Міністерство освіти і науки України

Національний університет «Києво-Могилянська академія»

Факультет інформатики

Кафедра математики

Магістерська робота

освітній ступінь – магістр

на тему: «**СЕМАНТИЧНА СЕГМЕНТАЦІЯ ЗОБРАЖЕНЬ З ВИКОРИСТАННЯМ TRANSFORMER АРХІТЕКТУРИ**»

Виконав: студент 2-го року навчання,

освітньо-наукової програми
«Прикладна математика», 113

Іванюк-Скульський Богдан
Віталійович

Керівник Швай Н.О.,

кандидат фіз.-мат. наук, доцент

Рецензент_____
(прізвище та ініціали)

Магістерська робота захищена
з оцінкою _____

Секретар ЕК _____
«____» _____ 2022 р.

Київ – 2022

ГРАФІК ПІДГОТОВКИ МАГІСТЕРСЬКОЇ РОБОТИ ДО ЗАХИСТУ

№ з/п	ПЕРЕЛІК РОБІТ	Термін виконання	Дата ознайомлення наукового керівника	Підпис наукового керівника	Примітки
1.	Вибір теми та закріплення наукового керівника.	22.10.21			
2.	Отримання списку основної літератури за темою роботи та індивідуального завдання	04.11.21			
3.	Базове ознайомлення з літературою, науковими статтями	04.11.21 – 25.11.21			
4.	Уточнення, обговорення і корекція запропонованих питань індивідуального завдання	25.11.21			
5.	Поглиблене вивчення літератури, наукових статей за пов'язаними темами, ознайомлення з усіма означеннями та твердженнями з літератури	*грудень - квітень*			
6.	Складання календарного графіку виконання, оформлення індивідуального завдання, написання розділів магістерської роботи	25.04.22 – 26.04.22			
7.	Написання магістерської роботи в цілому, ознайомлення з її першим варіантом наукового керівника				
	Розділ 1 Огляд моделей сегментації та мобільних моделей класифікації (постановка проблеми, теоретичні основи, огляд літературних джерел)	*травень*			
	Розділ 2 Моделі сегментацій MobileViT + MTD, ResNet + MTD	*травень*			
	Розділ 3 Дослідження варіантів покращення моделей (порівняння, аналіз і візуалізація результатів)	*травень - червень*			
8.	Повне завершення написання магістерської роботи, оформлення її згідно з вимогами й подання на відгук науковому керівнику	*червень*			
	Подання магістерської роботи для перевірки письмових робіт студентів НаУКМА на відповідність вимогам академічної доброчесності	*червень*			
	Подання роботи на зовнішню рецензію	*червень*			
	Підготовка до захисту магістерської роботи на засіданні кафедри: написання доповіді та виготовлення ілюстративного матеріалу	*червень*			
7.	Попередній захист магістерської роботи на засіданні кафедри	*червень*			
8.	Подання магістерської роботи на кафедру з усіма супроводжувальними документами	*липень*			
9.	Публічний захист магістерської роботи перед екзаменаційною комісією	*липень*			

Графік узгоджено «___» _____ 2022 р.

Науковий керівник Швай Н.О. (ПІБ)

Виконавець магістерської роботи Іванюк-Скульський Б.В. (ПІБ)

Національний університет «Києво-Могилянська академія»

Факультет інформатики
Кафедра математики
Освітній ступінь магістр
Спеціальність 113 Прикладна математика
(шифр і назва)

ЗАТВЕРДЖУЮ
Зав. кафедри математики
Проф., д. ф-м. н. Б.В. Олійник

“____” _____20_22_року

З А В Д А Н Н Я

ДЛЯ МАГІСТЕРСЬКОЇ РОБОТИ СТУДЕНТУ

Іванюку-Скульському Богдану Віталійовичу
(прізвище, ім’я, по батькові)

1. Тема роботи «**СЕМАНТИЧНА СЕГМЕНТАЦІЯ ЗОБРАЖЕНЬ З ВИКОРИСТАННЯМ TRANSFORMER АРХІТЕКТУРИ**»

керівник роботи Швай Н.О., доцент, кандидат фіз.-мат. наук
(прізвище, ім’я, по батькові, науковий ступінь, вчене звання)

затверджені наказом вищого навчального закладу від « 22 » жовтня 2021 року № 890-с

2. Строк подання студентом роботи до 30 червня 2022р.

