Fall 2019: CSI5139F
Assignment 3

Due: Tuesday, November 8th, 2019, 11:00pm in Virtual Campus
University of Ottawa - Université d'Ottawa

Jochen Lang

1 CNNs, Keras API and Optical Flow

This assignment will give you a chance to familiarize yourself with CNNs and with the different techniques for monitoring and controlling the training process in tensorflow. In this assignment, we will look at optical flow, i.e., we want to predict where the imaged point in the first image has moved to in the second image. This means that for each pixel in the first image, we want to estimate a distance in x and a distance in y corresponding to the flow vector.

You must use Keras with the tensorflow backend, i.e., the package `tensorflow.keras`. For this assignment, you may use other tensorflow packages and scikit-learn and scikit-image but *not* other deep learning frameworks, e.g., caffe, pytorch, theano etc.

2 Data Preparation

The data for this assignment are simple test images which show a brick-textured block at various locations in an image. The image-pairs consist of the first or start image and the second image or goal image. The ground truth consists of two additional images containing the flow vectors in the x-component and y-component, respectively. All data is conveniently stored as numpy array files with ending `.npy`. See the attached simple Python code snippet to read and display the images. The data is spread out over different files for training and testing, and for input images and groundtruth flow. Also, there are two datasets: In-plane rotations and translations only; and a mixture of rotations, shearing and stretching combined with translations. The files can be downloaded at the following google drive link.

`https://drive.google.com/drive/folders/1c5vKjoSHvE-6Rq21E6wFByNkRIJYgro6?usp=sharing`

Please note that the data is only available if you log in with your uottawa account to google (your email). You will find the following files:

File	Description
rot_images_train.npy	Goal and target RGB images for rotation training set.
rot_images_test.npy	Goal and target RGB images for rotation test set.
rot_flows_train.npy	Groundtruth for rotation training set.
rot_flows_test.npy	Groundtruth for rotation test set.
mix_images_train.npy	Goal and target RGB images for mixed training set.
mix_images_test.npy	Goal and target RGB images for mixed test set.
mix_flows_train.npy	Groundtruth for mixed training set.
mix_flows_test.npy	Groundtruth for mixed test set.

Note that the images are organized in one large four-dimensional array of dimensions $1800 \times 6 \times 64 \times 64$. The first dimension are the images, the second dimension the channels and the last two are the spatial dimensions. There are six channels which are $[Blue_1, Green_1, Red_1, Blue_2, Green_2, Red_2]$. The groundtruth is organized as $1800 \times 4 \times 64 \times 64$. Again, the first dimension are the images, the second dimension the channels and the last two are the spatial dimensions. There are four channels which are $[flow_x, flow_y, flow_{-x}, flow_{-y}]$. Only the first two channels are useful.

3 Network Design [4]

Using at most seven layers create your own CNN network for regression using the Tensorflow Keras sequential API. Train this simple network on the rotation training set and evaluate it on the rotation test set. You will find that the sequential API in Keras is very simple use. However, you will need to decide on a strategy to create appropriate input to the network, e.g., by somehow concatenating the start and goal image. Finally, you will need to select an appropriate loss function such that the two components of the flow are taken into account. When evaluating the network performance, make sure to clearly state how to measure performance and justify your choice. In particular, there are many ways to compare two vectors, i.e., the groundtruth flow and the estimated flow. Pick a reasonable metric.

4 Transfer Learning [4]

Read the Tensorflow tutorial on transfer learning at
`https://www.tensorflow.org/tutorials/images/transfer_learning`. For this question adapt VGG-16 pre-trained on ImageNet for the task. The pre-trained network is available from `tf.keras.applications.vgg16`. You want to suitably remove some layers (importing with `include_top=False` is a good start but less layers are likely sufficient and will run much faster) and add a regression layer (same or similar as above) at the output. You will need to train the new layers with the weights of the existing VGG layers fixed. Once you have a working regressor for the task, try to improve the flow results by training some layers a bit more (typically the higher-level) layers. Train and test your model als with the rotational and the mixed dataset. On which dataset does your network perform better.

4.1 Non-sequential Model [Bonus]

Typical networks for stereo and optical flow have a non-sequential layout. Often a siamese layout is prefered. For the bonus modify your model from Section 3 such that the start and goal image enter your model separately and the calculated features are combined (either concatenated or combined as different channels), e.g., after two or three layers. You will need to use the Keras Functional API for this purpose. Train and test your new model, trying to improve model performace. Do not increase network depth and still work with a maximum depth of seven layers.

5 Submission

You will need to submit your solution in a Jupyter file, do *not* submit the image data. Make sure you have run all the cells. All text must be embedded in the Jupyter file, I will not look at separately submitted text files. If your Jupyter file needs a local python file to run, please submit it as well. Assignment submission is only though Virtual Campus by the deadline. No late submissions are allowed, you can submit multiple times but only your last submission is kept and marked.