Vision Transformers II

Rethinking Semantic Segmentation from a Sequence-to-Sequence Perspective with Transformers

Katie Awty-Carroll

21 August 2023

Outline

CNN Recap

- CNNs have long been the SOTA for performing computer vision tasks with deep learning
- CNNs learn to extract image features using stacked convolutional kernels and downsampling
  - This architecture makes CNNs translationally equivariant and locally shift invariant
  - It also introduces a strong inductive bias and assumed locality
- Problem: Learning long-range dependencies is fundamentally difficult because of limited receptive fields.
  - Recent developments towards solving this issue include the use of larger kernel sizes, dilated (atrous) convolutions, and feature pyramids

Enter the Vision Transformer (ViT)

- Dosovitskiy et al. [2020] introduced the idea of the Vision Transformer as a pure transformer architecture for image classification
- ViT processes images by reducing them to a sequence of patches which can be treated in the same way as tokens in a NLP setting, and used as inputs to a transformer decoder
  - Unlike CNNs, ViT has very little image-specific inductive bias. As a result, performance is worse than CNNs on smaller datasets, but with sufficient pre-training, ViT can out-perform CNNs
- Zheng et al. [2020] expand the use of ViT to semantic segmentation
  - This is pixel level rather than image level classification
  - Segmentation models are usually Fully Convolutional Networks (they only contain convolutional and sampling layers)