3. План роботи

1. Формалізація нового методу сегментації зображень в умовах обмеженого ресурсу
2. Побудова експерименту
3. Проведення експерименте
4. Аналіз отриманих результатів
5. Оформлення тексту магістерської роботи

Contents

Introduction

Convolutional neural networks have been the standard for computer vision since AlexNet [2] demonstrated that deep convolution neural networks show state-of-the-art results on large image classification datasets such as ImageNet [3]. Their spatial inductive offsets allow the study of representations of images with fewer parameters, but these networks are spatially local. Vision Transformer (ViT) [4] architecture has been proposed to study global representations. The Transformer architecture is the de facto modern standard for natural language recognition through the Self-Attention block, which allows you to study representations for the entire sequence of word vectors at any given time. The main approach to using these models is to train on large amounts of text data, and then continue to train the model for a specific task. This training process is also useful for computer vision tasks. Preliminary training uses the teacher training method, where the input image is divided into patches (cut parts of the image), passed through a layer of projection into the vector space, and fed to the input of the Transformer. After that, the studied representations of images can be used in the tasks of object detection, image description, face recognition, and many others.

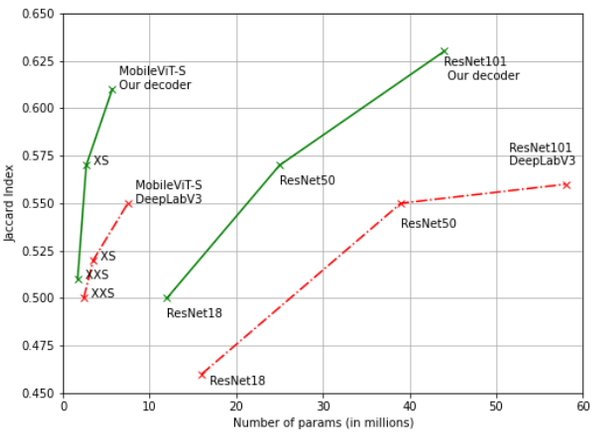

Figure 1: Model size vs model performance: proposed Transformer based decoder with Attention-Convolution layers outperforms corresponding models with DeepLabV3 [1] decoder. In particular, ResNet101 with our decoder improves Jaccard index by 7% with one third less parameters than corresponding DeepLabV3 decoder.

The general trend of research has been to create deep and complex networks to achieve greater accuracy. However, these advances in accuracy do not necessarily improve the model's efficiency in terms of size and speed. In many neural network applications, such as robotics, self-driving cars, and augmented reality, recognition tasks must be performed on limited computing resources and in limited time. For

this reason, models are created that show good results but require much less computing resources. Examples of such models are the MobileNet family [5] [6] [7] and the recently published MobileViT model [8], which combines the strengths of convolutional and ViT networks.

In this work, we focus on creating mobile segmentation decoder using Transformer blocks. General model architecture is presented in Figure 3.1 and is comprised on the MobileViT encoder and our decoder. Our proposed decoder is based on two consecutive Transformer Encoder blocks, three convolution layers with attention, and a segmentation head. This decoder improves model performance by at least 1% in the Jaccard index compared to DeepLabV3 [1], with fewer parameters.

Chapter 1

Related Work

For the past years, the dominant approach to segmentation tasks was to use encoder-decoder architecture. The basic building block for such models was Fully Convolutional Blocks (FCB). Initially, the architectures relied on stacks of FCB, [9, 10].

Later, FCB were combined with skip-connection of higher-level features from encoder models, [11, 12, 13]. These modified architectures improved the sharpness of segmentation masks because they combined lower-level and higher-level representations and provided sharpness information.

To further improve the performance, by increasing the receptive field, dilated convolutions were introduced, [14, 15, 16].

Recently, Transformer architecture gained popularity. Transformer architectures are widely used as an encoder network, [17, 18, 19], and are starting to be used as a decoder network, [20].

Chapter 2

Theory

2.1 Semantic segmentation

Image segmentation task is essential for a number of applications, including autonomous driving, medicine, etc. The purpose of this task is to classify each pixel of an image to a corresponding class, mathematically it can be formulated as following

$$f_\theta(X) = \hat{Y} \tag{2.1}$$

where X is an input image, f is a model with weights θ that transforms input image to an output segmentation mask $\hat{Y}$.

Figure 2.1: PASCAL VOC 2012 sample images with corresponding masks.

2.2 Convolution layer

Convolution kernel (filter) is a basic building block of many computer vision models. This convolution is applied to an input image, and the output is a transformation of an original image be the values from the kernel.

$$G(m,n) = \sum_j \sum_k h[j,k] f[m-j][n-k], \tag{2.2}$$

where h is a convolution kernel, f - input matrix (either image or feature map)[7], m and n stand for indexes of rows and column, respectively.

2.3 Transformer architecture

Transformer architecture was first introduced in a paper by Vaswani et al.[21]. Originally the paper addressed natural language processing task, in particular machine translation. Since than, Transformer architecture became widely used in computer vision in the models like Visual Transformer [4], Swin Transformer [**?**], SepViT [22], DeiT [23], etc.

