Image Segmentation with Transformers

February 2021
Marius Schmidt-Mengin
École des Ponts ParisTech
marius.schmidt.mengin@gmail.com

Abstract

It has recently been shown that Transformers can benefit computer vision tasks such as image classification and object detection. In this project, I combine innovations from Vision Transformers (ViT) [7] and Detection Transformers (DETR) [2]. I first design a simple patch-based semantic segmentation Tranformer and compare it to U-Net on Cityscapes. I then extend this concept to instance segmentation, using object queries as in DETR. Finally, I apply the bipartite matching scheme of DETR to train U-Net for instance segmentation.

1. Introduction

Semantic, Instance and Panoptic segmentation The goal of semantic segmentation is to segment an image into different regions (like sky, floor, ...), without explicitly distinguishing between object instance. On the other hand, instance segmentation is able to separates distinct object instances. Panoptic segmentation combines both: distinct object instances ("things") as well as undistinguishable ones ("stuff") are detected and segmented.

Segmentation with Deep Learning Nowadays, most (if not all) segmentation methods are base on convolutional neural networks. They usually consist of an encoder, which aggregates spatial information and increases the receptive field, and a decoder, which generates high definition segmentation maps. U-Net [16] is one of the most widely used CNNs for semantic segmentation. U-Net uses skip connections which connect intermediate stages of the encoder to the decoder. They help maintain spatial information along the depth of the network. Instance and semantic segmentation are more complicated. They are often extensions of object detection models and can roughly be classified into two-stage and one-stage methods. A notable two stage method is Mask-RCNN [8], which extends Faster-RCNN [15] by adding a mask branch.

Transformers in Computer Vision Recently, Transformers [18] have shown to be promising for different object recognition tasks. The Image Transformer [14] and Image GPT [3] adapt Transformers to the task of image generation and super-resolution. The Detection Transformer (DETR) [2] feeds the flattened feature maps of a standard convolutional backbone through a Transformer and generates a set of object classes and bounding boxes, achieving competitive performance on the COCO and Pascal VOC object detection datasets. DETR can also be extended for panoptic segmentation. Vision Transformers [7] reshape an image into a sequence of flattened patches and feed these to a BERT-like Transformer for classification. They are currently state-of-the-art on ImageNet.

Scope of the Project In this project, I combine elements from Vision Transformers, DETR, and U-Net [16]. First, I build a Vision Transformer inspired semantic segmentation neural network. Then, I extend this model to instance segmentation by adding object queries and a Hungarian loss as in DETR [2]. Last, I take the object queries and Hungarian loss and apply them to U-Net to yield a simple panoptic segmentation network.

2. Semantic Segmentation with Transformers

2.1. Principle

This part was the main goal of my project proposal. I start from Vision Transformers [7], which are image classification models that see an image as a sequence. The input image, of shape $W \times W$ (for the sake of simplcity, we assume it to be square) is first divided into an $S \times S$ grid of square patches of $s \times s$ pixels (so that $W = s \cdot S$). This grid is then flattened into a sequence of length $N = S \times S$. Each patch is flattened into an s^2-dimensional vector and projected linearly to a vector of dimension d_{model}. The resulting sequence of vectors is input to a Transformer encoder, whose hidden dimension is d_{model}. In the sequel, positional encodings are always added at the input of the Transformer. To generate the segmentation maps, I collect the hidden

