SEGMENTATION OF UAV IMAGES: COMPARISON OF LIGHT SOA MODELS

Andrea Montemurro
Department of Computer Science
University of Bari
Bari, Italy
a.montemurro23@studenti.uniba.it

November 17, 2022

ABSTRACT

Image segmentation is the process of partitioning an image into meaningful regions, and applying this process to images taken from drones or other aerial devices is useful in many processes. Such images are very complex to analyse and at the same time these devices have limited computational resources: it is therefore important to use architectures that work well even if they are not very heavy. In this work, two state-of-the-art networks are compared, which have a relatively small number of parameters in common with others found in the literature. In particular, we want to demonstrate how the SegFormer-based model performs better on a complex dataset containing more than 20 object classes. In our experiment, the *U-Net EfficientNet* architecture achieved about the 40% mIoU and the *SegFormer*-based model achieved the 53% mIoU in its *"B0"* variant, confirming that it performs very well on such complex datasets due to the structure of its encoder. Later, we also wanted to see the performance of a less lightweight variant of *SegFormer (B3)* and achieved a score of about 62% mIoU.

1 Introduction

The goal of semantic segmentation is to segment the input image according to semantic information and predict the semantic category of each pixel from a given label set. With the gradual intellectualization of modern life, more and more applications need to infer relevant semantic information from images for subsequent processing, such as augmented reality, autonomous driving, video surveillance, etc. [1]

More specific, aerial imagery is analyzed for a broad number of applications. Our work is inspired by the fact that high-resolution aerial imagery analysis helps to address topics such as deforestation [2], declining biological diversity [3], refugee camp planning and maintenance, global poverty[4], urban planning, precision agriculture, and geographic information system (GIS) updating [5].

2 Related work

Convolutional neural networks (CNNs) have shown impressive performance in many kinds of image segmentation. The well known *U-Net* architecture is a crucial baseline for image segmentation developments.It won the ISBI challenge for segmentation of neuronal structures in electron microscopic stacks. Furthermore, the authors show that their solution requires a very small number of training examples: it achieved a mean Intersection-Over-Union (mIoU) of 92% on a 'PhC-U373' cells dataset after training on 35 example 512x512 pixel images[6]. *U-Nets* are very effective for tasks where the output size is similar to the input size, where the output needs to be the same high resolution. This makes them very good for creating segmentation masks, amongst other tasks.

Segmenting datasets with one [7], [8] or a few classes [9] has been done using *U-Nets* or similar performing architectures, but aerial imagery's biggest potential is the ability to capture a large portion of the earth (with a larger number of classes)

Figure 1: Examples of couples image-mask from the dataset.

at the same time. Physically smaller classes (for example cars, persons) cause accuracy issues and class imbalance, presumably because small classes give the network a relatively small number of pixels to learn from and to do inference on. The challenge of aerial images' resolution gets amplified by the fact that the classes we are trying to predict in an image contain less pixels than the classes in traditional ground level images. This is where FCN's and *U-Nets* segmentation masks lack quality, because of excessive downsizing due to consecutive pooling operations.

New techniques for achieving good scores in segmentation of this type of image are mostly based on the *DeepLabV3+* model, which uses dilated convolution[10]. For example, there is a work [11], in which the autors reach the 52,5% of mIoU on the test set, using a *DeepLabV3+ Xception65* (a huge architecture) on the *Drone Deploy Dataset*, a dense labeled dataset with only 6 classes.

Currently, *ViT*-based models perform best on the semantic segmentation task. In particular Vision Transformer Adapter are the best performing architectures on image segmentation benchmarks, achieving about 60% mIoU on the *ADE20K* benchmark or about 85% on *Cityscapes*. This architecture, although very efficient is very expensive as it has more than 450M parameters to train[12].

However, a transformer, *SegFormer*, has recently been proposed in the literature that is very simple but offers acceptable performance. The lightest variant, with only 3.8M parameters achieves a score of about 38% mIoU, while the B5 variant reaches 52% mIoU on *ADE20K*, and has a complexity of 84M parameters[13].

3 Materials

Dataset

The dataset that we are going to use in order to explore the performance of the models in the scenario of the drone images semantic segmentation is the Semantic Drone Dataset[14] from the austrian academic group Institute of Computer Graphics and Vision.

