HY587: Assignment 3

Transfer Learning, Image-Class Saliency Map Visualization

Issued: 14 May 2020
Deadline: 28 May 2020, 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 is can be shown in Fig. 1
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 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, which consists of 4000 images of paintings of arbitrary sizes from 10 different styles - Baroque, Realism, Expressionism, etc.

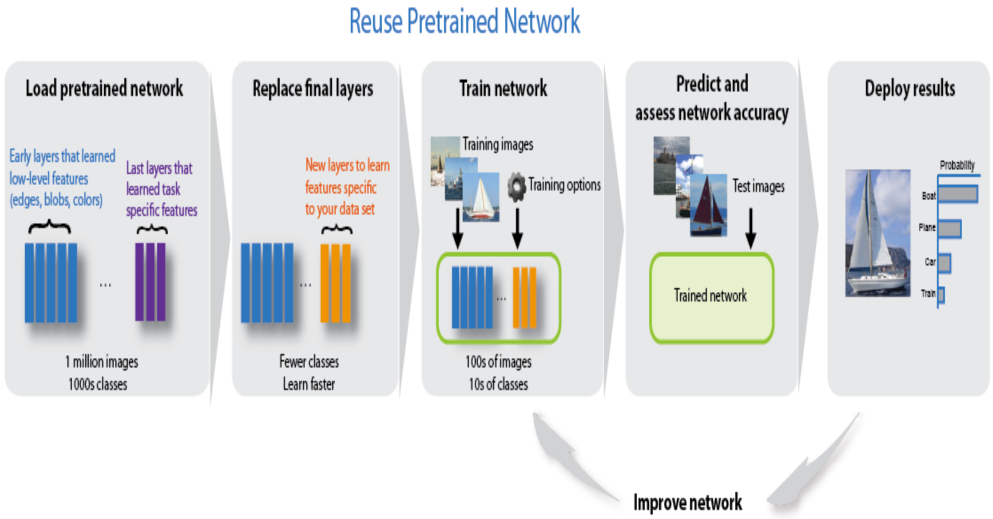

Figure 1: Transfer Learning Concept.

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

- Download the WikiArt dataset (just run download_wikiart.py).
- Download AlexNet, with the specific weights learned by the network (will be perrformed automatically during the first run of assignment_3a.py).
- Implement the loss function, the optimization process with GD algorithm, and the accuracy obtained.
- Fine-tune 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
- ”Monitor” the whole process in the training and validation sets for the loss, the gradients of trainable variables as well as the accuracy via creation of summaries in Tensorboard.
  - Storing of the requested data must be performed every 3 batches (use display_step variable)
  - Use tf.gradients( ) for storing the backpropagated gradients. Search the web and read the documentation on how to use it!
- Comment 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, e.t.c.) regarding the transfer learning formulation?

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 *vgg*16*.py* file is a tensor-flow 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 should 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 correct 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 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 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.

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

[2] `http://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.

You can use any available function of NumPy, SciPy, TensorFlow, Tensor-Board required for the aforementioned tasks.

For the second part we will use Classes and the concepts of Inheritance. So, to understand these concepts please refer to the following links:

1) https://www.w3schools.com/python/python_classes.asp?fbclid=IwAR1myi0ytOlbhWMd0eKxZIkzJch90Kx3dZuIFK9aJcPsNvN0R9HiOF24Toc

2) https://www.pythonforbeginners.com/super/working-python-super-function?fbclid=IwAR14vzi3gdFHYvk_8s49LjA__9_EK1kBJXLOTM_8tWsK1XWOlocjBigcWuo

Important Notes

Setup your Anaconda, Python, TensorFlow environment

Installation guidelines for Anaconda, Python, TensorFlow, TensorBoard, Scikit-learn, (optional: Jupyter).

Python 3.5, TensorFlow 1.0 and Scikit-learn library are required to run the provided code and accomplish your assignment. Read the setup-README.txt file in the main folder of the assignment and the slides of the first tutorial to get info on how to setup your Python environment successfully.

Link to slides of the course tutorials: `https://drive.google.com/open?id=1RanpIS8z72JlguTZAe1HsIfG7U4co61XrY_ 2nWoynOk`

Recommendation: Install Anaconda to get everything installed except TensorFlow+TensorBoard. Then follow the installation guide for TensorFlow to get done with the requirements. `https://www.tensorflow.org/install/`

Useful links: Concerning the first part of the Assignment, you will have to download the following packages in order to process the data: `https://anaconda.org/conda-forge/opencv` `https://anaconda.org/conda-forge/tqdm` `https://anaconda.org/anaconda/scikit-image` `https://anaconda.org/anaconda/pillow` `https://anaconda.org/ulmo/urllib3`

Dataset

Important Note: Before running any script, you must firstly run ”download wikiart.py” file, located in the ”Utilities” folder. You do not need to manually download the WikiArt 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 WikiArt dataset.

Mandatory Note: If you have PILLOW Version 4.1 or later, you should firstly uninstall it an install version 4.0 by executing the following command in Anaconda prompt (in the environment you are working):

pip install Pillow==4.0.0

Submission info

- Create a .pdf 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, login, ID in the report. Save this file in your working folder.
- Use zip/rar/gz to compress your working folderand rename it to cs587 mylogin assignment3.xxx in order to submit a single file.
- You will use the following link of Dropbox request to upload your submission as a SINGLE zip/rar/gz file. You will be requested fiil in your firstname/surname and an email address to upload. https://www.dropbox.com/request/GOIJpbF4WlDJ9sjUkP0u
- You do not need to have a Dropbox account to upload your submission.
- You can upload your submission as many times as you need keeping the same filename.
- Uploading will not be available after the deadline date-time.
- **Important:** Submit only the scripts not the datasets! Also, do not forget to include the Tensorboard event files.
- **Troubleshooting**

  In case you find any errors/bugs in the code please send an email to

  konarak@csd.uoc.gr or the course's mailing list!