Scaled Dot-Product Attention is performed using three matrices that represent queries (Q), keys (K), and values (V). These matrices are computed using a feed-forward neural network with a single hidden layer. Matrix output is computed using the following formula

$$Attention(Q, K, V) = softmax(\frac{QK^T}{\sqrt{d_k}})V \tag{2.3}$$

where dot product is computed on queries and keys, divided by $\sqrt{d_k}$ and applying softmax function to get the weights across values. Scaling factor $\frac{1}{\sqrt{d_k}}$ is used to prevent softmax function from shifting to the regions with small gradients.

Instead of using a single attention head, multiple (h) attention heads are trained in parallel. On each head, queries, keys, and values are learning representations from different subspaces. On the output, h d-dimensional vectors are concatenated and projected with a matrix W_0 as in the following formula

$$MultiHead(Q, K, V) = Concat(head_0, head_1, ..., head_n)W_O, \tag{2.4}$$

where $head_i = Attention(QW_i^Q, KW_i^K, VW_i^V)$. Projections are obtained by weight matrices $W_i^Q \in \mathbb{R}^{d_{model} \times d_k}$, $W_i^K \in \mathbb{R}^{d_{model} \times d_k}$, $W_i^V \in \mathbb{R}^{d_{model} \times d_v}$, and $W^O \in \mathbb{R}^{hd_v \times d_{model}}$.

2.4 Loss function

Dice Loss J [24], sometimes known as Jaccard index, was used as a loss function using which we are trying to optimize our models. The function is bounded by 0 and 1, and can be mathematically formulated as follows:

$$J(Y, \hat{Y}) = \frac{2 \cdot |Y \cap \hat{Y}|}{|Y| \cup |\hat{Y}|} = \frac{2 \cdot \sum_{i=1}^{N} y_i \hat{y}_i}{\sum_{i=1}^{N} y_i + \sum_{i=1}^{N} \hat{y}_i} \tag{2.5}$$

where we run over N pixels over ground truth mask $y_i \in Y$ and predicted mask $\hat{y}_i \in \hat{Y}$. In order to optimize model weights, the loss function is differentiable, yielding a gradient

$$\frac{dJ(Y,\hat{Y})}{d\hat{y}_j} = 2\left[\frac{y_j(\sum_{i=1}^{N} y_i^2 + \sum_{i=1}^{N} \hat{y}_i{}^2) - 2\hat{y}_j(\sum_{i=1}^{N} y_i\hat{y}_i)}{(\sum_{i=1}^{N} y_i + \sum_{i=1}^{N} \hat{y}_i)^2}\right] \tag{2.6}$$

with respect to the predicted pixel j.

Chapter 3

Proposed segmentation model

The classic approach of segmentation models is to use a pre-trained decoder on a large image classification dataset, like ImageNet [3], and to add a decoder network that aims to decode latent space information to a segmentation mask.

As part of our work, we have experimented with MobileVit [8] and ResNet [25] encoder models. From each model 4 feature maps were used, and corresponding feature map shapes are presented in the Table 4.1. The last feature map from the encoder model is used as a bottleneck of the model and is passed through the first layer of the decoder network. The other three feature maps are used in the later layers to pass higher-level information of an original image. These feature maps represent so-called skip-connection and are combined with lower-level features from a decoder to pass to the later decoder stages.

Decoder model consists of two Transformer layers, three Attention-Convolution layers and a final convolution classification head. Decoder input $x_0^{encoder} \in \mathbb{R}^{c_0 \times h_0 \times w_0}$, where c_0 is a number of channels at level 0, h_0 is a height of the feature map on level 0, and w_0 is a width of the feature map, is flattened to $x_0^{encoder} \in \mathbb{R}^{c_0 \times h_0 \cdot w_0}$ and passed through Transformer blocks. Output feature maps are back reshaped to the original shape $x_1^{decoder} \in \mathbb{R}^{c_0 \times h_0 \times w_0}$, and concatenated with the higher-level encoder feature map $x_1^{encoder} \in \mathbb{R}^{c_1 \times h_1 \times w_1}$.

$x_1^{decoder} \in \mathbb{R}^{c_0 \times h_0 \times w_0}$ is bilinearly upsampled, to match the shape of the encoder feature map, $x_1^{encoder} \in \mathbb{R}^{c_1 \times h_1 \times w_1}$. These two samples are then combined by the following formulas

$$x = ReLU(BatchNorm(Conv(x_1^{encoder}))) \tag{3.1}$$

$$\begin{aligned} y &= AvgPool(x_1^{decoder}) \\ y &= Sigmoid(BatchNorm(Conv(y))) \end{aligned} \tag{3.2}$$

$$z = x \cdot y$$

$$x_2^{decoder} = z + x_1^{decoder} \tag{3.39}$$

All the following two attention-convolution blocks are computed in the same manner as in the above formulas. Output $x_4^{decoder}$ is then passed to a convolution-classification head, that bilinearly upsamples feature map to an original image shape, and builds a separate channel for each segmentation class. Visualization of the encoder-decoder model can be seen on the Figure 3.1.