The Semantic Drone Dataset focuses on semantic understanding of urban scenes for increasing the safety of autonomous drone flight and landing procedures. The imagery depicts more than 20 houses from nadir (bird's eye) view acquired at an altitude of 5 to 30 meters above ground. A high resolution camera was used to acquire images at a size of 6000x4000px (24Mpx). The dataset is released by the author into two parts: a training set that contains 400 publicly available images and a the test set which is made up of 200 private images.

Since the "test set" is private, for our experiments we are going to use only the so-called "training set", but we are going to apply a train-validation-test splitting on the 400 public images to train and test our models.

In the dataset there are 23 classes: *paved-area, dirt, grass, gravel, water, rocks, pool, vegetation, roof, wall, window, door, fence, fence-pole, person, dog, car, bicycle, tree, bald-tree, ar-marker, obstacle, conflicting*.

Data Pre-processing

In order to make the data ready for the model, we pre-process the images of the dataset by resizing each couple image-mask with a dimension equal to 512x512px. We resized the images, using the *nearest-neighbor interpolation* as interpolation method. In addition, the pixel values of the RGB channels of the images were also normalised against the average ImageNet values.

Data Splitting

We divided the dataset composed by 400 images into three subsets: training, validation and test. The training set after the splitting contains 306 images, while the validation set contains 54 images and the test set 40 images.

4 Methods

In this work we are going to compare the performance on the previous described dataset of two state-of the-art models. The first, is an older one, the well known *U-Net EfficientNet* architecture, which is used as a "baseline" with respect to the newer *SegFormer* architecture, a new, powerful but very "lightweight" Vision Transformer. As said before in the document, the semantic segmentation is a complicated task but unfortunately drones do not have large computational resources, so in this paper it makes sense to compare two models that are state-of-the-art but at the same time do not require exaggerated computational resources. We therefore chose two models that, from a complexity point of view, have a similar number of training parameters:

- ***U-Net*** architecture with ***EfficientNet-b0*** encoder, which have around 11M paramaters;
- ***SegFormer-B0***, the simplest implementation of this architecture, with 3.8M of parameters.

Both architectures are pre-trained on the ImageNet dataset.

As already mentioned, we are going to use the mean Intersection over Union (mIoU), or Jaccard Index, as evaluation metric for our work. We are going to use this metric because works well for the class imbalance problem.

Later we wanted to extend this experimentation to include ***SegFormer-B3***, which has a total of 47M parameters.

$$mIOU = \frac{\sum_{i=1}^{C} \frac{A_1 \cap B_1}{A_i \cup B_i}}{C} \tag{1}$$

In equation 1 , the intersection $(A_i \cap B_i)$ for each class $1 \leq i \leq C$, comprises the pixels found in both the prediction mask A_i and the ground truth mask B_i. The union $(A_i \cup B_i)$ includes all pixels found in either the prediction mask or the ground truth mask. After computing the IOU for each trainable class, the score is averaged out over the number of classes C, with or without taking into account class imbalance. Most of the time the unbalanced version is used, as displayed in equation 1.

U-Net EfficientNet architecture

The *U-Net* was originally developed a network for bio medical image segmentation[6]. The architecture contains two paths.

The first path is the *contraction path* (also called as the encoder) which is used to capture the context in the image. The **encoder** is just a traditional stack of convolution and max pooling layers.

The second path is the symmetric *expanding path* (also called as the **decoder**) which is used to enable precise localization using transposed convolutions.

Thus it is an end-to-end fully convolutional network (FCN), i.e. it only contains convolutional layers and does not contain any dense layer because of which it can accept image of any size. In the original paper, the *U-Net* is described as follows:

The left hand side is the contraction path (encoder) where we apply regular convolutions and max pooling layers. In this path, the size of the image gradually reduces while the depth gradually increases.

The right hand side is the expansion path (decoder) where transposed convolutions are applied along with regular convolutions. In the decoder, the size of the image gradually increases and the depth gradually decreases.

