DASHBOARD VIDEO SEGMENTATION

Jared Ucherek

The University of Texas at Austin

ABSTRACT

In this paper, we cover the motivation and approach for accurate image segmentation from dashboard video. We specifically focus on performance for the Comma10k dataset, an open source dataset with 10,000 images with five separate segmentation classes [1]. A variety of semantic segmentation architectures would perform well on this dataset, and we discuss their differences. The performance for our model of choice is compared to other public scores cited on the Comma10k repo, including Comma10k baseline [2]. We cover important training and augmentation steps useful for performance gains and ultimately show great performance with contemporary transformer model Trans2Seg [3].

***Index Terms*—** Computer Vision, Image Segmentation, Transformer, Self Driving

1. INTRODUCTION

Self driving has seen an explosion of interest after computer vision models started to show human level performance in a variety of tasks. Private companies aim to tackle the challenge with rich and diverse sensor data constantly surveying the environment surrounding the car. Commaai controls vehicles for self driving through a single camera input mounted on the car dashboard. An accurate machine learning model trained for object detection would provide crucial information to latter parts of the self-driving software stack. For this reason, they curated and released the Comma10k dataset in order to refine their models for this task.

Many useful datasets in the self driving domain have been released for similar reasons. Open-sourced datasets offer great opportunities for collaboration and advancement in the field. Waymo has released the Waymo Open Dataset, containing both motion and perception content for 3D environments [4]. Lyft has a similar public dataset, Level 5 dataset, which has several thousand human-labeled annotated frames and semantic maps for 2D and 3D environments [5]. In the academic setting, UC Berkeley has a popular BDD100k dataset with a massive amount of annotated data with similar classes to the Comma10k dataset [6]. Each of these datasets present unique challenges to the field of perception in the self driving space, with a variety of complex architectures able to perform relatively well on the dynamic tasks.

The models used to accurately detect, segment, or predict actions greatly vary due to the input dimensions of the data and output dimensions requires for the problem. In this paper, we focus on the simple combination of single input frames to single output segmentation. In this domain, inference speeds can be improved to ensure these models run real-time. We proceed to describe the dataset, architectures, and our performance for this particular image segmentation task.

2. DATASET

The dataset contains 10,000 manually segmented images taken from a Commaai dashboard camera. The validation set consists of the 1000 numbered images ending in '9.png'. Thus, validation scores for performance comparison can be easily calculated using Categorical Cross Entropy (CCE). The dataset contains only a few sequential images, as most of the images are separate pictures from driving segments to help protect privacy and diversify the driving environments present. For semantic segmentation, any unique object pertaining to a particular class is given the same color. We display an example with transparent segmentation in figure 1, to give an idea of how the camera is mounted and which objects are segmented into each class.

Fig. 1. A sample image showing transparent semantic segmentation overlayed on the original image. Each frame is originally 654 pixels tall by 1164 pixels wide.

There are five individual classes described within the dataset: movable, my car, lane markings, road, and undriv-

able. Movable objects include other vehicles, people, and animals. My car includes anything inside the vehicle with the dashboard camera, including wires, mounts, etc. The lane markings exclude other commonly painted markings on the road such as turn arrows and crosswalks. The road is essentially anywhere a car can legally drive. Lastly, undrivable is the background class, which any pixel is labeled if it belongs to none of the other classes. As these images are standard size taken with equivalent cameras, cropping the images to remove certain sections should be avoided to robustly compare performance between models.

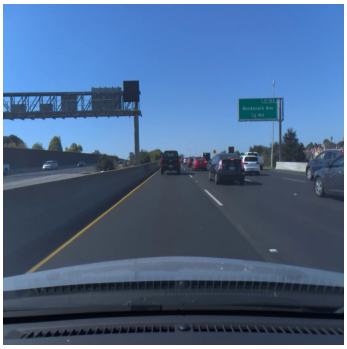

Fig. 2. A reshaped 512x512 validation image of the dataset. The images most readily vary by the time of day, the type of car being driven, the current weather, and the type of street being driven on. The ground truth and prediction images are compared below in figure 8.