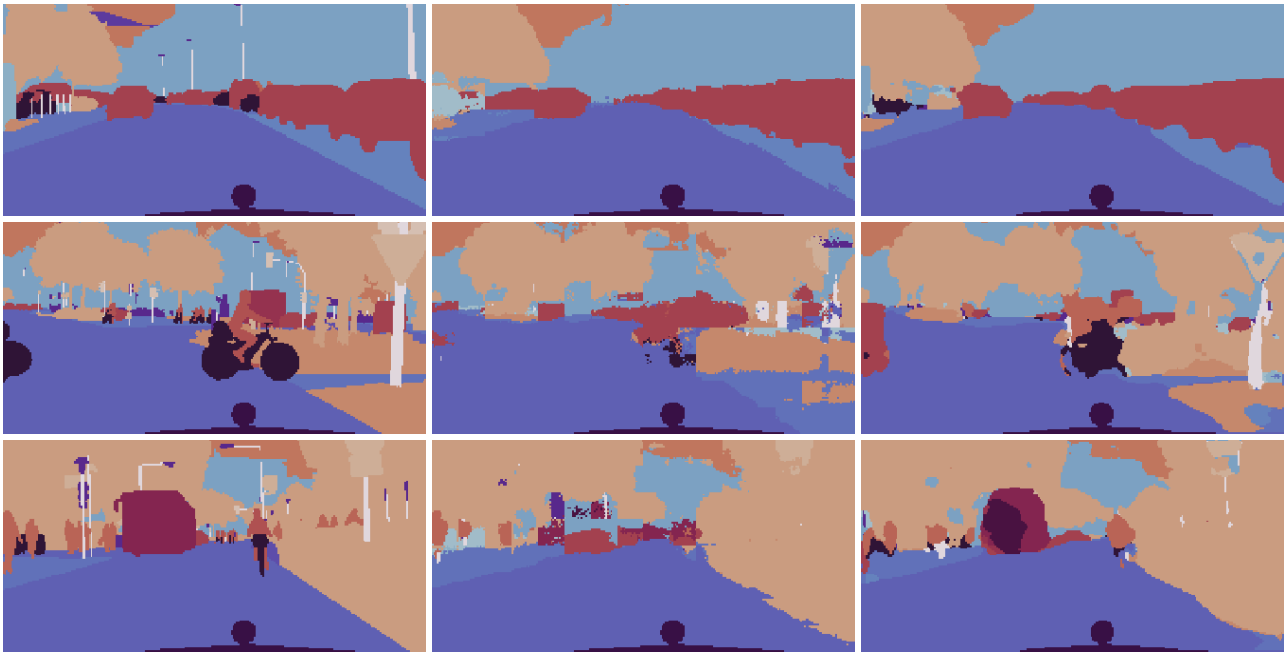

Figure 1. Semantic segmentation on Cityscapes [5]. Left: ground truth. Middle: Seq2Seg. Right: U-Net [16]

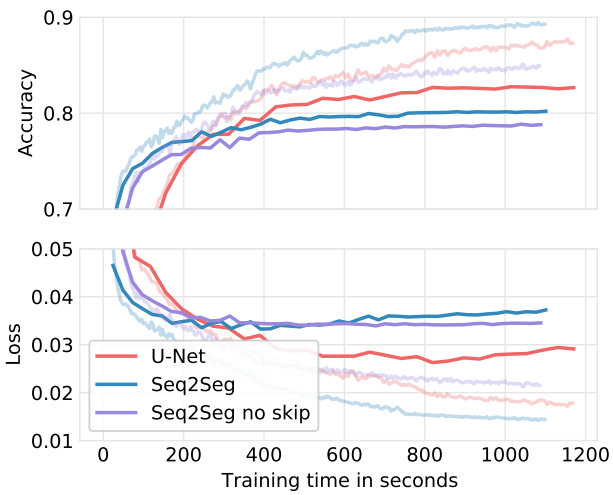

Figure 2. Semantic segmentation on Cityscapes [5]. The faded lines represent training, and the others validation.

state vectors from all layers of the Transformer encoder and sum them, creating direct skip connections to the output. The sequence of these N summed vectors is reshaped into a tensor of shape $d_{\text{model}} \times S \times S$. Finally, a transposed convolution maps it to a tensor of shape $N_{\text{class}} \times W \times W$, which contains the predicted segmentation maps. I call this network Seq2Seg.

A paper ("Rethinking Semantic Segmentation from a Sequence-to-Sequence Perspective with Transformers") [21] was published in December 2020, where the authors explore this idea, with two minor differences: (1) they further add several convolutional layers to generate the segmentation maps from the Transformer's outputs and (2) they do not use hidden states from intermediate layers. They show that this architecture produces state-of-the-art results on ADE20K [22] and Pascal Context [13]. As one of my goals is to explore what can be done with a very limited number of convolutions, I do not use a multi-layer convolutional network to generate the final segmentation maps from the encoder outputs.

2.2. Implementation details

I use HuggingFace's [20] implementation of BERT [6] for the Transformer encoder. I use 6 layers with a hidden size of 768 and feedforward size of 3072. I reshape the Cityscape images and masks to a size of 140×280 and extract a random (center during validation) crop of shape 128×256. I use a patch size of 16, which results in a sequence of $8 \times 16 = 128$ patches. I optimize all networks with binary cross entropy loss, Adam optimizer and a batch size of 24.