Transformer architecture on the lower-level features gives a model feature-level global information about the potential regions of a certain class. Whereas moving up to the classification head, convolution layers make use of the previously learned global information and try to build a local region representation, using feature maps from the corresponding level of an encoder model.

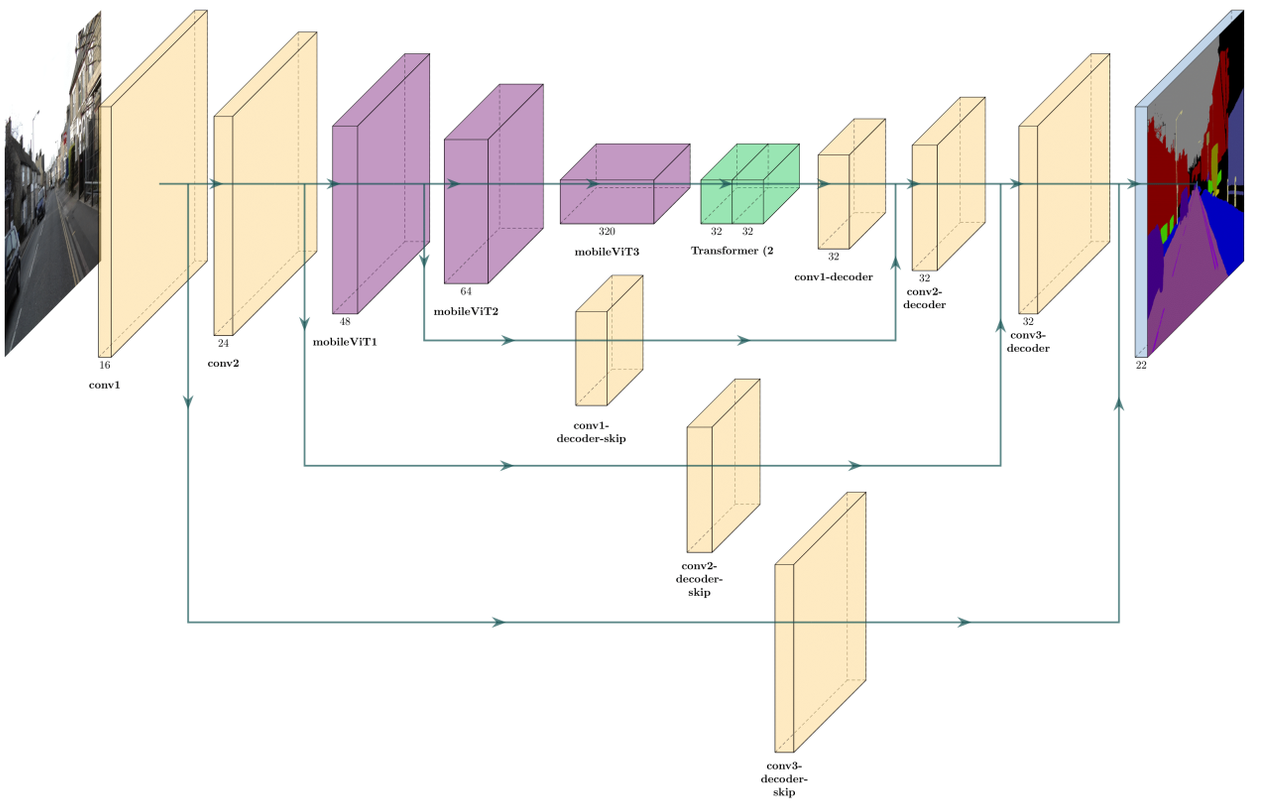

Figure 3.1: U-Net shape segmentation model architecture, with MobileViT pretrained encoder and proposed Transformer based decoder architecture with Attention-Convolutions layers

Chapter 4

Experiments

This section describes the training settings of all the experiments.

4.1 Dataset

We have experimented with PASCAL VOC2012 dataset [26] (sample batch is presented on Figure 2.1). 1464 samples in training set and 1449 in validation set. Each image was reshaped to a size $(3, 480, 480)$, and normalized with mean $(0.485, 0.456, 0.406)$ and standard deviation $(0.229, 0.224, 0.225)$ of ImageNet dataset [3].