While a typical driving segment would be time series, the images in this dataset are shuffled and contain images from a wide variety of segments. Therefore, there is minimal leakage between the training and validation images. A more advanced segmentation task for a sequence of images would need to account for this split for proper evaluation. We discuss this sequential task in the conclusion in addition to more sophisticated tasks, where the time series nature could be used to exploit faster inference or smooth predictions between frames.

3. TRAINING DETAILS

Before covering the architecture and results, we include several additional aspects that can influence the final performance of our model. These details are important to consider when comparing the performance with the two other models below. We trained our model using PyTorch 1.6.0 and Cuda 10.2 with four GeForce RTX 2090 Ti GPUs, each with 11 GB of VRam. Adam optimizer was used with an initial learning rate of 5e-5 [7]. With a batch size of 4 images per GPU and 9000 training images, about 500 iterations were run per epoch at 1.15 iterations per second. This resulted in about 10 hours of training overnight from a randomized initialization of weights. The convolutional backbone used was EfficientNet-b0, with pretrained weights from ImageNet [8] [9].

We used the Albumentations library to apply extensive image augmentations to training images [10]. 16 different transforms were used from Albumentations, which generally improve model performance and generalization on these learning tasks [11]. Specifically, augmentations that affect noise, blur, brightness, or contrast are important to apply randomly to the training data. These changes can appear naturally on the images in the dataset during the variety of driving conditions that take place. Examples of each of the transforms are shown in the appendix.

4. SEGMENTATION ARCHITECTURE

Segmentation requires attributing individual pixels from the input image to target classes. The masks for each class that represent objects within the image may overlap, but this does not apply in the Comma10k dataset. As discussed in section 2, the output size from the model should loosely match the output size for accurate segmentation masking. Thus, most architectures contain contraction to a latent space then expansion to the relative input size. This paradigm evolved from the tendency of classification models to contract input features into low-dimensions before classification. Instead of classification, these features are progressively expanded back to the original input size to predict the final output mask.

Popular models that utilize this general procedure include FCN, U-Net and Mask R-CNN [12] [13] [14]. For reference, simplified versions of each of these architectures are described in figures 3, 4, and 5. U-Net is used as the segmentation model in the Comma10k baseline for excellent performance results [2]. These convolutional networks differ from our architecture of choice, Trans2Seg, that utlizes the attention mechanism to encode and decode features for segmentation.

Many architectures within the computer vision domain have shifted toward the attention mechanism, replacing either convolutions or feature extraction operations [16]. For this reason, we choose to apply Trans2Seg to our segmentation task [17]. Trans2Seg was originally developed for use on a novel transparent segmentation dataset, and offers a coupled CNN-Transformer architecture for segmentation. This architecture utilizes multiple layers of dot product attention to encode input features, and decode the embedding into an attention map. As shown in figure 6, each of the M attention heads from the decoder outputs an attention map that loosely represents coarse logit predictions of the N classes. The coarse predictions are combined with an early layer of the CNN backbone before a fully connected layer outputs the final logits in the shape of the original image. The details of these final steps are illustrated in figure 7.

U-Net, Mask R-CNN, and Trans2Seg all utilize a convolutional backbone for feature extraction from the input im-

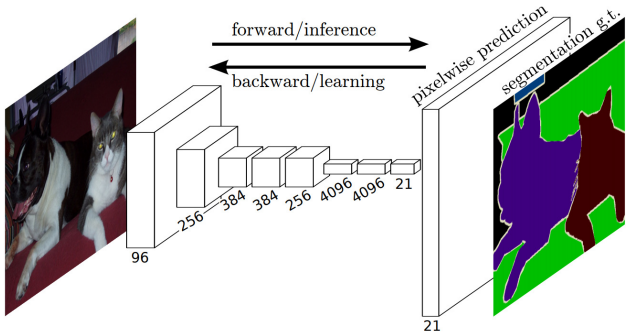

Fig. 3. FCN, or fully convolutional networks, translated successful architectures in the classification space to segmentation [12]. Just as with other traditional deep learning architectures, FCN extracts features from the input image into a low dimensionsal embedding using convolutional layers. Instead of flattening these embeddings into a fully connected layer for classification, they are upsampled with deconvolution, and a final layer of 21 1x1 kernels outputs the finalized prediction.

