Reconnaissance des formes pour l'analyse et l'interprétation d'images

Homework 2-a: Transfer Learning

Students: VU Anh Thu 21322736
LE Thi Minh Nguyet 21401438

Transfer learning is highly useful when we aim to leverage a foundation model pretrained on a versatile, general-purpose dataset to perform a specific task, thereby saving both time and money. In this lab, we implemented a transfer learning framework using the entire VGG16 network, except for its last two layers, with pretrained weights on ImageNet, combined with a linear SVC to perform a classification task on the *15 Scene* dataset. Once this approach proved effective, we experimented with other substitutions for the linear SVC to achieve improvements in performance and training time.

Section 1: VGG16 Architecture

(Q1) The number of parameters of the fully connected layers is: $(7 \times 7 \times 512) \times 4096 + 4096 \times 4096 + 4096 \times 1000 = 123,633,664$. Because the fully connected layers account for the majority of the parameters in the model, the total number of parameters in VGG16 is approximately 124 million.

(Q2) The output of the last layer is a vector of size 1000, where each element represents the probability that the input belongs to one of the 1000 ImageNet classes.

(Q3) We have tested the network with different images and obtained the following results:

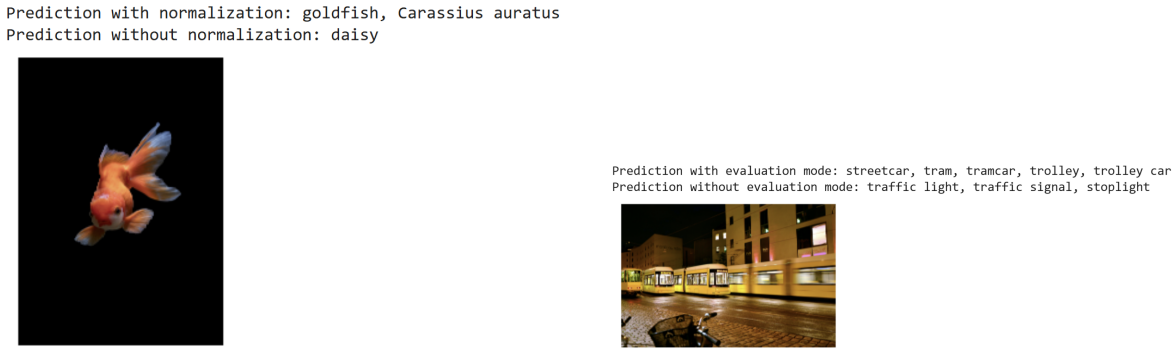
- For images with characteristics similar to those on which VGG16 is trained — such as being object-centric and from similar domains — the network generally produces good predictions.
- For those which are not from similar domain as ImageNet images, sometimes the model might get confused as shown in Figure 1b.

(a) Correct prediction **(b)** Incorrect prediction

Figure 1: Some predictions on different images

a. Since VGG16 is trained on normalized ImageNet images, normalizing the input image before feeding it into the network ensures it undergoes the same transformations as the training images. This practice allows the model to make consistent and reliable predictions. Without normalization, the model may produce significantly inaccurate predictions as shown in Figure 2a.

b. The VGG16 model in Pytorch contains dropout regularization for the first two fully connected layers. Additionally, neural networks with dropout have a different behavior depending on whether we are in inference or learning. Therefore, setting the model to evaluation mode is necessary, as it allows the model to behave consistently during both the training and testing phases. Without this adjustment, the model may fail to produce accurate prediction as shown in Figure 2b.



(a) With and without normalization **(b)** With and without evaluation mode

Figure 2: Necessity of normalization and evaluation mode setting

(Q4) We observe that each activation map captures or focuses on specific patterns in the image, such as

areas above, below, or around the center, as well as corners, edges and light transitions, etc. The early convolutional layers of the network are responsible for detecting these basic patterns which are essential because they provide the foundation for more complex features in deeper layers.

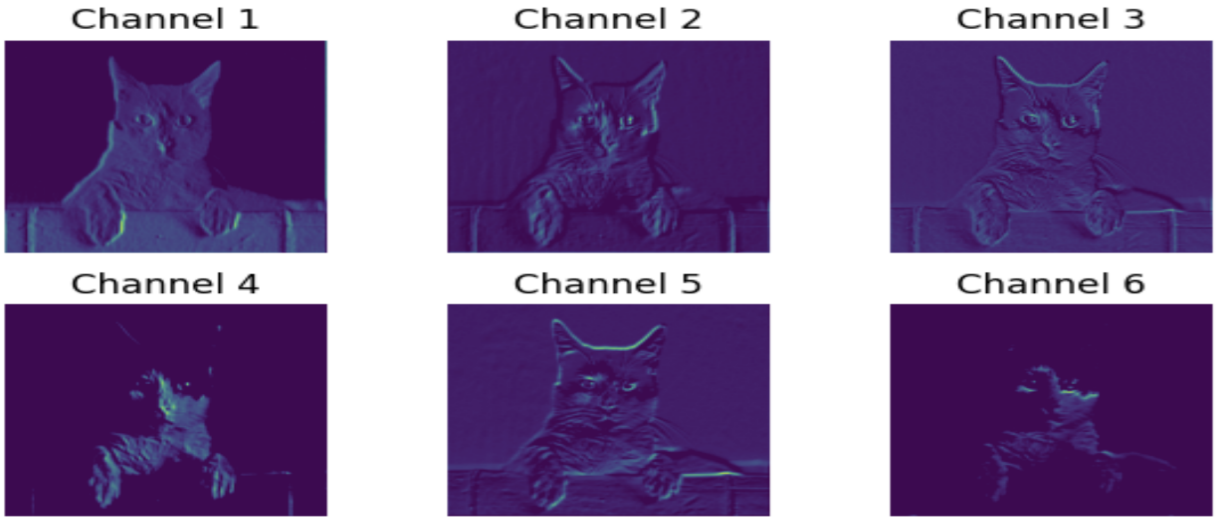


Figure 3: Some activation maps after the first convolution

Section 2: Transfer Learning with VGG16 on 15 Scene

(Q5) The *15 Scene* dataset is much smaller than ImageNet, and with limited data available, training a deep model like VGG16, which has a large number of parameters, could lead to overfitting. This is because there is not sufficient meaningful information for the model to learn from. Additionally, training a complex model like VGG16 from scratch requires significant computational resources and time.

