Ryan Ott
14862565
ryan.ott@student.uva.nl

1 Transfer Learning

1.1 Comparing Models

1.1.a Inference Speed per Image against Accuracy and Number of Parameters

With a Pearson correlation coefficient of 0.85 there seems to be a strong positive linear correlation between the top-1 accuracy and the inference speed. The ViT-B/32 model performs best overall with a higher accuracy for its inference speed than the trend suggests. Shown in Figure 1.

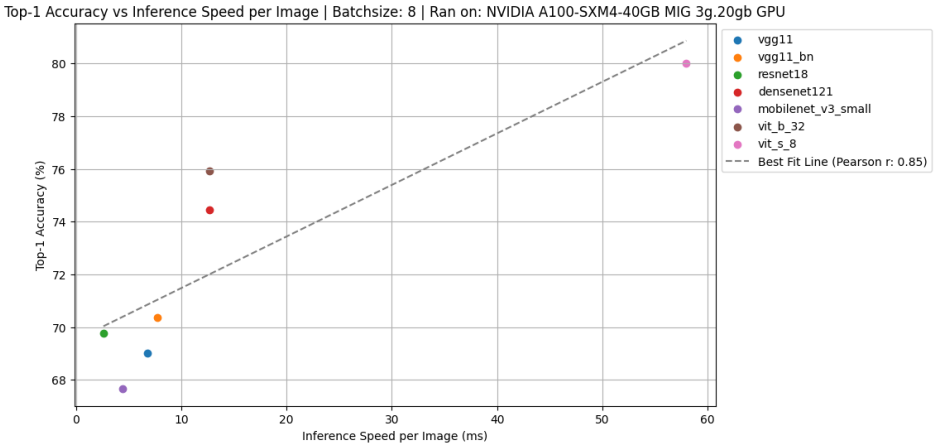

Figure 1: Top-1 Accuracy vs Inference Speed per Image

For inference speed against number of trainable parameters, we see no correlation. The number of trainable parameters should only play a role when the model is being trained, not during inferencing, so this result matches our expectations. Shown in Figure 2.

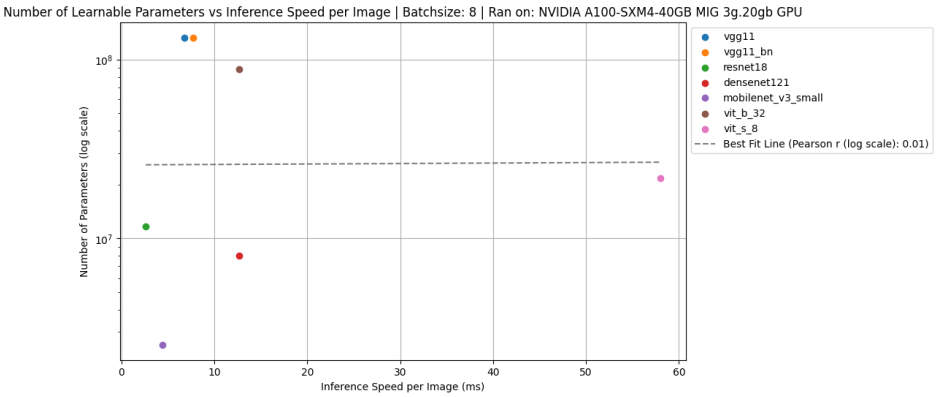

Figure 2: Number of Trainable Parameters vs Inference Speed per Image

1.1.b Inference Speed per Image with and without torch.no_grad()

With torch.no_grad() the inference speed should be faster because gradients are not computed and stored for the backward pass within the context manager's scope, thus requiring less operations and saving compute. This is confirmed by the results, the inference speed is slightly faster with torch.no_grad() across the models. Shown in Figure 3.

