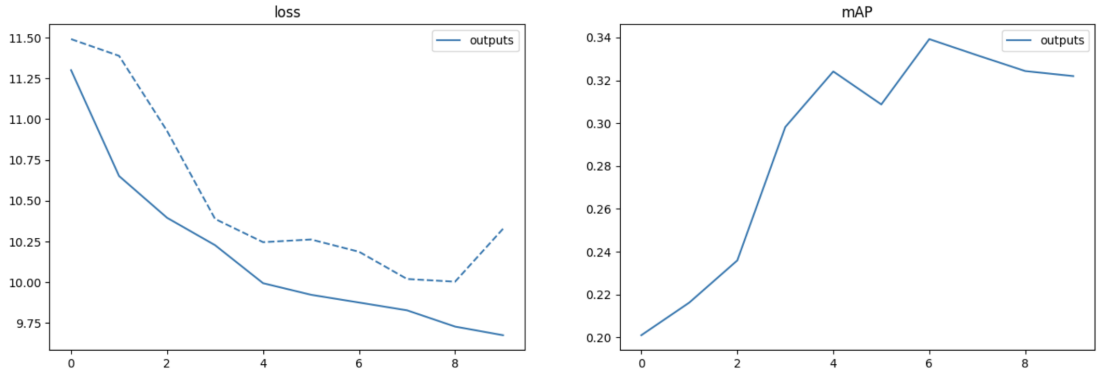

Figure 7: *Run 3:* Training time: 1:51:42. Precision 35.0 at Epoch 6

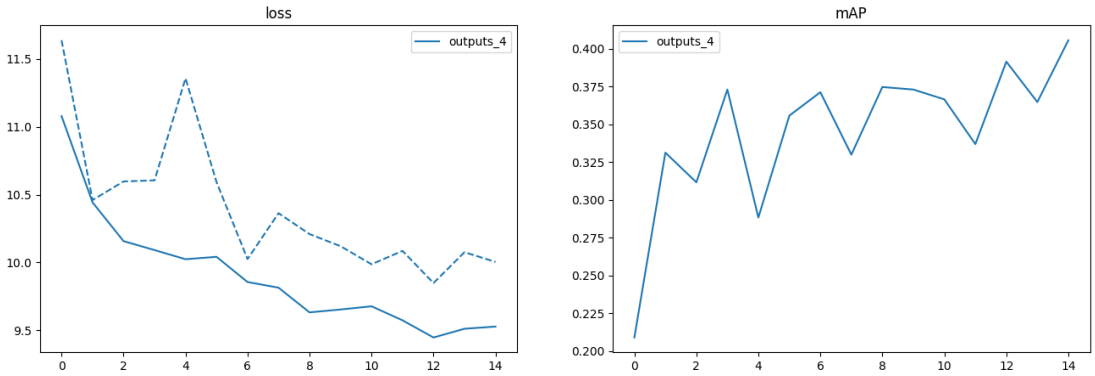

Figure 8: *Run 4:* Training time: 3:04:26. Precision 40.4 at Epoch 14

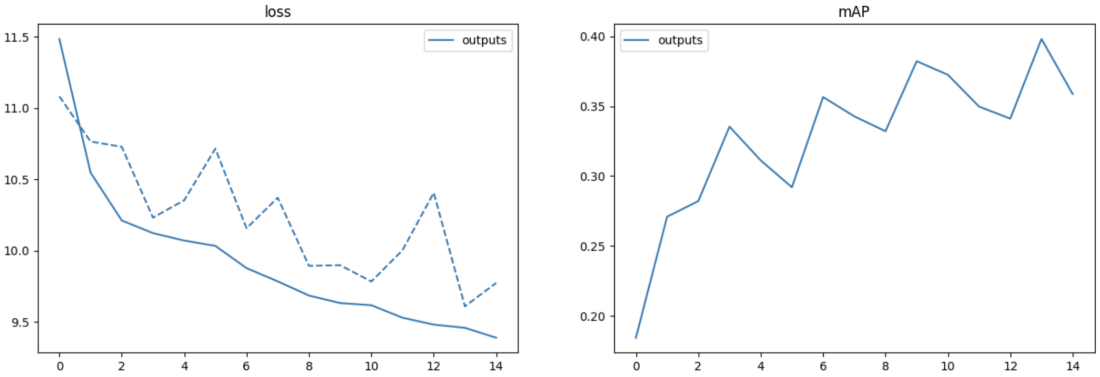

Figure 9: *Run 5:* Training time: 2:32:45. Precision 40.9 at Epoch 13

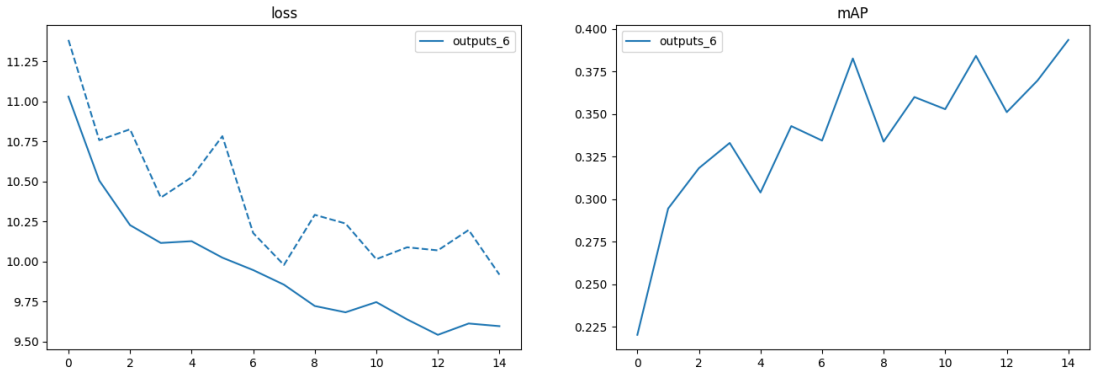

Figure 10: *Run 6:* Training time: 2:54:51. Precision 40.2 at Epoch 14

References

[1] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers, 2020.

[2] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. *CoRR*, abs/1604.01685, 2016.

[3] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross B. Girshick. Mask R-CNN. *CoRR*, abs/1703.06870, 2017.

[4] Tsung-Yi Lin, Michael Maire, Serge J. Belongie, Lubomir D. Bourdev, Ross B. Girshick, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft COCO: common objects in context. *CoRR*, abs/1405.0312, 2014.