4.2 Optimization

Each experiment was conducted using Adam[27] optimizer with cosine annealing learning rate [28] according to the formula.

$$\eta_t = \eta_{min} + \frac{1}{2}(\eta_{max} - \eta_{min})(1 + \cos{(\frac{T_{curr}}{T_{max}}\pi)} \tag{4.1}$$

where η_{max} is a base learning rate of $9e^{-4}$, and η_{min} - minimum learning rate was set to $9e^{-5}$. T_{curr} and T_{max} are a current number of epochs and maximum number of epochs, respectively.

4.3 Hardware

We have trained our models on one Tesla V100-SXM2-16GB GPU. All models were trained for 3000 steps. MobileViT-based models were trained for up to 1 hour and ResNet-based models were trained for up to 2 hours.

Model runtime performance stated in Tables 5.1 and 5.2 was measured on Tesla V100-SXM2-16GB GPU and Intel Xeon CPU (2.00GHz).

4.4 Model configurations

Our experiments were conducted using two sets of models: MobileVit [8] and ResNet [25]. Depending on the model variant different numbers of channels were used. These channels were passed through later model blocks and were passed as a skip-connection to decoder-model blocks. These channel shapes are presented in Table 4.1.

	# Channels
MobileViT_XXS	[24, 48, 64, 320]
MobileViT_XS	[48, 64, 80, 384]
MobileViT_S	[64, 96, 128, 640]
ResNet18	[64, 128, 256, 512]
ResNet50	[256, 512, 1024, 2048]
ResNet101	[256, 512, 1024, 2048]

Table 4.1: Number of channels a model outputs on each model block.

The model is comprised of two Transformer blocks. Each block uses d_model of 225, n_heads of 5, $d\_feedforward$ of 256, and a $dropout$ of 0.2.

Extensive parameter search is a part of out future work.

Chapter 5

Results

5.1 Metric performance

On PASCAL VOC 2012 dataset, the largest encoder-models of their class (MobileViT_S and ResNet-101), combined with the Transformer based decoder with Attention-Convolution layers, performed the best in therms of Jaccard index metric performance - 61% and 63%, respectively. If we make encoder to encoder comparison, our proposed decoder outperforms DeepLabV3 decoder from 1% to 6% for MobileViT encoder and from 2% to 7% for ResNet encoder. Being precise, our decoder combined with MobileViT_XXS outperforms by 1%, with MobileViT_XS by 4%, and MobileViT_S by 6%. Same performance improvement can be seen in the case of Residual Networks. Our decoder architecture outperforms DeepLabV3 decoder combined with ResNet-18 by 4%, ResNet-50 by 2%, and ResNet-101 by 7%.

5.2 Inference speed

In terms of inference speed, there is a correlation between number of parameters and frames per second (FPS) speed. MobileVit_XXS with our decoder slightly underperforms MobileViT_XXS with DeepLabV3, with a use of just-in-time (JIT) compiler, on both GPU and CPU accelerators. The opposite situation can be seen in the case of ResNet models, inference speed of ResNet-18 with our decoder performs almost two times faster than corresponding model with DeepLabV3 decoder.

This can be explained by the time complexity of the operations, Multi-Head Attention block time complexity is $O(n^2 \cdot d + n \cdot d^2)$, where n is a sequence length (length of image patches), d - depth. Convolution complexity is $O(n \cdot d^2)$. Therefore, when the difference in number of paremeters between convolution-based decoder and transformer-based decoder is small, less than a million parameters in a case of MobileViT_XXS, time complexity of the opperetions is crutial. On

the other hand, when the difference is counted as number of millions, as in the case of ResNet-18 (the difference is four million parameters), time complexity of separate model blocks plays less important role.

Detailed summarization of tested models is present on Tables 5.1 and 5.2, and the predicted segmentation masks can be seen on the Figures 5.1 and 5.2

Table 5.1: Results obtained with MobileViT encoder models on Pascal VOC 2012 validation set. Results include model performance on Jaccard index metric as well as inference speed, measured in frames per second. Inference speed was measured using GPU and CPU accelerators, with default and optimized (JIT) compiler for PyTorch models.

	# Params	Jaccard index	FPS	Batch size	Accelerator	JIT
MobileViT_XXS + DeepLabV3	2.4m	50% ±0.2%	3.44	2	CPU	False
			3.49	2	CPU	True
			85.57	8	GPU	False
			96.75	8	GPU	True
MobileViT_XS + DeepLabV3	3.5m	53% ±0.2%	1.90	2	CPU	False
			1.95	2	CPU	True
			86.84	8	GPU	False
			87.66	8	GPU	True
MobileViT_S + DeepLabV3	7.5m	55% ±0.2%	1.47	2	CPU	False
			1.51	2	CPU	True
			80.24	8	GPU	False
			81.36	8	GPU	True
MobileViT_XXS + Transformer	1.7m	51% ±0.5%	3.47	2	CPU	False
			3.43	2	CPU	True
			94.95	8	GPU	False
			96.23	8	GPU	True
MobileViT_XS + Transformer	2.7m	57% ±0.1%	1.94	2	CPU	False
			1.99	2	CPU	True
			86.01	8	GPU	False
			86.38	8	GPU	True
MobileViT_S + Transformer	5.7m	**61% ±0.2%**	1.52	2	CPU	False
			1.51	2	CPU	True
			79.62	8	GPU	False
			84.07	8	GPU	True