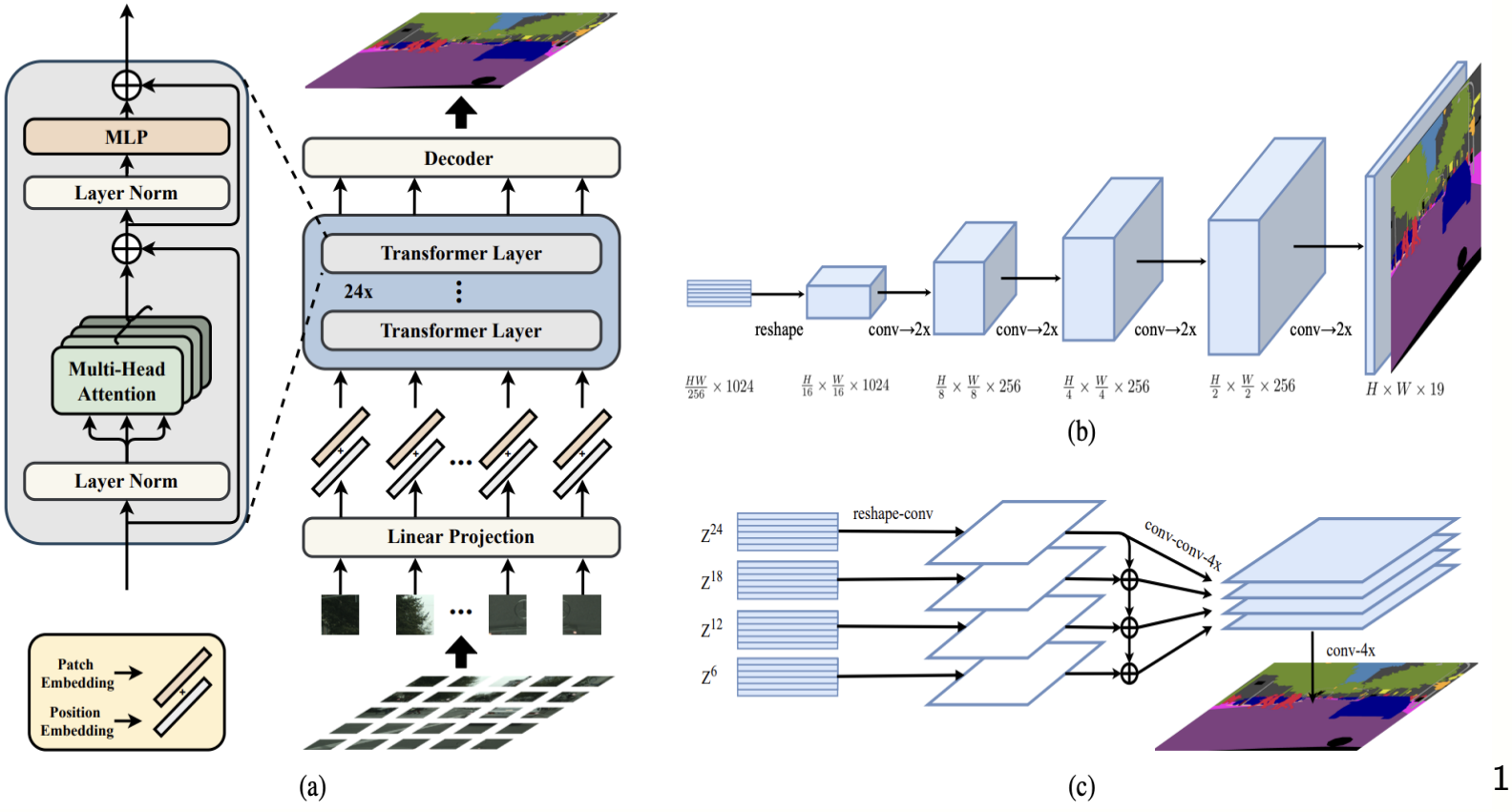
SETR

- Zheng et al. [2020] introduce the SEgmentation TRansformer (SETR)
  - Like ViT, SETR has a decoder arm consisting of transformer layers which accept a sequence of image patch representations
  - SETR replaces the ViT classification head with a decoder arm which generates a pixel-wise segmentation from the transformer features
- SETR was shown to give SOTA performance on two benchmark datasets, ADE20K and Pascal Context

Encoder recap

- Transformer accepts a 1D sequence of feature embeddings $Z \in \mathbb{R}^{L \times C}$ where L is the length of the sequence and C is the hidden channel size
  - Image sequentialization is needed to convert an image $x \in \mathbb{R}^{HxWx3}$ into Z (where 3 is the number of channels)
- A typical CNN encoder would downsample a 2D image into a feature map $x_f \in \mathbb{R}^{\frac{H}{16} x \frac{W}{16} x C}$
- Transformer sequence length L can therefore be set as $\frac{H}{16} x \frac{W}{16} = \frac{HW}{256}$
- To obtain the sequence of length $\frac{HW}{256}$, we divide the image $x \in \mathbb{R}^{HxWx3}$ into a uniform grid $\frac{H}{16} x \frac{W}{16}$, and then flatten this grid into a sequence
- Each vectorized patch p is further mapped into a latent C-dimensional embedding space using a linear projection function $f : p \rightarrow e \in \mathbb{R}^C$
- To encode spatial information, we learn a position encoding p_i for every location i which is added to e_i to form the final sequence input $E = \{e_1 + p_1, e_2 + p_2, ..., e_L + p_L\}$

Architecture



[1]

[1] Zheng et al. [2020]

Transformer

- Given the 1D embedding space E as input, a pure transformer-based encoder is employed to learn feature representations
  - The transformer encoder architecture consists of L_e layers of multi-head self-attention (MSA) and Multilayer Perceptron (MLP) blocks
- This means each transformer layer has a global receptive field
- Transformer features are denoted as $\{Z^1, Z^2, ..., Z^{L_e}\}$ where each feature vector Z has shape $\frac{HW}{256} xC$
- Note that Zheng et al. [2020] include two variants, T-Base where $C = 768$ and T-Large where $C = 1024$

Model	T-layers	Hidden size	Att head
T-Base	12	768	12
T-Large	24	1024	16

[2]

[2] Zheng et al. [2020]