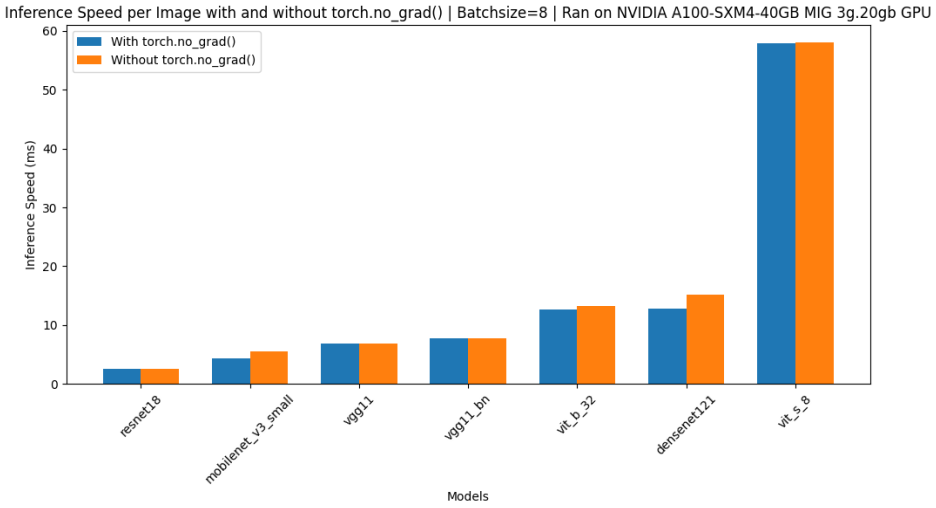

Figure 3: Inference Speed per Image with and without torch.no_grad()

1.1.c vRAM Usage with and without torch.no_grad()

Like with the inference speed, the vRAM usage should be lower with torch.no_grad() because gradients are not stored for the backward pass when performing tensor operations, freeing up memory. Much more considerably than the inference speed, the vRAM usage is lower when using torch.no_grad() as seen in the plot. Shown in Figure 4.

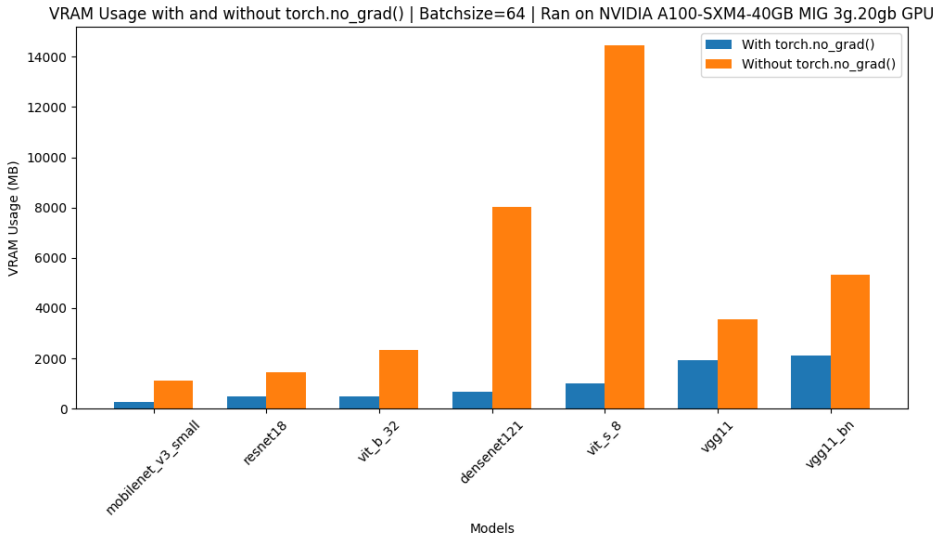

Figure 4: vRAM Usage with and without torch.no_grad()

1.2 Fine-tuning

1.2.a Retraining ResNet-18's Fully Connected Layer

Using the specifications and default hyperparameters from the assignment, the retrained model achieved a test accuracy of 0.5855 in CIFAR-100.

1.2.b Increasing Model Performance using Data Augmentation

By applying `torchvision.transforms.RandomHorizontalFlip(p=0.5)` around half of the images in the training set are flipped horizontally. This should increase the model's performance because it is being trained on more varied data, thus making it more robust to different inputs. The model achieved a test accuracy of 0.5927, a small improvement.

1.2.c Last vs First Convolutional Layer

The first convolutional layers tend to capture lower-level features like edges, corners or textures, while the last layers capture higher-level features like shapes and objects in a larger receptive field. As such, better performance should be achievable by fine-tuning the last layers (along with the classifier layers) because they are more specialized to the downstream task than the first layers.

2 Visual Prompting

2.1 CLIP Baseline

2.1.a Top-1 Accuracy on CIFAR-10 and CIFAR-100 with CLIP ViT-B/32 Backbone

Dataset	Train Accuracy (%)	Test Accuracy (%)
CIFAR-10	88.726	88.940
CIFAR-100	63.564	63.150

Table 1: Zero-shot CLIP Top-1 Accuracy on CIFAR-10 and CIFAR-100

2.1.b Prompting CLIP for a New Classification Task

Prompt: *"This is an image mostly coloured ___"*
Labels: *"red", "green", "blue"*