(Q6) VGG16 is pre-trained on ImageNet, which contains more than one million images across many types and domains. Therefore, using VGG16 with pre-trained weights can produce meaningful representation of `15 Scene` images, which are useful for the downstream classification task using SVM.

thus its feature extraction ability can generally applicable to many types of images, including those in the 15 Scene dataset.

(Q7)
- VGG16 is trained on ImageNet-1K, which is primarily object-centric and designed for object classification tasks rather than scene recognition. Therefore, this may limit the model's feature extraction capability when applied to *15 Scene*.
- The images in the *15 Scene* dataset vary in shape and resolution, and most of them have a resolution larger than the input size of VGG16 (224×224 pixels). Although there exist appropriate methods to resize images while preserving their qualities and important spatial information, or if we consider the network's fully connected layers as convolutional ones, the network can still adapt to different resolutions, doing that cannot fully leverage its ability.

- The images in *15 Scene* have only one color channel, while VGG16 is trained on full three-color channel images. This discrepancy may hinder the model's ability to utilize its learned features effectively, as it lacks the expected color information in the input.

(Q8) The features are extracted at the *relu7* layer of VGG16 (the layer just before the classification layer), which serves as a summarizing layer, combining all the important information of the image extracted by the previous layers.

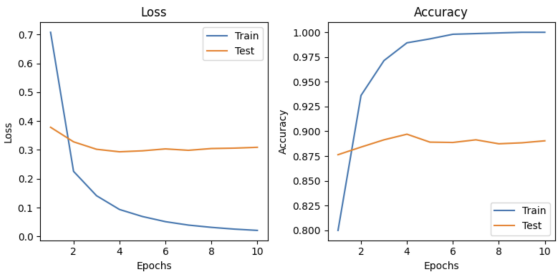
(Q9) In order to address the issue of black-and-white images, we can duplicate the *15 scene* image three times along the channel dimension to convert it into an RGB-like image.

Section 3: Training SVM classifiers

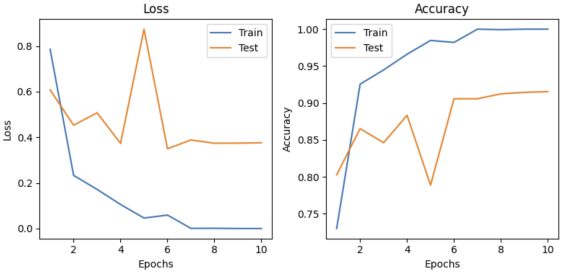
(Q10) It is impossible to just use the neural network because the ouput of VGG16 is a vector of size 1000, while there are only 15 classes in the *15 Scene* dataset.

(Q11) We have implemented three methods and obtained the following results:

- We tuned the hyperparameter `C` using `Optuna` and obtained an optimal value $\texttt{C} = 0.6698$, which achieved an accuracy of 88.9447%, slightly higher than the accuracy that we obtained with $\texttt{C} = 1.0$, which was 88.8107%.
- We implemented the PCA dimensionality reduction algorithm before classification, setting the hyperparameter $\texttt{n\_components} = 0.9$, which limits the relative error of our dimensionality reduction to less than 10%. This approach reduced the size of our vectors from 4096 to 396 and achieved the same performance as the model without dimensionality reduction but with an execution time that was reduced by 35% - 0.469485 seconds compared to 0.72305 seconds. This improvement can be explained by the fact that the representation vectors extracted using VGG16 are likely to contain many high correlated features. Therefore, by using PCA, we can significantly reduce the dimension of the data while losing very little of its important information (in this case, the relative error of the reduction is less than 10%). As a result, the execution time is noticeably reduced while the performnace is maintained.
- We replaced the last layer of VGG16 with a new fully connected layer, trained the network on *15 Scene*, both with and without propagating the gradients to the rest of the network, and obtained the following results:
  - When training without propagating the gradients to the VGG16 part, we achieved the best results at epoch 4, with a test accuracy of 89.7059%, as shown in Figure 4a. This result is higher than the one obtained with Linear SVC, which is 88.8107%.
  - When fine-tuing the whole network with the VGG16 part being updated with a smaller learning rate than the last fully connected layer, we achieved the best results at epoch 10 with a test accuracy of 91.5441%, as shown in Figure 4b. This result is significantly higher than the one obtained with Linear SVC. This considerable improvement is due to the fact that, rather than using the pretrained weights of VGG16 as fixed parameters, we allow them to be updated. By employing a small learning rate, these weights are fine-tuned to better adapt to our specific problem while preserving the valuable features learned during pretraining. This careful adjustment strikes a balance between leveraging prior knowledge and tailoring the model to the new task, leading to superior performance.



(a) Without gradients propagation to VGG16 part
Training using Adam algorithm for 10 epochs with a batch size of 8 and a learning rate of 5×10^{-4}



(b) With gradients propagation to VGG16 part
Training using Adam algorithm for 10 epochs with a batch size of 8, a learning rate of 5×10^{-5} for the VGG16 part and a learning rate of 5×10^{-4} for the last fully connected layer

Figure 4: Replace SVM by a fully connected layer