For U-Net, I use an initial learning rate of 10^{-3}, which is reduced to $5 \cdot 10^{-4}$ and 10^{-4} after 10 and 20 epochs. In total, U-Net is trained for 30 epochs of 123 mini batches.

I observe that a training step with Seq2Seg is faster (0.2 vs 0.3 seconds for U-Net, with batch size 24). I therefore train it for 45 epochs in total, so that the total training time is similar to U-Net. I use an initial learning rate of 10^{-4} and

reduce it to $5 \cdot 10^{-5}$ and 10^{-5} after 15 and 30 epochs.

I also try to remove the direct skip connection to the output by applying the ultimate transposed convolution to the last hidden state of the encoder directly. This variant fails to train with a learning rate of 10^{-4}. I therefore start with a learning rate of $5 \cdot 10^{-5}$ and reduce it to 10^{-5} and $5 \cdot 10^{-6}$ after 15 and 30 epochs.

2.3. Results

Figure 2 shows the evolution of loss and accuracy during training. Seq2Seg obtains only 3 percent less in accuracy than U-Net, which seems good because unlike U-Net, Transformers were not designed for image segmentation. Table 1 shows the benefit of using a skip connection that connects all hidden states to the output. Qualitative results are presented on Figure 1. The segmentation maps produced by U-Net look much more natural than those of Seq2Seg. This can probably be explained by the fact that convolutional neural network are naturally good at generating image [17].

	U-Net	Seq2Seg	Seq2Seg no skip
Parameters	7.8M	50.8M	50.8M
Training Step	$\approx$ 0.3s	$\approx$ 0.2s	$\approx$ 0.2s
Train accuracy	**90.76**	89.65	85.32
Val accuracy	**82.75**	80.20	78.85

Table 1. Seq2Seg versus U-Net. The duration of a training step was measured with a batch size of 24.

2.4. Scaling it up

One substancial problem with Transformers is that the runtime and memory cost of the attention layers are quadratic in the sequence length. In Seq2Seg, if the patch size remains fixed, the number of patches grows quadratically with the image size, and so the ressources needed are quartic in the image size (meaning that doubling the image size multiplies the computational cost by 8). One possibility would be to increase the patch size. This, though, results in a big loss of information in the first linear projection layer (one might as well downsample the image), unless the hidden size of the transformer is increased proportionally to the number of pixels in a patch. But this has again a quadratic cost.

Another option would be to use evolutions of the Transformer, such as the Longformer [1], the Reformer [10] or the Performer [4]. These models modify the attention layers so as to reduce their computational complexity and memory footprint. I tried to use a Reformer and a Performer, but they did not yield satisfactory results. Implementations of these models are recent and still experimental.

3. Instance Segmentation

3.1. DETR [2]

DETR uses an encoder-decoder Transformer for object detection and instance segmentation. First, it extracts features with a backbone such as ResNet [9], resulting in a tensor of shape $d \times h \times w$. This tensor is flattened to a sequence of feature vectors of shape $d \times hw$ and input to a Transformer encoder. The input of the Transformer decoder is a sequence of $N_{\text{obj}} = 100$ object queries, which are just learnable embeddings. These object queries are decoded in parallel (i.e. without causal masking, which differs from most language models, where decoding is done autoregressively). Each of these object queries produces an output embedding, which is fed to a small MLP to produce the class logits and bounding box prediction. The generation of instance masks is more complicated and I do not explain it here. To match targets and predictions during training, DETR uses a novel bipartite matching scheme, which does not use any handcrafted rules. First, a cost is computed between each prediction and target. This cost can be the loss, for example, and measures how similar a prediction is to a target. Then, targets are assigned to predictions in an optimal way, using the Hungarian algorithm. Object queries that are not matched to any target are assigned to an additional class, "no object". The final loss that is computed using these assignments is called Hungarian loss.

3.2. Seq2Seg for instance segmentation

Inspired by this, I extend Seq2Seg to produce instance masks. Contrary to DETR, my model only consists of a Transformer encoder. I extend Seq2Seg by prepending the sequence of flattened patches with N_{obj} learnable embeddings (the object queries), before feeding the whole sequence to the encoder. This is comparable to the [CLS] token of BERT [6], whose output is used to classify a sentence. I then append a linear layer to the last hidden state of the object queries to predict the class and the mask. As in DETR, I use the Hungarian algorithm in order to match prediction and targets during training.