SETR decoders

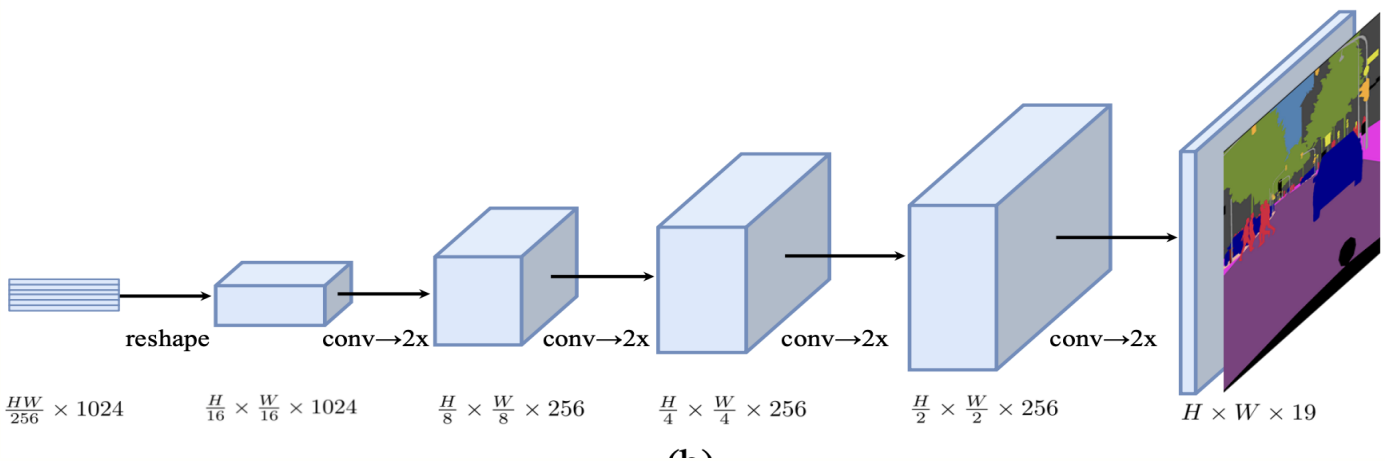
- The goal of the decoder is to generate the segmentation results in the original 2D image space *HxW*
  - To achieve this we need to reshape the encoder's features (Z) from a 2D shape of $\frac{HW}{256}xC$ to a standard 3D feature map $\frac{H}{16}x\frac{W}{16}xC$
  - The reshaped feature map is then fed into a decoder
- SETR compares three decoder designs:
  - Naive upsampling
  - Progressive upsampling (PUP)
  - Multi-Level feature Aggregation (MLA)

Naive upsampling

- First the transformer feature Z^{L_e} is projected into the dimension of the category number
  - e.g. ADE20K has 150 classes, so Z^{L_e} is projected from shape $\frac{H}{16}x\frac{W}{16}x1024$ to $\frac{H}{16}x\frac{W}{16}x150$ (for T-Large)
- This is done using a simple 2-layer architecture: 1x1 conv + (sync) batch norm + 1x1 conv
- Then bilinearly upsample the output to full image resolution, followed by a classification layer with pixel-wise cross-entropy loss

Progressive UPsampling (PUP)

- Instead of one-step upscaling, PUP uses a progressive upsampling strategy that alternates conv layers with upsampling operations
- Upsampling is restricted to 2x, so 4 operations are required to get back to the original image resolution



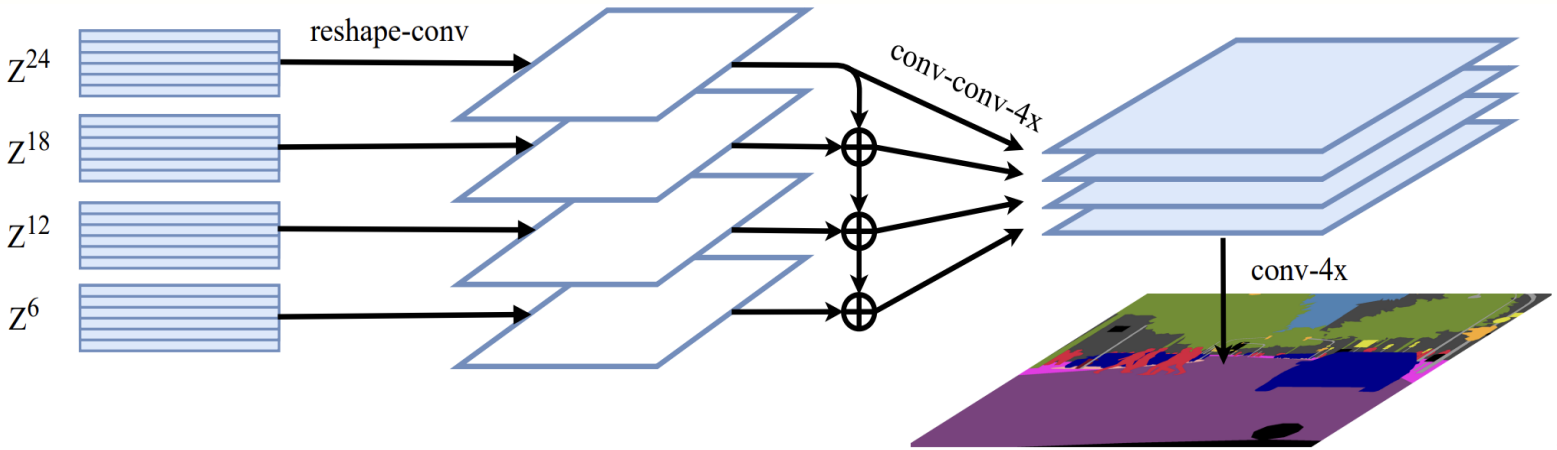
[3]