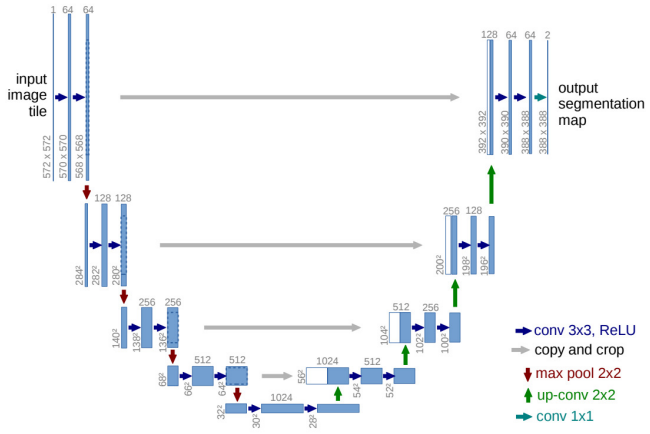

Fig. 4. U-net example architecture [13]. Many variants exist with changes made to input/ouput dimensions, embedding dimensions, and sizes of the intermediate operations. Each blue box corresponds to a multi-channel feature map. The number of channels is denoted on top of the box. The x-y-size is provided at the lower left edge of the box. White boxes represent copied feature maps.

age before further manipulation into a latent space. Most of the original architectures favor a classical pretrained ResNet architecture for this phase [18]. However, the baseline implementation opts to use a contemporary convolutional network shown to outperform most convolutional networks on image classification datasets, EfficientNet [8]. We opt to utilize this backbone for our architecture as well, as it behaves similarly to ResNet, downsizing the dimensions by 2 as you move down each layer.

Most importantly, these transformer models may offer

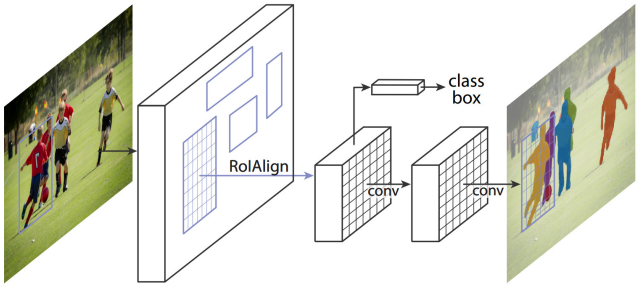

Fig. 5. Mask R-CNN extends the capabilities of its object detection counterpart, Faster R-CNN [14] [15]. This is done by adding a small FCN branch to each Region of Interest (RoI). These RoIs introduce several parameters and subtle design choices that play a crucial role for overall model performance. Nonetheless, this architecture intuitively reworks a traditional bounding box detection model to simultaneously output segmentation masks.

great improvements for time series based vision tasks, as the transformer models emerged from NLP tasks requiring sequential inputs to the networks [19]. Although some architectures have emerged for contextual video understanding [20], there will need to be much more development for datasets and pretrained models before widespread adoption of transformers are used for most video related tasks.

5. RESULTS

After implementing the architecture with generalizable code and training with a variety of augmentations overnight, the performance of the model is comparable to two baseline validation scores shown on the Comma10k repository page [1]. The ground truth and predicted segmentation masks from figure 2 are shown in figure 8. We believe this shows promise for future transformer architectures within the space, but more work will be needed to improve their performance and generalize the architecture across a variety of tasks.

Method	Val CCE
Commaai Proprietary Model	0.051
Comma10k Baseline	**0.045**
Trans2Seg on Comma10k	0.075

Table 1. Comparison of the reported CCE loss on the 1000 validation images of the dataset. Our model, Trans2Seg on Comma10k, shows promise when compared to the state-of-the-art baselines. These models all report over 99% accuracy and segment most images near human-level.