Table 5.2: Results obtained with ResNet encoder models on Pascal VOC 2012 validation set. Results include model performance on Jaccard index metric as well as inference speed, measured in frames per second. Inference speed was measured using GPU and CPU accelerators, with default and optimized (JIT) compiler for PyTorch models.

	# Params	Jaccard index	FPS	Batch size	Accelerator	JIT
ResNet18 + DeepLabV3	16m	46% ±0.3%	0.51	2	CPU	False
			0.49	2	CPU	True
			46.38	8	GPU	False
			51.69	8	GPU	True
ResNet50 + DeepLabV3	39m	55% ±0.3%	0.21	2	CPU	False
			0.20	2	CPU	True
			25.87	8	GPU	False
			26.08	8	GPU	True
ResNet101 + DeepLabV3	58m	56% ±0.4%	0.13	2	CPU	False
			0.14	2	CPU	True
			18.94	8	GPU	False
			19.04	8	GPU	True
ResNet18 + Transformer	12m	50% ±0.1%	**3.13**	2	CPU	False
			3.09	2	CPU	True
			102.63	8	GPU	False
			81.67	8	GPU	True
ResNet50 + Transformer	25m	57% ±0.2%	1.21	2	CPU	False
			1.25	2	CPU	True
			73.23	8	GPU	False
			68.44	8	GPU	True
ResNet101 + Transformer	44m	**63% ±0.2%**	0.74	2	CPU	False
			0.75	2	CPU	True
			57.97	8	GPU	False
			56.44	8	GPU	True