[3] Zheng et al. [2020]

Multi-Level feature Aggregation (MLA)

- Recall that each feature representation Z^l has the same shape $(\frac{HW}{256}xC)$
- Take as input the feature representations $\{Z^m\}(m \in \{\frac{L_e}{M}, 2\frac{L_e}{M}, ..., M\frac{L_e}{M}\}$ from M layers uniformly distributed across the layers with step $\frac{L_e}{M}$
- Then deploy M streams with each focusing on one specific layer
  - Within each stream we first reshape Z^l to $\frac{H}{16}x\frac{W}{16}xC$
  - A 3-layer (kernel size 1x1, 3x3, 3x3) network is applied with the feature channels halved at the first and third layer respectively
  - To enhance interaction across the streams, top-down aggregations via element-wise addition performed after the first layer
  - This is followed by 4x bilinear upscaling
- After the third convolutional layer, the feature maps are fused via channel-wise concatenation
- The fused feature map is then bilinearly upsampled 4x to obtain the original image resolution

SETR-MLA



[4]

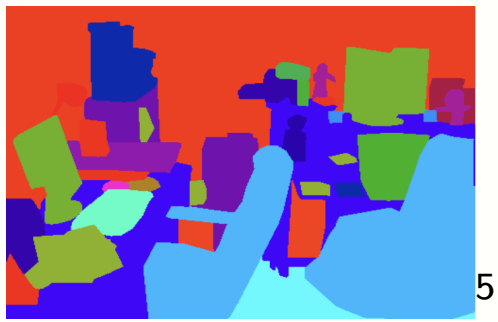
[4] Zheng et al. [2020]

Auxiliary loss

- All three decoder variants are trained using pixel-wise cross-entropy loss
- Zheng et al. [2020] also introduce auxiliary loss heads, using the outputs from different encoder layers
  - SETR-Naive: Layers 10,15,20
  - SETR-PUP: Layers 10,15,20,24
  - SETR-MLA: Layers 6,12,18,24
- Outputs are sent through a two layer Feed-Forward Network before bilinear interpolation
- $Loss = L_{CE} + \gamma \sum_i L_{Aux_i}$

Datasets

- Cityscapes [Cordts et al., 2016]
  - 19 object classes, 3K training images, 500 val images, 1500 test images
- ADE20K [Zhou et al., 2017]
  - 150 semantic classes, 20k training images, 2k val images, 3k test images
  - Challenging scene parsing benchmark
- Pascal Context [Mottaghi et al., 2014]
  - 60 most frequent classes, 5k training images, 5k val
  - Segments whole scene into "thing" and "stuff" classes

[5]

[5] Mottaghi et al. [2014]

Implementation details

- Augmentation pipeline applies random resizing, random horizontal flipping, random cropping
- Batch size 16 for ADE20K and Pascal Context, 8 for Cityscapes
- Polynomial learning rate decay schedule with SGD optimizer
  - Initial learning rate of 0.001 on ADE20K and Pascal, 0.01 on Cityscapes
- Patch size 16x16
- Pre-trained weights from ViT [Dosovitskiy et al., 2020] or DeiT [Touvron et al., 2021] to initialise transformer layers
- Evaluation using Mean Intersection Over Union (mIOU)

Comparing SETR variants

- Comparison of different pre-training strategies and backbones
  - "Pre" denotes the pre-training of transformer part. "R" means the transformer part is randomly initialized [Zheng et al., 2020]
  - Training dataset used is either ImageNet-1k or ImageNet-21k
  - Evaluated on Cityscapes validation set
  - R-101 = ResNet-101