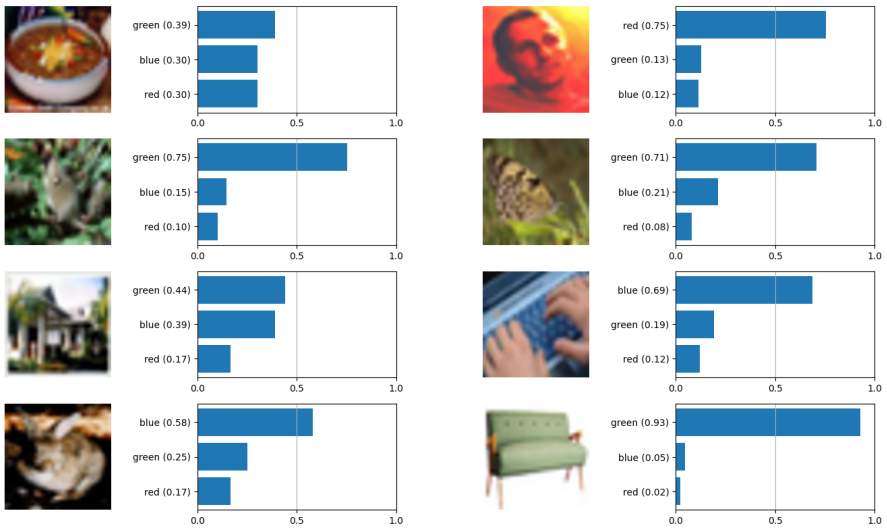

Figure 5: Primary Colour Identification

We see in Figure 5 that the model is able to identify the primary colour of the image with a high degree of accuracy. For images with a distinct primary colour, the model is able to identify it with high confidence, and for images with no clear distinct primary colour, it returns failry spread out probabilities. The contrastive learning approach of CLIP allows it to generalize well to new tasks using prompting.

Prompt: *"This image displays an object that is ___"*
Labels: *"human-made", "from-nature"*
Here too the model performs quite well, considering the task is slightly more abstract than identifying just primary colours. For clearly human-made objects like a computer keyboard or a couch, the model is able to identify it with high confidence. It does however mislabel some examples, like the image of the bunny being labelled as human-made. The more even spread of probabilities for the bunny image suggests that the model is not very confident in its prediction. See Figure 6.

It therefore seems that CLIP is able to generalize well to new tasks using prompting, but it is not perfect and can make mistakes when the task is more abstract. This approach to zero-shot learning is very powerful

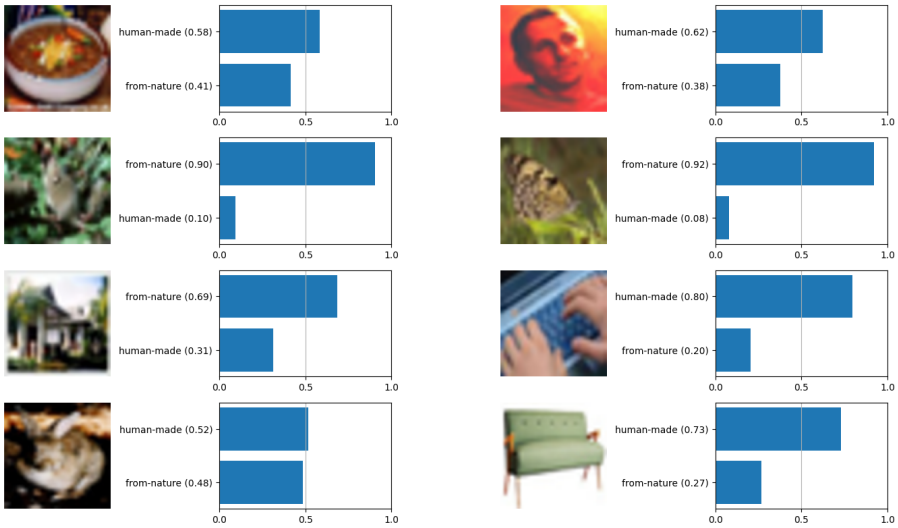

Figure 6: Distinguishing Human-made and Natural Objects

because it allows us to perform a wide range of tasks without having to train a new model for each task. This is especially useful when the task is not very complex and does not require a lot of training data, because training a new model for each task would be very inefficient.