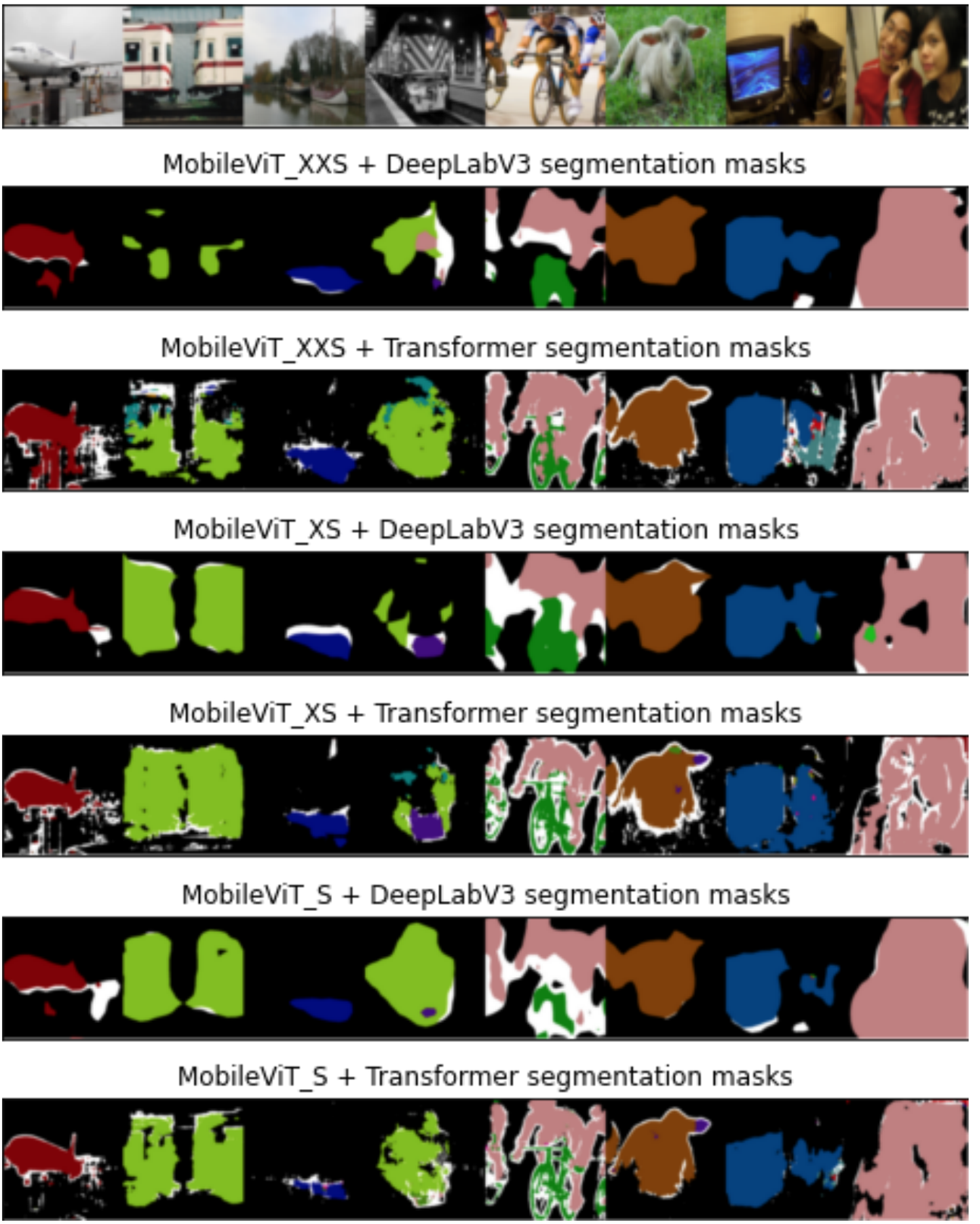

Figure 5.1: Visualization of segmentation maps predicted by DeepLabV3 and Transformer decoders with MobileViT-XXS, MobileViT-XS, MobileViT-S backbones.

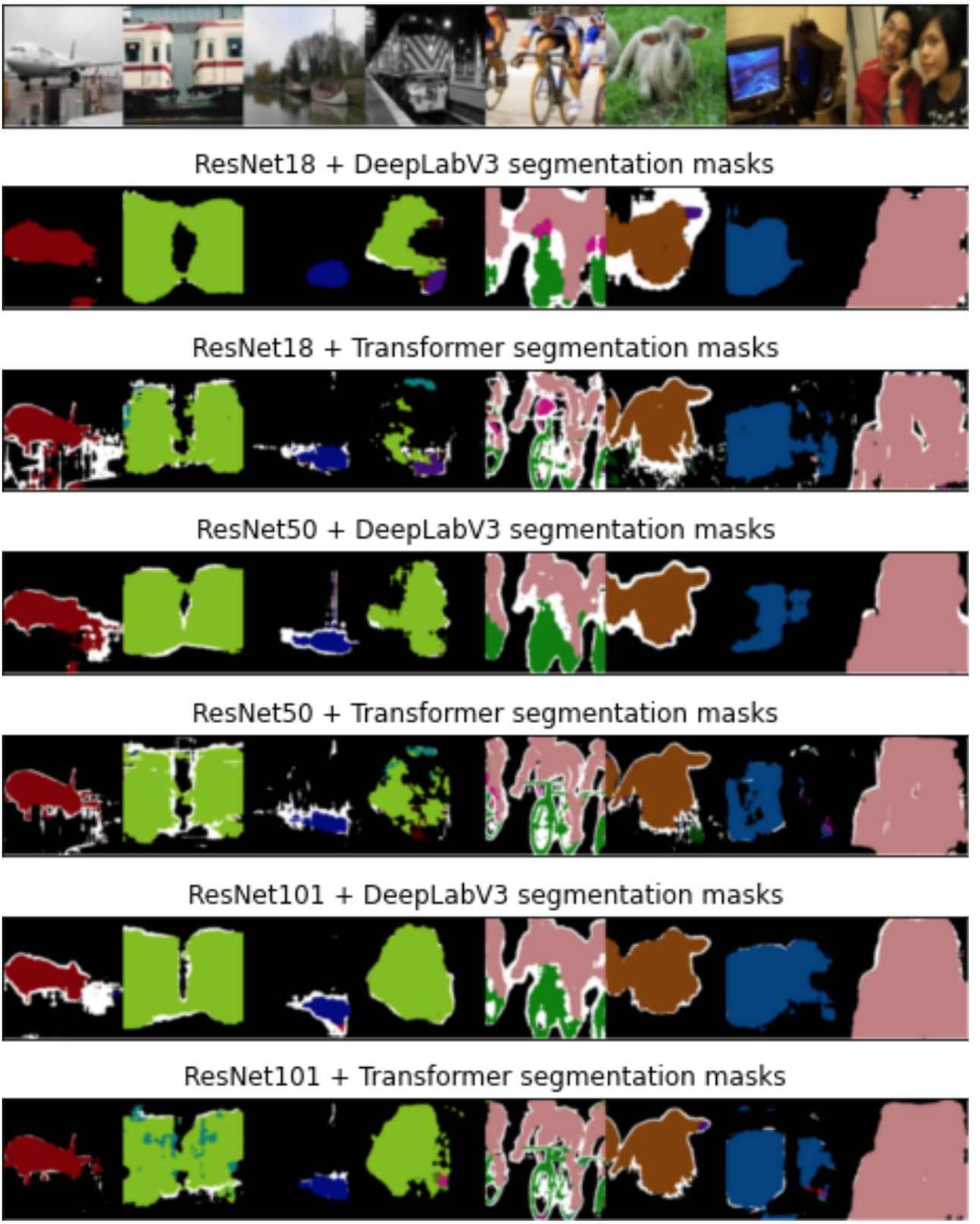

Figure 5.2: Visualization of segmentation maps predicted by DeepLabV3 and Transformer decoders with ResNet-18, ResNet-50, ResNet-101 backbones.