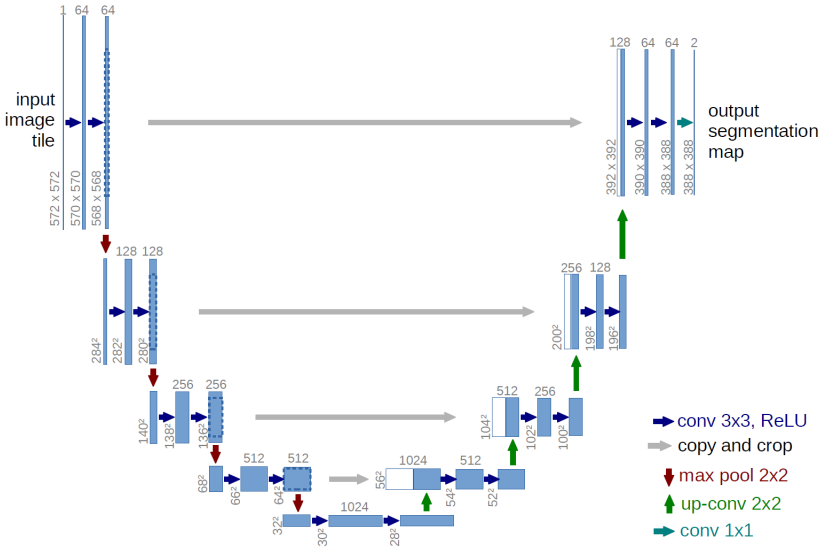


Figure 2: U-Net architecture.

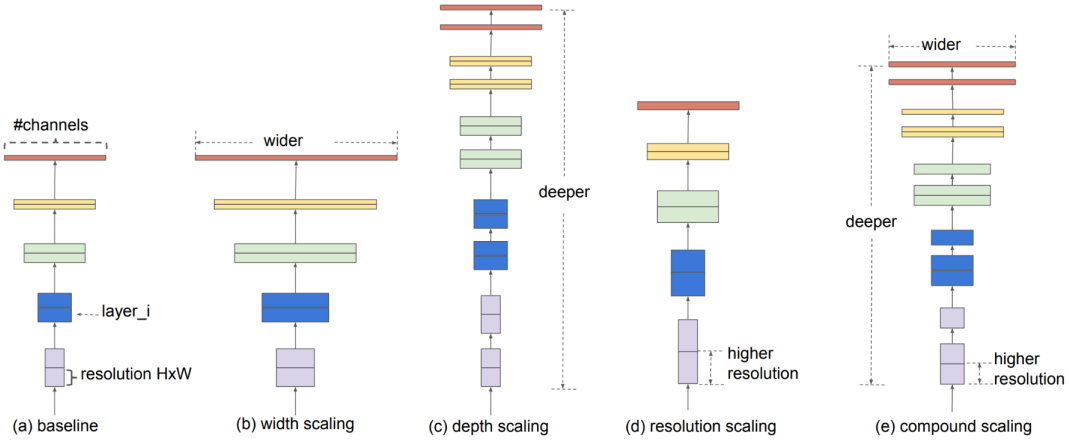


Figure 3: Scaling types in a CNN.

In this work we decided to use as the network encoder, another type of convolutional network that is "lighter" than the one used in the original paper. It is *EfficientNet* in its B0 variant.

EfficientNet is a convolutional neural network architecture and scaling method that uniformly scales all dimensions of depth/width/resolution using a compound coefficient. Unlike conventional practice that arbitrary scales these factors, the *EfficientNet* scaling method uniformly scales network width, depth, and resolution with a set of fixed scaling coefficients. For example, if we want to use 2^N times more computational resources, then we can simply increase the network depth by α^N, width by β^N, and image size by γ^N, where α, β, γ are constant coefficients determined by a small grid search on the original small model. *EfficientNet* uses a compound coefficient ϕ to uniformly scales network width, depth, and resolution in a principled way.

The compound scaling method is justified by the intuition that if the input image is bigger, then the network needs more layers to increase the receptive field and more channels to capture more fine-grained patterns on the bigger image[15].

The base *EfficientNet-B0* network is based on the inverted bottleneck residual blocks of *MobileNetV2*[16], in addition to squeeze-and-excitation blocks.