Method	Pre	Backbone	#Params	40k	80k
FCN [39]	1K	R-101	68.59M	73.93	75.52
Semantic FPN [39]	1K	R-101	47.51M	-	75.80
Hybrid-Base	R	T-Base	112.59M	74.48	77.36
Hybrid-Base	21K	T-Base	112.59M	76.76	76.57
Hybrid-DeiT	21K	T-Base	112.59M	77.42	78.28
SETR-*Naïve*	21K	T-Large	305.67M	77.37	77.90
SETR-*MLA*	21K	T-Large	310.57M	76.65	77.24
SETR-*PUP*	21K	T-Large	318.31M	78.39	79.34
SETR-*PUP*	R	T-Large	318.31M	42.27	-
SETR-*Naïve-Base*	21K	T-Base	87.69M	75.54	76.25
SETR-*MLA-Base*	21K	T-Base	92.59M	75.60	76.87
SETR-*PUP-Base*	21K	T-Base	97.64M	76.71	78.02
SETR-*Naïve-DeiT*	1K	T-Base	87.69M	77.85	78.66
SETR-*MLA-DeiT*	1K	T-Base	92.59M	78.04	78.98
SETR-*PUP-DeiT*	1K	T-Base	97.64M	**78.79**	**79.45**

[6]

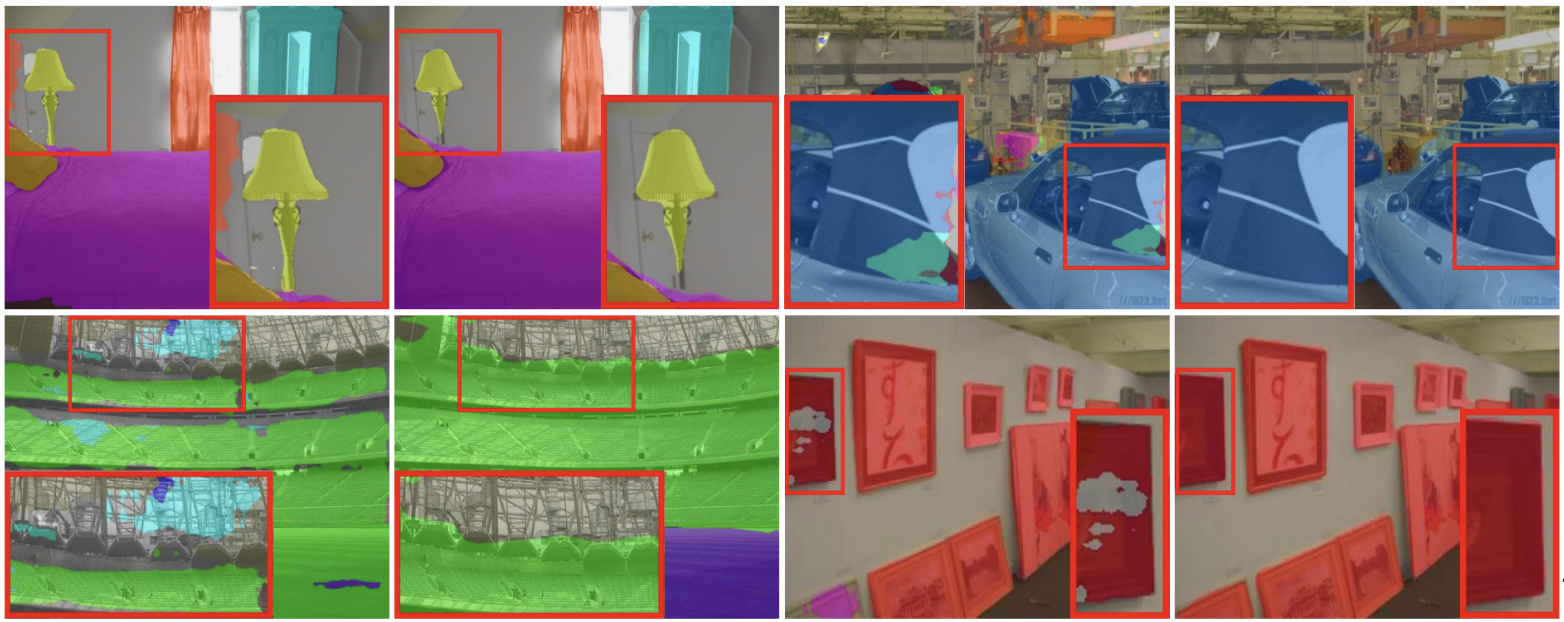
[6] Zheng et al. [2020]