3.3. Implementation details

To train and test my model, I design a toy instance segmentation dataset which allows me to experiment in a low resource regime. I generate images of size 64×64 which contain between 1 and 4 digits of MNIST randomly scattered across the image. I make one experiment where the digits are scattered so that they are non-overlapping (easy) and one without any contraint (hard).

I implement the Transformer encoder using HuggingFace's [20] implementation of BERT [6]. I use 4 hidden BERT layers, a hidden size of 512, a FFN size of 2048 and 8 self-attention heads. I use $N_{\text{obj}} = 6$ object queries. I chose

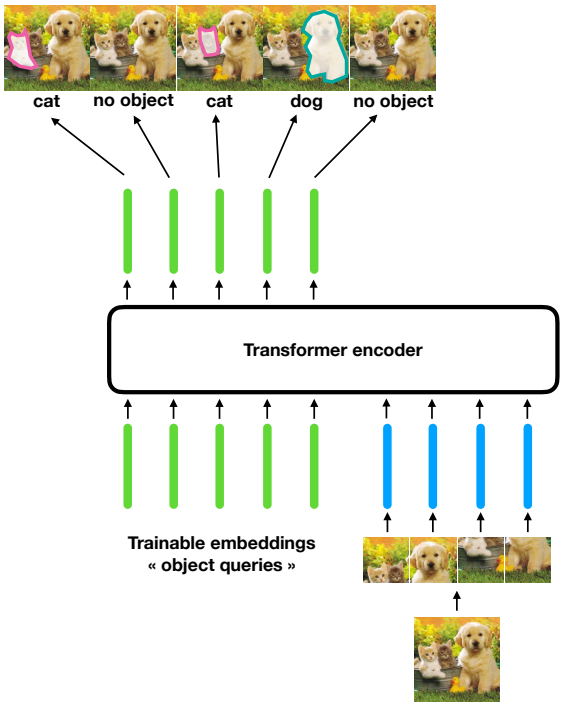

Figure 3. Overview of the proposed instance segmentation network. The input image is input to a Transformer encoder as a sequence of flattened patches, together with object queries. The object queries generate the final instance masks and class predictions.

Dice loss [12] for the mask loss, cross-entropy for the class loss. Chosing Dice loss over binary cross-entropy for masks resulted in a big improvement. I observed that it is necessary to reduce the relative weight of the class loss in order to achieve good results. I therefore set the weight of the class loss to 0.1 compared to 1 for the mask loss. Furthermore, as the "no object" class is predominant, I set its weight to 0.5 (compared to 0.1 in DETR [2]). I train for 11 256 steps with a batch size of 64, using Adam with learning rate 10^{-4}.

3.4. Results

Qualitative results for the hard and easy datasets are shown on Figures 4 and 5 respectively.

Hard images Multiple overlapping digits make the recognition and disentanglement of instances difficult. The network reaches 62% accuracy on digit classification and a mask dice score of 0.2.

Easy images In this case, the task is easily solved by the netword. Classification and mask reconstruction are almost perfect in most cases. The network reaches 97% accuracy on digit classification and a mask dice score of 0.03.

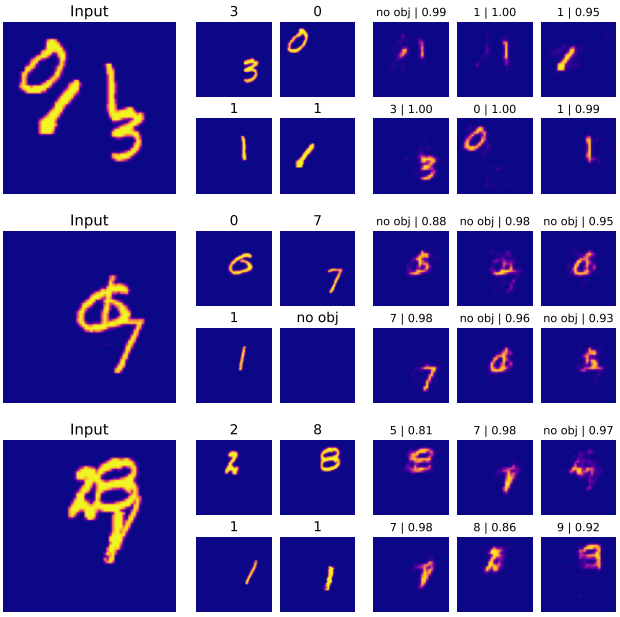

Figure 4. Examples of detections on hard images. The first column is the input, the second and third are the ground truth masks and the rest are the output from the 6 object queries together with the predicted class and confidence.