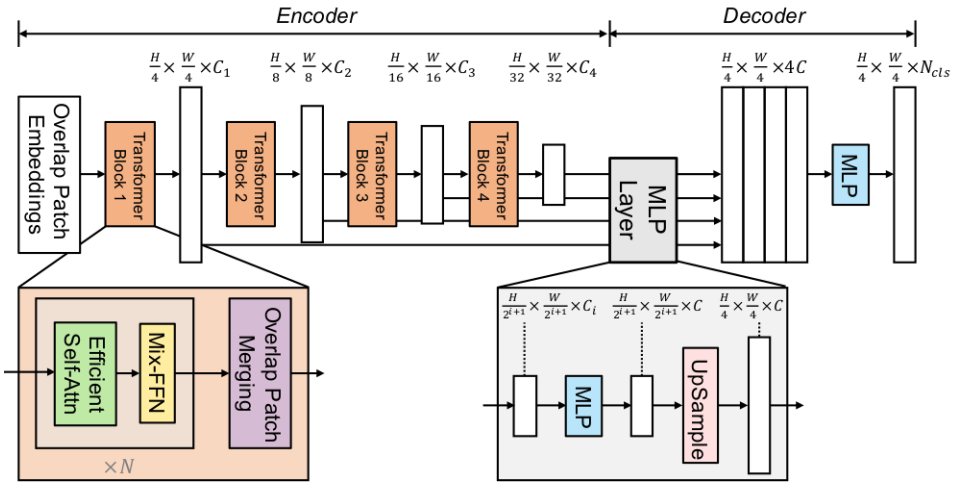


Figure 4: SegFormer architecture.

SegFormer architecture

SegFormer is a simple, efficient yet powerful semantic segmentation framework which unifies Transformers with lightweight multilayer perception (MLP) decoders. The architecture of this model is based on an encoder that outputs multi-scale features and a Multi-Layer Perceptron (MLP) decoder that aggregates this information from different layers[13].

As shown in the image describing the architecture, the left part is the encoder part while the right part is the decoder part.

In the **encoder**, the input image is divided into 4x4 patches as is done in ViTs[17], in which, however, a size of 16 is used for the patches. The use of smaller sized patches makes better dense the prediction tasks. Next is the first of 4 **transformer blocks**, which is itself composed of 3 modules:

1. **Efficient Self Attention**, which reduces the sequence to lower computational cost.
2. **Mixed FFN**, used to implement data driven positional encoding.
3. **Overlap Patch Merging**, used to reduce the feature map size.

In subsequent encoder blocks, the size of the features is reduced. The **decoder** part is simpler than the encoder. In this decoder, all the features of different sizes produced by the encoder, are "fused" together. The entire decoder consists of four main steps:

1. Features from the encoder go through the MLP to unify in channel dimension.
2. Features are up-sampled to 1/4th of its size and are concatenated together.
3. A MLP is adopted to fuse the concatenated features.
4. Another MLP layer takes the fused features to predict the segmentation mask.

Fine-tuning

We are going to fine tune such models on the Semantic Drone Dataset with similar settings in order to compare their performance on this task.

We used the Segmentation Models framework with PyTorch[18] for implementing the *U-Net EfficientNet* architecture, and the HuggingFace Tranformers framework[19] for the *SegFormer* implementation.

We train our models both in 20 epochs, using a batch size equal to 4.

Since we are in multi-class classification problem we use the Cross Entropy as loss function.

$$-\sum_{c=1}^{M} y_{o,c} \log\left(p_{o,c}\right) \tag{2}$$

mIoU %	***U-Net***	***SegFormer-B0***	**SegFormer-B3**
Train Set	41,8%	56,9%	**75,7%**
Valid Set	37,2%	52,5%	***63,4%***
Test Set	39,6%	52,5%	***61,7%***

Table 1: mIoU scores for each set of data.

Where, the M is the number of classes, the c is the specific class and o is the specific observation for that class.

As optimization algorithm, we used Adam by setting as the maximum learning rate. For this setting we have a difference, since we used a smaller learning rate in the *SegFormer* training. In fact, we set as max learning rate 0.00006 and for the *U-Net EfficientNet* we set 0.001. We chose these values with a trial-and-error approach, in particular for the *SegFormer* model, we noticed that a lower learning rate was necessary as by setting it similar to the other model we could not decrease the value of the loss at each epoch by much.

The learning process took about one hour for the *U-Net EfficientNet* model and about three hours for the *SegFormer*-based model, using the *Nvidia Tesla P100 GPU* provided by the *Kaggle* platform for free.