Comparison to SOTA

- ADE20K
  - All SETR variants beat previous SOTA
  - SETR-MLA achieves superior mIOU of 52.8% compared to previous SOTA of 46.4%
- Pascal Context
  - SETR-MLA achieves mIOU of 55.83%, compared to previous SOTA of 54.7%
- Cityscapes
  - Performs well but slightly behind other methods, with SETR-PUP achieving 81.64% compared to SOTA at 81.9%

Qualitative results

ADE20K

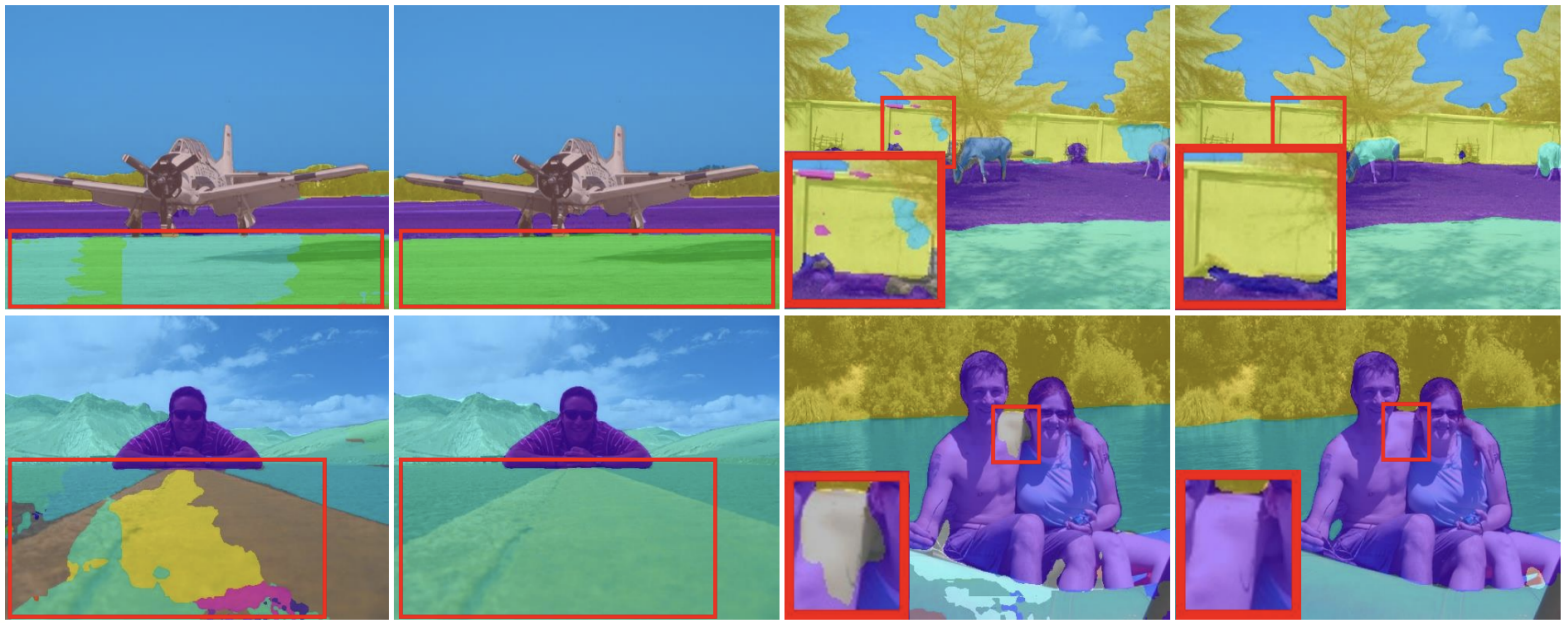


[7]

[7] Zheng et al. [2020]

Qualitative results

Pascal Context



[8]

[8] Zheng et al. [2020]