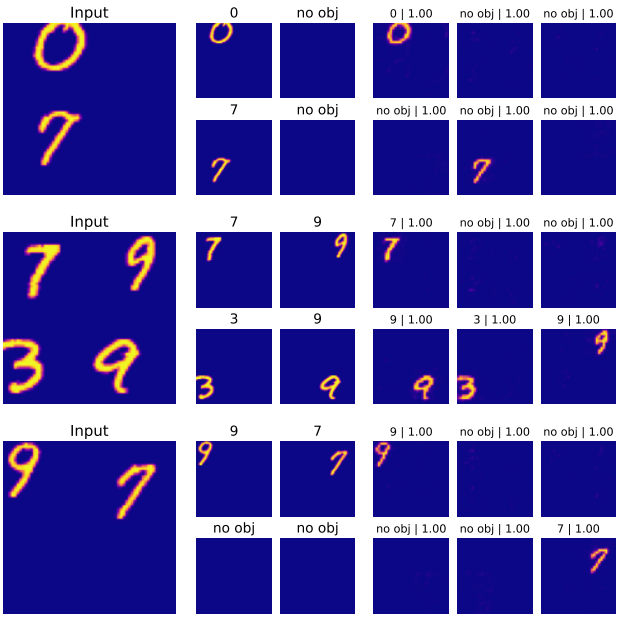

Figure 5. Examples of detections on easy images. The first column is the input, the second and third are the ground truth masks and the rest are the output from the 6 object queries together with the predicted class and confidence.

3.5. Possible extensions

The instance masks of Seq2Seg have a low resolution compared to the input image. This stems from the fact that each instance is only predicted using one hidden state (the hidden state of the corresponding object query). Two possible solutions to this problem are:

- In DETR [2], segmentation masks are predicted by first computing attention heatmaps between the hidden states of each object query and the hidden states of the encoder (and then, the outputs of its convolutional backbone are mixed in using several convolutional and upsampling layers). In the case of Seq2Seg, one could try to compute the attention heatmaps between the hidden states of each object query and the hidden states of all images patches.
- One could also try to predict the instance masks in a coordinate system around the target instance. One would predict the bounding box of the object, and the mask inside of this bounding box only, which would increase the spatial resolution of the mask, especially for small objects. For reference, the masks produced by DETR have a stride of 4 (meaning that they are 4 times smaller than the image).

4. U-Net for Instance Segmentation

In this section, I extend a U-Net [16] for instance segmentation using the Hungarian loss [2]. For semantic segmentation, the output of U-Net is a tensor that has N_{class} channels of the same spatial size as the input image. I instead output N_{obj} channels, one for each object query. To predict the class, I append a linear layer to the output of the encoder. During training, each of the outputs (class and mask) is matched to a ground truth using the Hungarian algorithm. Computing the matching cost requires computing the loss between each ground truth/objecy query pair. Using the complete masks results in a high memory footprint. I thus compute the matching cost by first downsampling the predicted and target masks to a resolution of 32×64.

I train this model for 30 epochs on the Cityscapes [5] dataset (fine annotations). I use Dice loss [12] for the mask loss, cross-entropy for the class loss, Adam with a learning rate of 10^{-3} decayed to $5 \cdot 10^{-4}$ and 10^{-4} after 10 and 20 epochs. I set $N_{\text{obj}} = 100$. I do not have time to evaluate the network precisely using panoptic segmentation metrics, and therefore only report qualitative results.

4.1. Results

Although the results are far from perfect, they show that U-Net is able to separate different instances of cars (Figure 6). But there are also lots of redundant detections. I believe that this issue could be adressed by further fine tuning the relative scale of the classification loss compared to the mask loss, or using focal loss [11]. The full output mask of each of the 100 object queries is given in the appendix for several input images.

Figure 6. A U-Net trained for instance segmentation with a Hungarian loss is able to separate different instances.

5. Conclusion

This project confirms that Transformers are effective for segmentation tasks, although the quadratic cost of attention remains a burden for the developpment of these models in image recognition. Furthermore, the semantic segmentation network that I implemented underperforms a simple convolutional neural networks. There have been several works to unify attention with convolutional neural networks, such as Non-Local Neural Networks [19].

