CS587: Assignment 4

Transfer Learning, Image-Class Saliency Map Visualization

Issued: Wednesday, 24/04/2024
Deadline: Sunday, 12/05/2024, 23:59

Description

The goal of this assignment is two-folded:

- **First Part:** Get familiar with transfer learning, by fine-tuning in a new dataset a Convolutional Neural Network (CNN) that has been trained in another dataset
- **Second Part:** Familiarize with Image-Specific Class Saliency Visualisation

First Part: Transfer Learning

Most deep learning applications use the transfer learning approach, a process that involves fine-tuning a pre-trained model. You start with an existing network, such as AlexNet or GoogLeNet, and feed in new data containing previously unknown classes. After making some tweaks to the network, you can now perform a new task, such as categorizing only dogs or cats instead of 1000 different objects. An example of the concept of transfer learning can be shown in Figure 1.

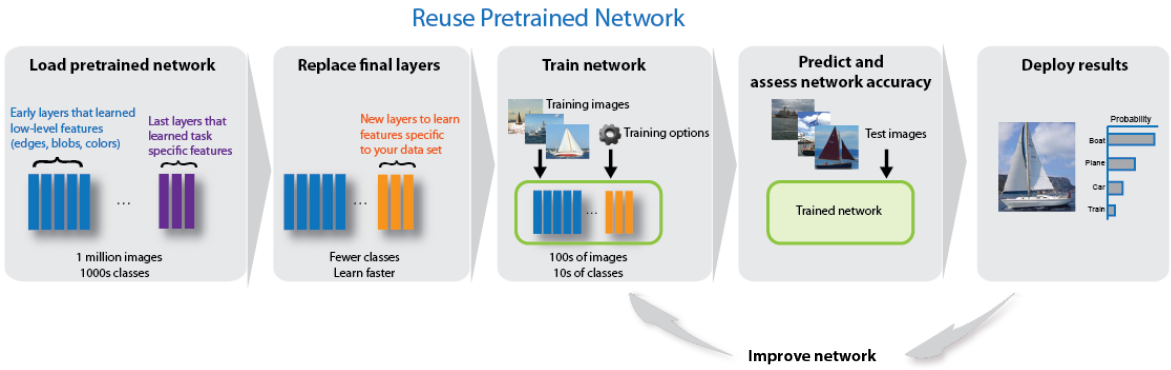

Figure 1: Transfer Learning Concept.

This approach also has the advantage of needing much less data (processing thousands of images rather than millions), so computation time drops to minutes or hours. Transfer learning requires an interface to the internals of the pre-existing network, so it can be surgically modified and enhanced for the new task.

In this assignment, you will be fine-tuning *AlexNet*, a popular CNN architecture, that has been pre-trained on the ImageNet dataset. AlexNet has been trained on over a million images and can classify images into 1000 object categories (such as *keyboard*, *coffee mug*, *pencil*, and many animals). The network has learned rich feature representations for a wide range of images. The network takes an image as input and outputs a label for the object in the image together with the probabilities for each of the object categories.

Your network will be fine-tuned for the task of recognizing art painting categories in a large dataset of art painting images, known as Wikiart. The WikiArt dataset consists of 4000 images of paintings of arbitrary sizes from 10 different styles: *Baroque*, *Realism*, *Expressionism*, etc.

More specifically, you will have to follow the upcoming steps:

1. Download the WikiArt dataset (just run *download_wikiart.py*).
2. Download AlexNet, with the specific weights learned by the network (will be performed automatically during the first run of *assignment_4a.py*).
3. Implement the loss function and the optimization process with the GD algorithm, and calculate the obtained accuracy.
4. Fine-tune the AlexNet network, by subtracting the higher layers of the model. Specifically, in the context of the assignment, you are requested to explore what happens by removing:
   - The last Fully-Connected (FC) layer
   - The last two final FC layers

   and adding a new Fully-Connected layer instead
5. "Monitor" the training loss, as well as the accuracy in the training and validation sets via the creation of summaries in Tensorboard.
6. Comment on the obtained results, and provide respective intuitions:
   - What happens if we remove the last or last two FC layers?
   - Can we justify the overall accuracy (will be small) based on the characteristics of the dataset (size, image style, etc.) regarding the transfer learning formulation?
7. Answer the following theoretical questions briefly in your report:
   - My dataset is small but similar to the original dataset. Should I fine-tune?
   - My dataset is large and similar to the original dataset. Should I fine-tune or train from scratch?
   - My dataset is different from the original. Should I fine-tune?