An important qualitative metric involves the real-time capabilities for each model. Recent work has developed a framework for comparing semantic segmentation mod-

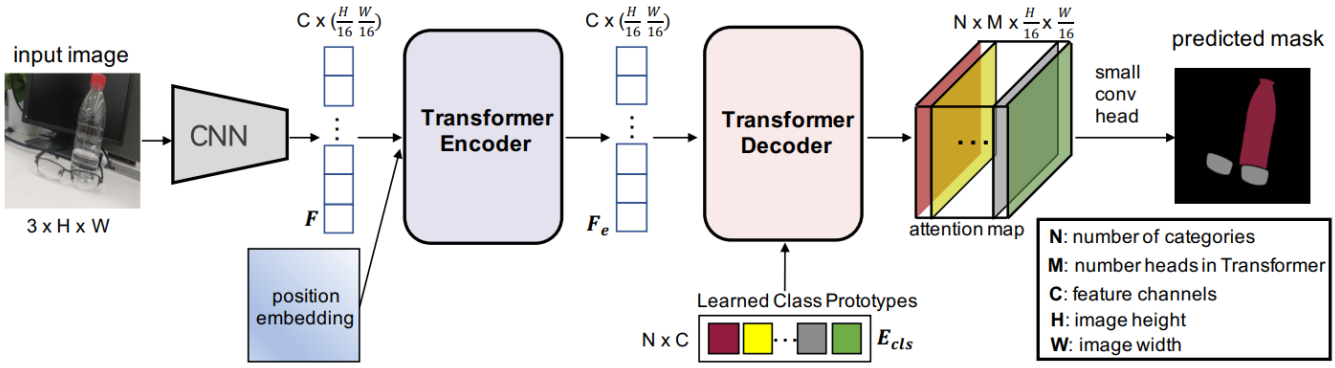

Fig. 6. The original pipeline of the Trans2Seg hybrid CNN-Transformer architecture. The image is reshaped to 512x512, and fed into the CNN to extract features. These features pass through a transformer encoder and decoder with learned class prototypes. The decoder outputs a coarse-grain prediction of the segmentation mask for each class. The predicted segmentation results from a pixel-wise argmax on the output. More details of transformer decoder and small conv head are shown in figure 7.

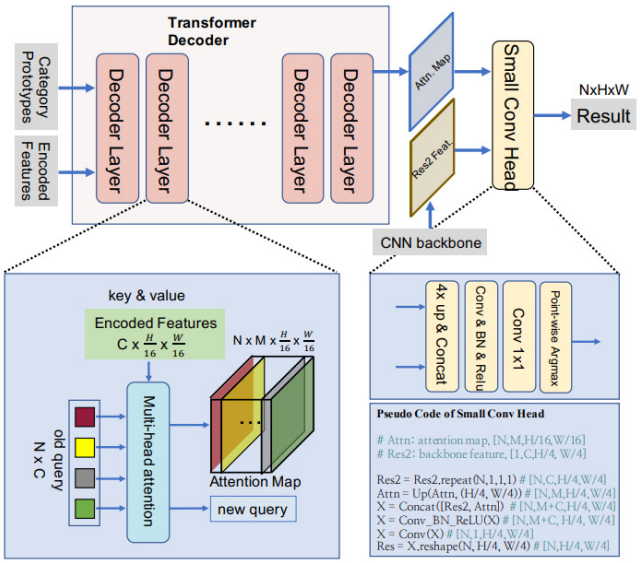

Fig. 7. The detailed view of the decoder and small conv head from figure 6. The query into the decoder are the learnable category prototypes, and the key/value pairs are the encoded features. Multi-layered attention within the decoder outputs the attention map, which is combined with the Res2 layer to output the predicted mask.