The Hungarian matching scheme has proved effective at training various models, removing the need for handcrafted priors. I would like to experiment further with this "Hungarian U-Net". Notably, it could be interesting to see if adding a few BERT layers between the encoder and the decoder could improve performance.

References

[1] Iz Beltagy, Matthew E. Peters, and Arman Cohan. Longformer: The long-document transformer. *CoRR*, abs/2004.05150, 2020. 3

[2] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm, editors, *Computer Vision - ECCV 2020 - 16th European Conference, Glasgow, UK, August 23-28, 2020, Proceedings, Part I*, volume 12346 of *Lecture Notes in Computer Science*, pages 213–229. Springer, 2020. 1, 3, 4, 5

[3] Mark Chen, Alec Radford, Rewon Child, Jeffrey Wu, Heewoo Jun, David Luan, and Ilya Sutskever. Generative pretraining from pixels. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pages 1691–1703. PMLR, 2020. 1

[4] Krzysztof Choromanski, Valerii Likhosherstov, David Dohan, Xingyou Song, Andreea Gane, Tamás Sarlós, Peter Hawkins, Jared Davis, Afroz Mohiuddin, Lukasz Kaiser, David Belanger, Lucy Colwell, and Adrian Weller. Rethinking attention with performers. *CoRR*, abs/2009.14794, 2020. 3

[5] 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. 2, 5

[6] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. In Jill Burstein, Christy Doran, and Thamar Solorio, editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics, 2019. 2, 3

[7] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. *CoRR*, abs/2010.11929, 2020. 1

[8] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross B. Girshick. Mask R-CNN. *CoRR*, abs/1703.06870, 2017. 1

[9] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pages 770–778. IEEE Computer Society, 2016. 3

[10] Nikita Kitaev, Lukasz Kaiser, and Anselm Levskaya. Reformer: The efficient transformer. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020. 3

[11] Tsung-Yi Lin, Priya Goyal, Ross B. Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017*, pages 2999–3007. IEEE Computer Society, 2017. 5

[12] Fausto Milletari, Nassir Navab, and Seyed-Ahmad Ahmadi. V-net: Fully convolutional neural networks for volumetric medical image segmentation. In *Fourth International Conference on 3D Vision, 3DV 2016, Stanford, CA, USA, October 25-28, 2016*, pages 565–571. IEEE Computer Society, 2016. 4, 5

[13] Roozbeh Mottaghi, Xianjie Chen, Xiaobai Liu, Nam-Gyu Cho, Seong-Whan Lee, Sanja Fidler, Raquel Urtasun, and Alan Yuille. The role of context for object detection and semantic segmentation in the wild. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014. 2

[14] Niki Parmar, Ashish Vaswani, Jakob Uszkoreit, Lukasz Kaiser, Noam Shazeer, and Alexander Ku. Image transformer. *CoRR*, abs/1802.05751, 2018. 1

[15] Shaoqing Ren, Kaiming He, Ross B. Girshick, and Jian Sun. Faster R-CNN: towards real-time object detection with region proposal networks. In Corinna Cortes, Neil D. Lawrence, Daniel D. Lee, Masashi Sugiyama, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, pages 91–99, 2015. 1

[16] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In Nassir Navab, Joachim Hornegger, William M. Wells III, and Alejandro F. Frangi, editors, *Medical Image Computing and Computer-Assisted Intervention - MICCAI 2015 - 18th International Conference Munich, Germany, October 5 - 9, 2015, Proceedings, Part III*, volume 9351 of *Lecture Notes in Computer Science*, pages 234–241. Springer, 2015. 1, 2, 5

[17] Dmitry Ulyanov, Andrea Vedaldi, and Victor S. Lempitsky. Deep image prior. *Int. J. Comput. Vis.*, 128(7):1867–1888, 2020. 3

[18] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008, 2017. 1

[19] Xiaolong Wang, Ross B. Girshick, Abhinav Gupta, and Kaiming He. Non-local neural networks. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pages 7794–7803. IEEE Computer Society, 2018. 5

[20] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama

Drame, Quentin Lhoest, and Alexander M. Rush. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online, Oct. 2020. Association for Computational Linguistics. 2, 3