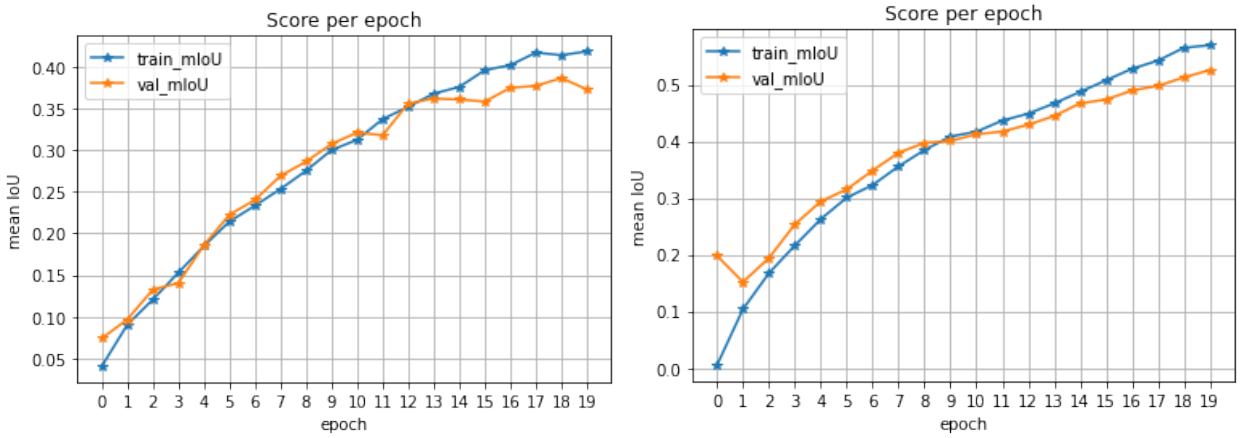
5 Results

In this section we go into more detail about the performance of the models in the training phase and at the end of this phase.

After 20 epochs we obtained the results in the table 1. As we can see our *SegFormer-B0* transformer performs quite better than the *U-Net EfficientNet* model.

The *SegFormer-B0* has good performance (around 50% mIoU), with onli 3.8M of trainable parameters, while the *U-Net EfficientNet* don't have good performance on such dataset, as expected.

The graphs, show the performance trends for each epoch in the training/validation phase of the two models. From the graph we can see that the *SegFormer-B0* reaches the *U-Net EfficientNet* score in half of the epochs, and that both models could be trained for more epochs, but since the score on the validation set does not increase as much as the score on the training set we would risk an overfitting situation.



(a) U-Net EfficientNet mIoU.

(b) SegFormer-B0 mIoU.

Figure 5: Plots of the score reached in every epoch for both models.

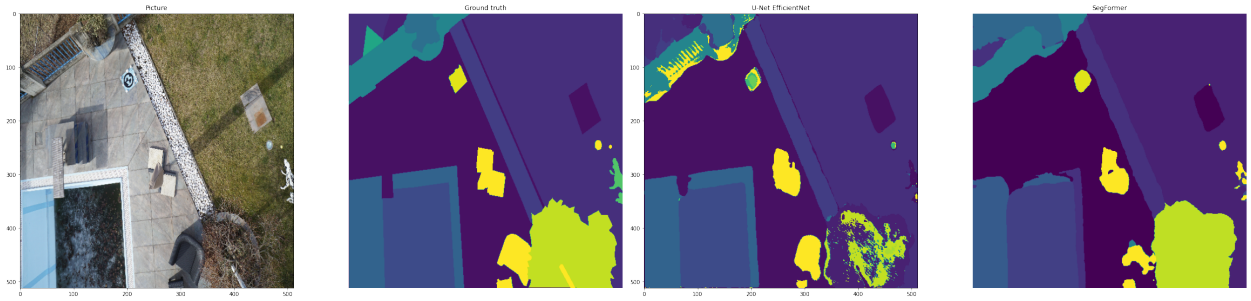
As mentioned earlier, we also trained a *SegFormer-B3* , whose achieved mIoU values are given only in the table for completeness.

6 Conclusions

As shown in the previous paragraph in the training and testing results of the two models the one based on *SegFormer* performs quite better than the one based on *U-Net EfficientNet*.

In particular this is due to a better ability of this model to pick up differences on the smallest details and thus to be able to recognize the classes of pixels that are less present in an image, due to its encoder architecture.

We can also see this from an image and how it is classified by both models.



(a) An image and its mask.

(b) Predicted masks.

Figure 6: Comparison of the predictions and the ground truth.

The image depict well this situation, the first difference we can see in the ability to recognize the top right gate quite well with *SegFormer*, while *U-Net EfficientNet* completely misses. In addition, the part of vegetation in the lower right corner is significantly better in the mask predicted by *SegFormer*.

Overall, we can say that the results are acceptable. The dataset used is a very complex benchmark in that it includes a large number of classes and it is therefore difficult for any model to correctly recognize all classes, especially those that describe small objects/subjects and are therefore represented in the minority. Most importantly, we obtained acceptable results with scores similar to [11], but using a more complex dataset having 23 classes versus only 6, and using a significantly lighter model in terms of trainable parameters.