els in relation to their performance on self driving datasets [21]. Without implementing the specific models used on the Comma10k dataset into their framework, comparison of the GFLOPs for each respective model must suffice. Originally, U-Net with ResNet18 is reported to have 43.9 GFLOPs [21] on 512x1024 input images. A table in [17] shows Trans2Seg-medium with ResNet50c to have 49.0 GFLOPs on 512x512 input images, comparable in speed to many other segmentation models. The operations can be reduced in the transformer

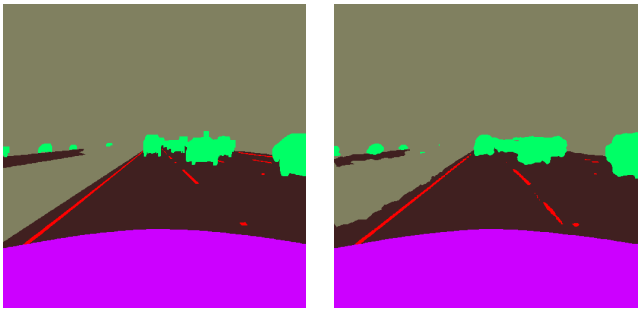

Fig. 8. Ground truth (left) and prediction (right) for image 4861 of the Comma10k dataset. The model accurately segments the objects into the appropriate classes, but fails to finely distinguish the edges of objects. The original image is shown above in figure 2

architecture either by reducing the size of the backbone CNN, or by employing attention mechanisms with reduced complexity [22] [23]. Ultimately, this suggests that this model can be readily modified to operate in real-time settings without major trade-offs to its performance.

6. CONCLUSION

Our work details the architecture and performance of two models for segmenting dashboard images. U-Net and Trans2Seg both exhibit great performance on the straightforward task, with promise for potential adoption by Commaai if they choose to update or change their proprietary model for any reason. The open-sourced Comma10k dataset will surely be helpful to use for segmentation benchmarks in both the academic and private setting. We expect to see the newer transformer models to be adopted for extensive use in this space.

As always, important consideration should be given to the complexity trade-offs with these newer models. When approaching unsolved problems with deep learning, we should be aware of what modifications are being made to differentiate between machine learning innovation and standard software development. This mindfulness for generalization is important, as an increasing number of institutions and companies are relying on engineers and data scientists to automate challenging tasks with machine learning. Many of these tasks will require new data pipelines or output targets, but might be solvable with the models considered outdated by research standards.

Sequential data from video will continue to gain interest, and contains important relevance in the self driving domain that will in turn help machine learning improve across the field. Many simple tasks involving sequential data or forecasting are extremely difficult to approach due to a lack of datasets or pretrained models to use. An interesting multimodal learning project might involve combining our speed detection work from last semester with this segmentation task in a single end-to-end model. This includes multi-input-multi-output for video tasks, a common situation for the massive amount of aggregated data we have today.

7. REFERENCES

[1] commaai, "comma10k," *GitHub repository*, 2020.

[2] Yassine Yousfi, "comma10k-baseline," *GitHub repository*, 2018.

[3] Enze Xie, "Trans2seg," *GitHub repository*, 2018.

[4] Pei Sun, Henrik Kretzschmar, Xerxes Dotiwalla, Aurelien Chouard, Vijaysai Patnaik, Paul Tsui, James Guo, Yin Zhou, Yuning Chai, Benjamin Caine, et al., "Scalability in perception for autonomous driving: Waymo open dataset," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 2446–2454.

[5] John Houston, Guido Zuidhof, Luca Bergamini, Yawei Ye, Ashesh Jain, Sammy Omari, Vladimir Iglovikov, and Peter Ondruska, "One thousand and one hours: Self-driving motion prediction dataset," 2020.

[6] Fisher Yu, Haofeng Chen, Xin Wang, Wenqi Xian, Yingying Chen, Fangchen Liu, Vashisht Madhavan, and Trevor Darrell, "Bdd100k: A diverse driving dataset for heterogeneous multitask learning," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[7] Diederik P. Kingma and Jimmy Ba, "Adam: A method for stochastic optimization," in *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, Yoshua Bengio and Yann LeCun, Eds., 2015.