[21] Sixiao Zheng, Jiachen Lu, Hengshuang Zhao, Xiatian Zhu, Zekun Luo, Yabiao Wang, Yanwei Fu, Jianfeng Feng, Tao Xiang, Philip H. S. Torr, and Li Zhang. Rethinking semantic segmentation from a sequence-to-sequence perspective with transformers. *CoRR*, abs/2012.15840, 2020. 2

[22] Bolei Zhou, Hang Zhao, Xavier Puig, Sanja Fidler, Adela Barriuso, and Antonio Torralba. Scene parsing through ADE20K dataset. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pages 5122–5130. IEEE Computer Society, 2017. 2

Appendices

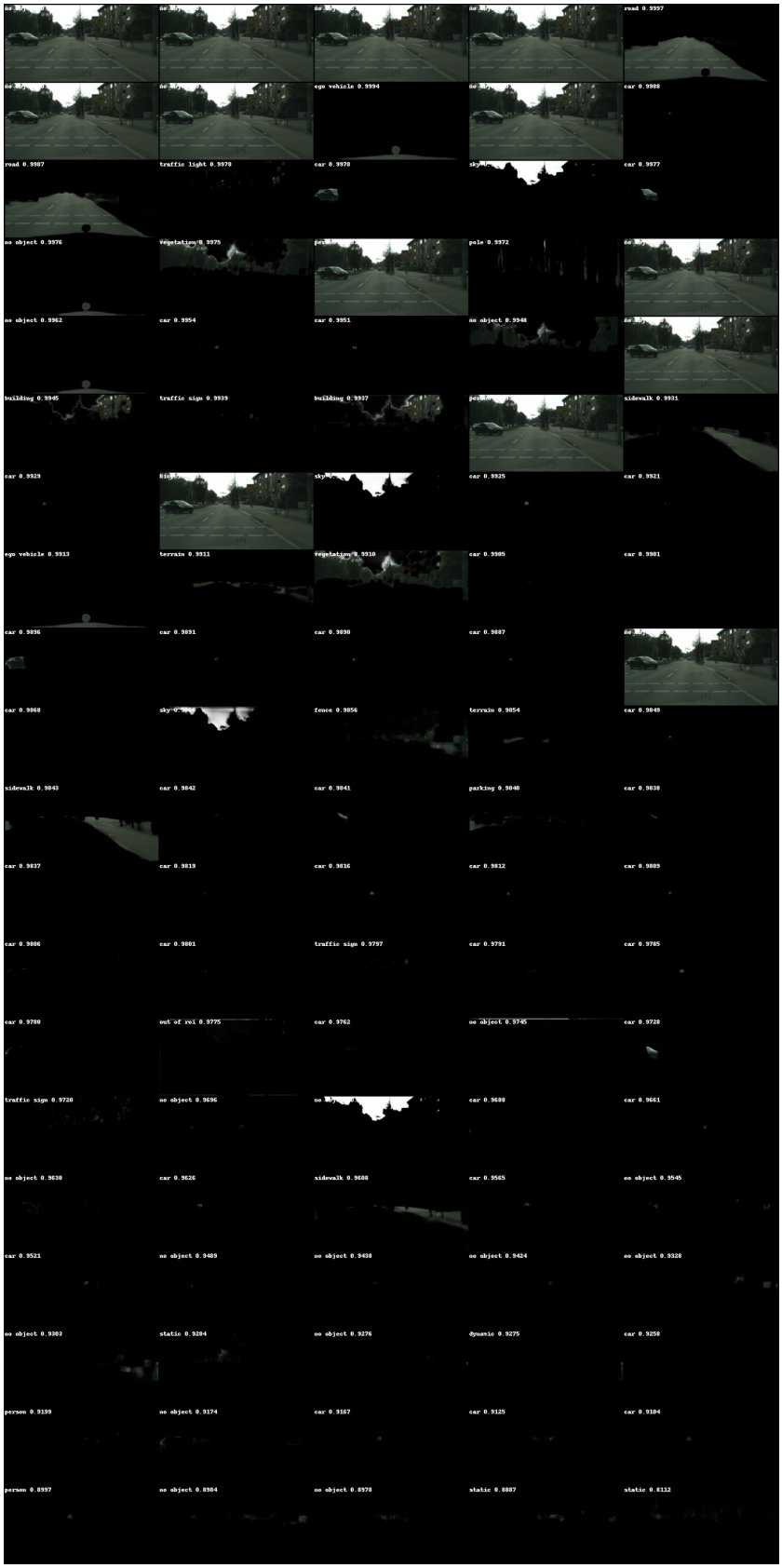

Figure 7. Output of each of the 100 object queries in U-Net for instance segmentation.