References

[1] Mo, Yujian and Wu, Yan and Yang, Xinneng and Liu, Feilin and Liao, Yujun. Review the state-of-the-art technologies of semantic segmentation based on deep learning. In *Neurocomputing*, volume 493, pages 626–646. Elsevier, 2022.

[2] Green, Glen M and Sussman, Robert W. Deforestation history of the eastern rain forests of Madagascar from satellite images. In *Science*, volume 248, pages 212–215. American Association for the Advancement of Science, 1990.

[3] Richards, Daniel R and Friess, Daniel A. Rates and drivers of mangrove deforestation in Southeast Asia. In *Proceedings of the National Academy of Sciences*, volume 113, pages 344–349. National Acad Sciences, 2016.

[4] Roser, Max and Ortiz-Ospina, Esteban. Global extreme poverty. In *Roser, Max and Ortiz-Ospina, Esteban*, Our world in data, 2013.

[5] Cheng, Gong and Han, Junwei. A survey on object detection in optical remote sensing images. In *ISPRS Journal of Photogrammetry and Remote sensing*, volume 117, pages 11–28. Elsevier, 2016.

[6] Ronneberger, Olaf and Fischer, Philipp and Brox, Thomas. U-net: Convolutional networks for biomedical image segmentation.

[7] Shermeyer, Jacob and Hogan, Daniel and Brown, Jason and Van Etten, Adam and Weir, Nicholas and Pacifici, Fabio and Hansch, Ronny and Bastidas, Alexei and Soenen, Scott and Bacastow, Todd and others. SpaceNet 6: Multi-sensor all weather mapping dataset. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 196–197. 2020.

[8] Soman, Kritik. Rooftop detection using aerial drone imagery. In *Proceedings of the ACM India Joint International Conference on Data Science and Management of Data*, pages 289–296. 2018.

[9] Sang, Dinh Viet and Minh, Nguyen Duc. Fully residual convolutional neural networks for aerial image segmentation. In *Proceedings of the Ninth International Symposium on Information and Communication Technology*, pages 281–284. 2019.

[10] Chen, Liang-Chieh and Papandreou, George and Schroff, Florian and Adam, Hartwig. Rethinking atrous convolution for semantic image segmentation. In *arXiv preprint arXiv:1706.0558*. 2017.

[11] Heffels, Michael R and Vanschoren, Joaquin. Aerial imagery pixel-level segmentation. In *arXiv preprint arXiv:2012.02024*. 2020.

[12] Chen, Zhe and Duan, Yuchen and Wang, Wenhai and He, Junjun and Lu, Tong and Dai, Jifeng and Qiao, Yu. Vision Transformer Adapter for Dense Predictions. In *arXiv preprint arXiv:2205.08534*. 2022.

[13] Xie, Enze and Wang, Wenhai and Yu, Zhiding and Anandkumar, Anima and Alvarez, Jose M and Luo, Ping. SegFormer: Simple and efficient design for semantic segmentation with transformers. In *Advances in Neural Information Processing Systems*, volume 34, pages 12077–12090. 2021.

[14] ICG Drone Dataset, `https://www.tugraz.at/index.php?id=22387r`

[15] Tan, Mingxing and Le, Quoc. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International conference on machine learning*, PMLR, pages 6105–6114. 2019.

[16] Sandler, Mark and Howard, Andrew and Zhu, Menglong and Zhmoginov, Andrey and Chen, Liang-Chieh. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4510–4520. 2018.

[17] Dosovitskiy, Alexey and Beyer, Lucas and Kolesnikov, Alexander and Weissenborn, Dirk and Zhai, Xiaohua and Unterthiner, Thomas and Dehghani, Mostafa and Minderer, Matthias and Heigold, Georg and Gelly, Sylvain and others. An image is worth 16x16 words: Transformers for image recognition at scale. In *arXiv preprint arXiv:2010.11929*. 2020.

[18] Segmentation Models Pytorch, `https://github.com/qubvel/segmentation_models.pytorch`

[19] Thomas Wolf and Lysandre Debut and Victor Sanh and Julien Chaumond and Clement Delangue and Anthony Moi and Pierric Cistac and Tim Rault and Rémi Louf and Morgan Funtowicz and Joe Davison and Sam Shleifer and Patrick von Platen and Clara Ma and Yacine Jernite and Julien Plu and Canwen Xu and Teven Le Scao and Sylvain Gugger and Mariama Drame and 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*, Association for Computational Linguistics, `https://www.aclweb.org/anthology/2020.emnlp-demos.6`, pages38–45, oct. 2020.