[8] Mingxing Tan and Quoc Le, "Efficientnet: Rethinking model scaling for convolutional neural networks," in *International Conference on Machine Learning*. PMLR, 2019, pp. 6105–6114.

[9] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE Conference on Computer Vision and Pattern Recognition*, 2009, pp. 248–255.

[10] Alexander Buslaev, Vladimir I. Iglovikov, Eugene Khvedchenya, Alex Parinov, Mikhail Druzhinin, and Alexandr A. Kalinin, "Albumentations: Fast and flexible image augmentations," *Information*, vol. 11, no. 2, 2020.

[11] Tong He, Zhi Zhang, Hang Zhang, Zhongyue Zhang, Junyuan Xie, and Mu Li, "Bag of tricks for image classification with convolutional neural networks," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 558–567.

[12] Jonathan Long, Evan Shelhamer, and Trevor Darrell, "Fully convolutional networks for semantic segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 3431–3440.

[13] Olaf Ronneberger, Philipp Fischer, and Thomas Brox, "U-net: Convolutional networks for biomedical image segmentation," in *International Conference on Medical image computing and computer-assisted intervention*. Springer, 2015, pp. 234–241.

[14] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick, "Mask r-cnn," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2961–2969.

[15] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun, "Faster r-cnn: towards real-time object detection with region proposal networks," *IEEE transactions on pattern analysis and machine intelligence*, vol. 39, no. 6, pp. 1137–1149, 2016.

[16] Jean-Baptiste Cordonnier, Andreas Loukas, and Martin Jaggi, "On the relationship between self-attention and convolutional layers," in *International Conference on Learning Representations*, 2020.

[17] Enze Xie, Wenjia Wang, Wenhai Wang, Peize Sun, Hang Xu, Ding Liang, and Ping Luo, "Segmenting transparent object in the wild with transformer," *arXiv preprint arXiv:2101.08461*, 2021.

[18] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[19] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, Minneapolis, Minnesota, June 2019, pp. 4171–4186, Association for Computational Linguistics.

[20] Rohit Girdhar, João Carreira, Carl Doersch, and Andrew Zisserman, "Video Action Transformer Network," in *CVPR*, 2019.

[21] Mennatullah Siam, Mostafa Gamal, Moemen Abdel-Razek, Senthil Yogamani, Martin Jagersand, and Hong Zhang, "A comparative study of real-time semantic segmentation for autonomous driving," in *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, 2018, pp. 587–597.

[22] Nikita Kitaev, Lukasz Kaiser, and Anselm Levskaya, "Reformer: The efficient transformer," *arXiv preprint arXiv:2001.04451*, 2020.

[23] Sinong Wang, Belinda Li, Madian Khabsa, Han Fang, and Hao Ma, "Linformer: Self-attention with linear complexity," *arXiv preprint arXiv:2006.04768*, 2020.

[24] Pavel Yakubovskiy, "Segmentation models pytorch," *GitHub repository*, 2020.

[25] WA Falcon and et al., "Pytorch lightning," *GitHub repository*, 2019.

8. APPENDIX

The code can be found at this repo: `https://github.com/jareducherek/drive_segmentation`. We utilize the training code structure from the comma10k baseline [2], which required implementing our architecture of choice in PyTorch Segmentation models [24], enabling us to easily use the EfficientNet backbone with PyTorch-Lightning [8] [25]. Our contributions include this architecture implementation, and updating the original baseline to a new version of PyTorch Lightning.

Below are the 16 transforms applied to the training data discussed in section 3. The original image is shown in figure 2.

Fig. 9. Horizontal Flip and Gaussian Noise.

Fig. 10. Grid distortion and elastic transform.

Fig. 11. Shift/scale/rotate and optical distortion.

Fig. 12. Contrast Limited Adaptive Histogram Equalization (CLAHE) and random brightness.

Fig. 13. Ground truth and horizontal flip.

Fig. 14. Sharpen and random contrast.

Fig. 15. Blur and motion blur.

Fig. 16. hue saturation and cutout.