Second Part: Saliency Visualisation

You will use a pre-trained model, in our case a VGG_{16}, to compute class saliency maps as described in Section 3.1 of [1].

In the attached code, the *vgg16.py* file is a PyTorch implementation of the VGG-16 model. This model has been trained on a subset of the ImageNet database [2], which is used in the ImageNet Large-Scale Visual Recognition Challenge (ILSVRC) [3]. ImageNet is a widely used dataset for the task of image classification, consisting of more than a million images and can be classified into 1000 distinct categories. For example, *keyboard*, *mouse*, *pencil*, and many animals. As a result, the model has learned rich feature representations for a wide range of images.

In this part of the assignment, we will explore and implement the process of creating an Image-Class Saliency Map. A saliency map tells us the degree to which each pixel in the image affects the classification score for that image. To compute it, we compute the gradient of the un-normalized score corresponding to the highest prediction class, (which is a scalar) with respect to the pixels of the image. If the image has shape $(H, W, 3)$ then this gradient will also have shape $(H, W, 3)$, where H stands for image height, W for image width, and 3 for RGB image channels; for each pixel in the image, this gradient tells us the amount by which the classification score will change if the pixel changes by a small amount. To compute the saliency map, we take the absolute value of this gradient, then take

[1] Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. "Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps", ICLR Workshop 2014.

[2] https://www.image-net.org/

[3] Russakovsky, O., Deng, J., Su, H., et al. "ImageNet Large Scale Visual Recognition Challenge." International Journal of Computer Vision (IJCV). Vol 115, Issue 3, 2015, pp. 211-252.

the maximum value over the 3 input channels; the final saliency map thus has shape (H, W), and all entries are non-negative.

Specifically, in order to compute the saliency map of a certain class, you have to follow the upcoming steps:

1. Do a forward pass of the image through the network.
2. Calculate the scores for every class.
3. Enforce the derivative w.r.t. I (i.e. input image) of score vector S at the last layer for all classes except class C to be 0, while for C set it to 1.
4. Back-propagate this derivative until the start.
5. Render the gradients and obtain the Saliency Map.
6. Repeat the process for the rest of the images located in the folder "My-Images".

In the present task of the assignment, you will need to fill-in the missing code concerning $1 - 4$ of the above steps. You will also need to briefly comment on your results. Are the images classified correctly? Is the classification based on the relevant parts of the image?

You can use any available function of *NumPy*, *SciPy*, *PyTorch*, *TensorBoard* required for the aforementioned tasks.

Important Notes

Setup your environment

You can continue using the Anaconda environment from the previous assignments for this task. However, for visualization purposes, you'll also need to install the TensorBoard package. The installation instructions can be found in the relevant Class Tutorial or within the provided assignment code.

Dataset

You do not need to manually download the Digits dataset. It will automatically be set once you run the given source code. Check the provided comments in your code for more details on the Digits dataset.

Submission info

- Create a .pdf (preferred) or .doc file to report the resulting scores, images/figures, and any other comments or description of your work you may need to submit. Do not forget to include your name and ID in the report. Save this file in your working folder.
- After you have finalized your coding and report, remove the datasets folder from the working directory to be submitted.
- Do not forget to include your TensorBoard summaries in your submission.
- Use zip/rar/gz to compress your working folder and rename it to cs587_mylogin_assignment3.xxx in order to submit a single file.
- The submission of your implementation will be via the e-learn platform. Submissions via e-mail will not be accepted.
- You can upload your submission as many times as you need, keeping the same filename.
- Late submissions will be accepted **within two days** of the original deadline. However, **a penalty of 25% per day** will be applied to the final grade for each late day. Assignments submitted more than two days late will not be accepted.

Academic integrity

The assignment is individual. You may discuss with each other in general terms, but the code and the report should be written individually. If you use existing material, be sure to cite the sources in your report. The use of AI language models to produce the report or your implementation is strictly prohibited, and submissions will be verified with an AI detection tool. You may be asked to take an additional short oral examination.

Troubleshooting & Contact

If you encounter any errors or bugs in the provided code, you can report them by sending an email to kaziales@csd.uoc.gr or the course mailing list hy587-list@csd.uoc.gr. Please note that the course mailing list is the preferred channel for general questions about the assignment, as it allows everyone in the class to benefit from the answer.

For any questions of a more personal nature that are not relevant to the entire class, you can reach out to the teaching assistant directly at his email address kaziales@csd.uoc.gr.