Conclusion

In this work we have presented a model that efficiently balances between local representations obtained by convolution blocks and a global representations obtained by transformer blocks. Proposed model outperforms, previously, standard decoder architecture DeepLabV3 by at least 1% Jaccard index with smaller number of parameters. In the best case this improvement is of 7%.

As part of our future work we plan to experiment with (1) MS COCO dataset pretraining [29] (2) hyperparameters search

References

[1] Liang-Chieh Chen, George Papandreou, Florian Schroff, and Hartwig Adam. Rethinking atrous convolution for semantic image segmentation. *arXiv preprint arXiv:1706.05587*, 2017.

[2] Alex Krizhevsky. One weird trick for parallelizing convolutional neural networks. *CoRR*, abs/1404.5997, 2014.

[3] 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*, pages 248–255. Ieee, 2009.

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

[5] Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*, 2017.

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

[7] Andrew Howard, Mark Sandler, Grace Chu, Liang-Chieh Chen, Bo Chen, Mingxing Tan, Weijun Wang, Yukun Zhu, Ruoming Pang, Vijay Vasudevan, et al. Searching for mobilenetv3. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1314–1324, 2019.

[8] Sachin Mehta and Mohammad Rastegari. Mobilevit: light-weight, general-purpose, and mobile-friendly vision transformer. *arXiv preprint arXiv:2110.02178*, 2021.

[9] Clement Farabet, Camille Couprie, Laurent Najman, and Yann LeCun. Learning hierarchical features for scene labeling. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(8):1915–1929, 2013.

[10] 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*, pages 3431–3440, 2015.

[11] Vijay Badrinarayanan, Alex Kendall, and Roberto Cipolla. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(12):2481–2495, 2017.

[12] Guosheng Lin, Anton Milan, Chunhua Shen, and Ian Reid. Refinenet: Multi-path refinement networks for high-resolution semantic segmentation. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5168–5177, 2017.

[13] Tobias Pohlen, Alexander Hermans, Markus Mathias, and Bastian Leibe. Full-resolution residual networks for semantic segmentation in street scenes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4151–4160, 2017.

[14] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE transactions on pattern analysis and machine intelligence*, 40(4):834–848, 2017.

[15] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. Semantic image segmentation with deep convolutional nets and fully connected crfs. *arXiv preprint arXiv:1412.7062*, 2014.

[16] Fisher Yu and Vladlen Koltun. Multi-scale context aggregation by dilated convolutions. *arXiv preprint arXiv:1511.07122*, 2015.

[17] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10012–10022, 2021.

[18] Zhe Chen, Yuchen Duan, Wenhai Wang, Junjun He, Tong Lu, Jifeng Dai, and Yu Qiao. Vision transformer adapter for dense predictions. *arXiv preprint arXiv:2205.08534*, 2022.

[19] Haotian Yan, Chuang Zhang, and Ming Wu. Lawin transformer: Improving semantic segmentation transformer with multi-scale representations via large window attention. *arXiv preprint arXiv:2201.01615*, 2022.

[20] Robin Strudel, Ricardo Garcia, Ivan Laptev, and Cordelia Schmid. Segmenter: Transformer for semantic segmentation. *arXiv preprint arXiv:2105.05633*, 2021.

[21] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[22] Wei Li, Xing Wang, Xin Xia, Jie Wu, Xuefeng Xiao, Min Zheng, and Shiping Wen. Sepvit: Separable vision transformer. *arXiv preprint arXiv:2203.15380*, 2022.

[23] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. In *International Conference on Machine Learning*, pages 10347–10357. PMLR, 2021.

[24] Fausto Milletari, Nassir Navab, and Seyed-Ahmad Ahmadi. V-net: Fully convolutional neural networks for volumetric medical image segmentation. In *2016 fourth international conference on 3D vision (3DV)*, pages 565–571. IEEE, 2016.

[25] 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*, pages 770–778, 2016.

[26] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results. http://www.pascal-network.org/challenges/VOC/voc2012/workshop/index.html.

[27] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[28] Ilya Loshchilov and Frank Hutter. Sgdr: Stochastic gradient descent with warm restarts. *arXiv preprint arXiv:1608.03983*, 2016.

[29] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014.