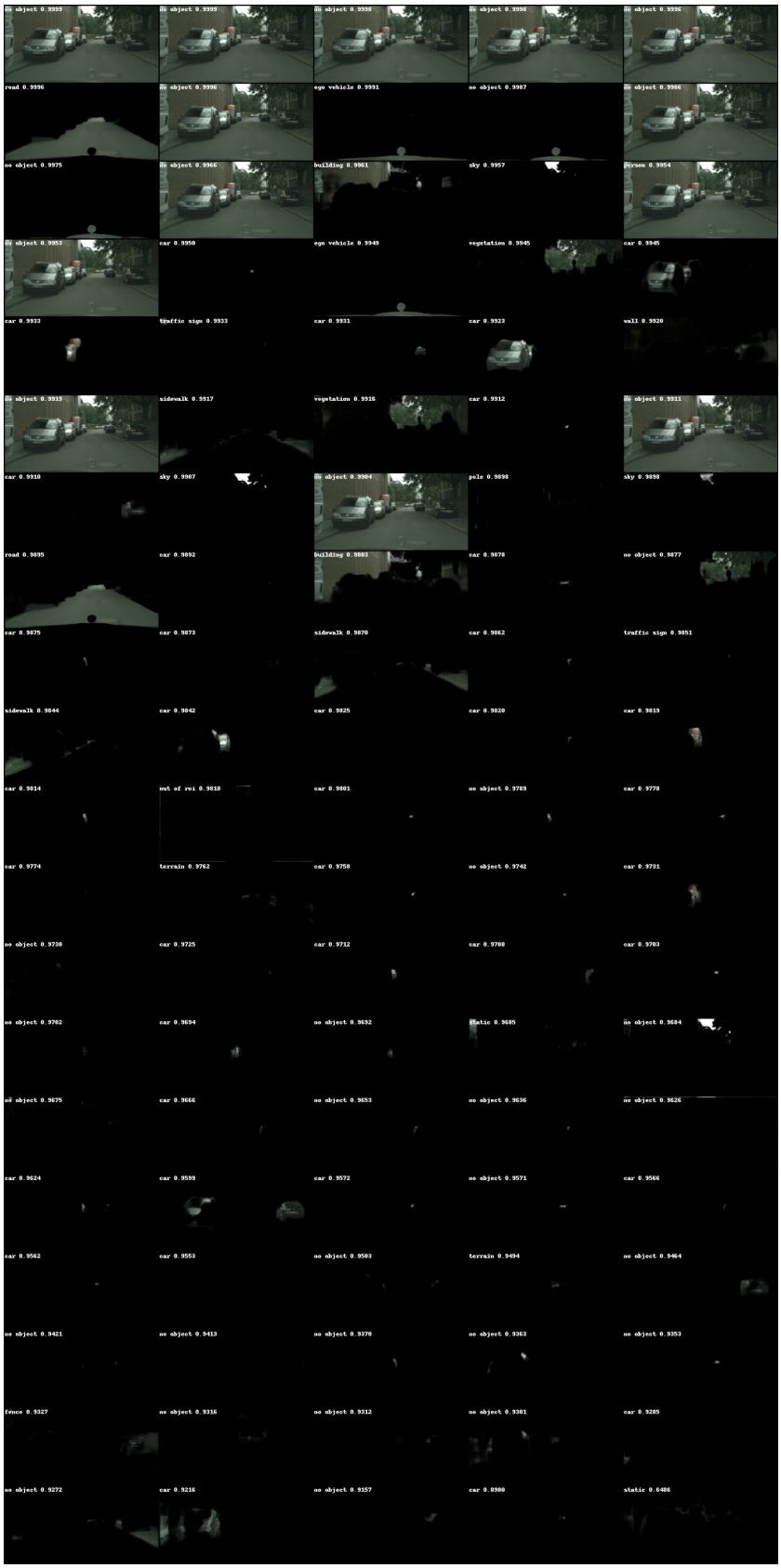

Figure 8. Output of each of the 100 object queries in U-Net for instance segmentation.

Figure 9. Output of each of the 100 object queries in U-Net for instance segmentation.