Qualitative results

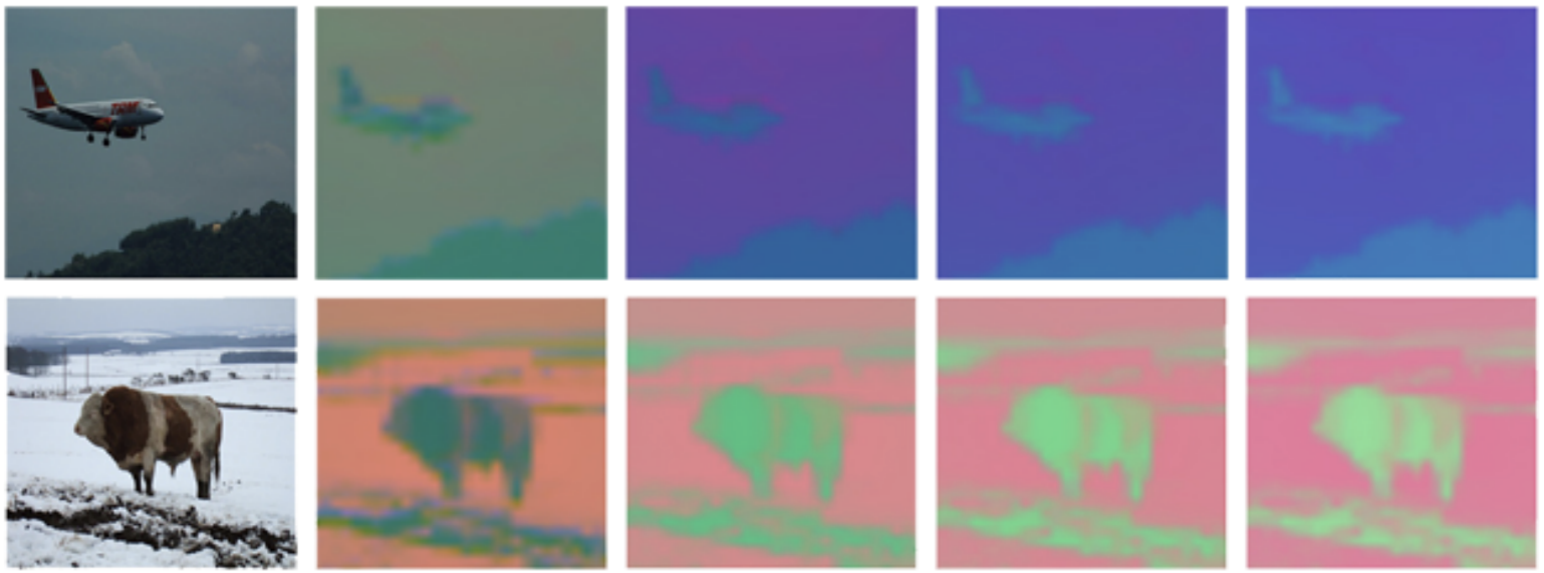
Cityscapes

[9]

[9] Zheng et al. [2020]

Qualitative results

Feature visualisations (layers 1, 9, 17, and 24)



10

[10] Zheng et al. [2020]

Qualitative results

Attention maps

11

[11] Zheng et al. [2020]

Conclusions

- Presented an alternative perspective for semantic segmentation by introducing a sequence-to-sequence prediction framework [Zheng et al., 2020]
- SETR performs well on three standard benchmarking datasets
- Removes reliance on FCN for image segmentation tasks
- Solves issue of limited receptive fields

References

Cordts, M., Omran, M., Ramos, S., Rehfeld, T., Enzweiler, M., Benenson, R., Franke, U., Roth, S., and Schiele, B. (2016). The cityscapes dataset for semantic urban scene understanding. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3213–3223.

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., et al. (2020). An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. *arXiv preprint arXiv:2010.11929*.

Mottaghi, R., Chen, X., Liu, X., Cho, N.-G., Lee, S.-W., Fidler, S., Urtasun, R., and Yuille, A. (2014). The role of context for object detection and semantic segmentation in the wild. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Touvron, H., Cord, M., Douze, M., Massa, F., Sablayrolles, A., and Jégou, H. (2021). Training data-efficient image transformers & distillation through attention. In *International conference on machine learning*, pages 10347–10357. PMLR.

Zheng, S., Lu, J., Zhao, H., Zhu, X., Luo, Z., Wang, Y., Fu, Y., Feng, J., Xiang, T., Torr, P. H., et al. (2020). Rethinking semantic segmentation from a sequence-to-sequence perspective with transformers.

Zhou, B., Zhao, H., Puig, X., Fidler, S., Barriuso, A., and Torralba, A. (2017). Scene parsing through ade20k dataset. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 633–641.