2.2 Effect of Prompt Types on Classification Accuracy

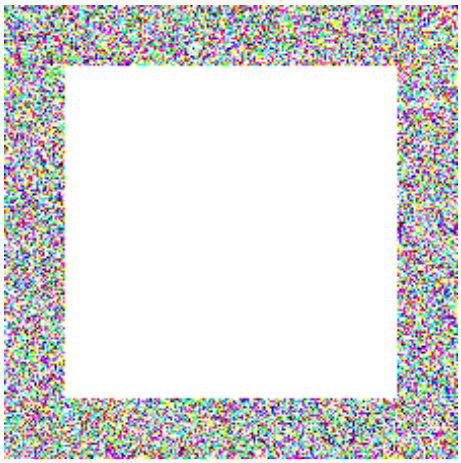

Figure 7: Fixed Patch Visual Prompt (Single Pixel Top-Left)

Figure 8: Padding Visual Prompt

Dataset	Fixed Patch Accuracy (%)	Padding Accuracy (%)	Baseline Accuracy (%)
CIFAR-10	89.570	92.740	88.940
CIFAR-100	64.490	71.390	63.150

Table 2: CLIP Top-1 Accuracy on CIFAR-10 and CIFAR-100's Test with Different Prompt Types Compared to Baseline

Comparing these results obtained using visual prompts on the CIFAR-10 and 100 test set to the baseline results in seen in Table 4, we see that the accuracy is higher for both prompt types, with the padding prompt type performing significantly better than the fixed patch prompt type. This is likely because the padding prompt type has a lot more trainable parameters (69840 for padding size 30) than the fixed patch prompt type (3 for patch size 1, one per channel), and thus is able adapt better to the task. Espeically for the CIFAR-100 dataset this improvement is noticeable. likely due to the fact that the CIFAR-100 dataset is more complex than the CIFAR-10 dataset, and thus requires more trainable parameters to perform well.

2.3 Deep Prompts

From Figure 3 in the assignment PDF we see how we can inject a learnable deep prompt into the CLIP model. In the diagram it is appended to the beginning of the sequence that is fed into a Transformer block. I initially accidentally appended it to the end of the sequence, but then chose to compare the two approaches which was quite interesting to see. I also investigated the effect of different numbers of deep prompts (1, 4 & 8) as well as the injection layer. The results are shown in Table 3, as well as a comparison between the best visual prompt accuracies and baseline readings in Table 4.

Dataset	Injection Layer	Prompts	Front Acc (%)	Back Acc (%)
CIFAR-10	0	1	95.320	**95.480**
		4	**96.310**	94.970
		8	**96.590**	94.670
	6	1	94.630	**94.820**
		4	**95.820**	94.460
		8	**96.000**	94.230
	11	1	91.170	**92.440**
		4	91.180	**92.350**
		8	89.250	**92.190**
CIFAR-100	0	1	75.900	**76.270**
		4	**78.430**	75.780
		8	**79.980**	74.860
	6	1	73.720	**74.290**
		4	**76.950**	74.010
		8	**78.110**	73.560
	11	1	**69.620**	68.390
		4	**69.610**	68.000
		8	**69.640**	67.670

Table 3: Image Classification Performance with CLIP using Different Numbers of Prompts, Injected at Different Layers and Appended to the Front or Back of the Sequence

Injecting prompts at early layers yields higher classification performance across the board than at later layers. This could be because it allows the model to learn more specialized lower-level features that are more useful for the classification task at hand, although this could differ with dataset too. The difference between injecting at layer 0 and 6 is also less pronounced than between layer 6 and 11, which might hint to the fact that deep prompts are most useful for lower-level to mid-level features.

Increasing the number of prompt generally correlates well with an increase in classification accuracy too, having the largest effect when injecting at early to middle layers on the CIFAR-100 dataset. This could be because the CIFAR-100 dataset is more complex than the CIFAR-10 dataset and the model benefits from having more guidance from the prompts.

Appending the prompts to the front of the sequence does seem to yield slightly better results than appending them to the back of the sequence. Although the attention mechanism of the Transformer should allow the model to learn to attend to the prompts regardless of their position in the sequence, appending our style of prompt (”This is a photo of a {}”) perhaps is better suited to being at the beginning, whereas more of a question or summary style prompt could be better suited to being at the end of the sequence. This is just specularitory and could be interesting for futher investigation.

We see that the best performing deep prompt model outperforms the visual prompt model, resulting in a considerable improvement over the baseline model.

Dataset	Deep Prompt Acc (%)	Visual Prompt Acc (%)	Baseline Acc (%)
CIFAR-10	96.590	92.740	88.940
CIFAR-100	79.980	71.390	63.150

Table 4: CLIP Top-1 Accuracy on CIFAR-10 and CIFAR-100’s Test Set with Different Prompt Types Compared to Baseline

The reason why deep prompts might outperform the visual prompts is that they can interact with the latent representation of the image in the model, allowing more complex interactions between the prompt and the image. This can guide the network better than the visual prompts which only can augment the raw input data.

2.4 Robustness to Noise