Lecture Machine Vision 2019/20

Institut für Mess- und Regelungstechnik, Karlsruher Institut für Technologie
Dr. Martin Lauer, Christian Kinzig, M.Sc.

Practical Exercises: Deep Learning

These exercises are made to become familiar with some concepts of deep learning. There exist some very useful add-ons for MATLAB that can be used to quickly start learning with neural networks:

- `Neural Network Toolbox` provides a lot of different functions that can be used for creating, modifying, training and classification with neural networks. Further there exist several pretrained network models that can be used as a jump start to deep learning, like:
- `Neural Network Toolbox Model for AlexNet Network` is a neural network trained to recognize 1000 different classes of objects from images. It won the ImageNet Large Visual Recognition Challenge in 2012 with an error of just under 16%.

Your task is to classify different traffic signs. In the folder *traffic_sign_training.zip* you find several folders each with a unique number representing the class of the traffic signs depicted in the images. You can automatically create an image store including labels for training by calling `imageDatastore(image_dir,'IncludeSubfolders',true, 'LabelSource', 'foldernames')` with image_dir containing the path to the folder which holds the labeled image folders.

There is a common deep learning technique called transfer learning. The idea is that most layers perform different preprocessing tasks related to object and shape recognition that are universally applicable. Only the last few layers of a network actually connect abstract shapes and features to a class label. In order to save time and to reduce the amount of training samples it is possible to take a deep neural network that was trained for a different task and retrain only the last few layers for the new classification task. We will use this technique in order to use the pretrained alexnet for our traffic sign classification.

- In order to familiarize yourself with image classification start by loading a pretrained alexnet. Have a look at the network architecture saved in the `Layers` attribute. The last layer includes the labels the net was classified on which can be accessed by `net.Layers(end).ClassNames`. Choose any label, google for pictures of such objects and try using the `classify` function to classify your image. Level of difficulty: easy
- Create a new neural network using all pretrained layers from alexnet except for the last three layers. Replace the last three layers with a new fully connected layer, Level of difficulty: medium

a softmax layer and a classification layer. Use the `splitEachLabel` function to create a training set and a validation set. Train your newly created net and test your classifier by applying the net to the validation set.

- Optimize the training process by creating additional training data through image manipulation of the training set. (e.g. slight rotation, affine transformations, color manipulation, flipping if sensible) Optimize the classification by applying suitable preprocessing steps before training and classification.

  Level of difficulty: hard

  If you are confident in your network, write an email to the exercise supervisor and you can present your optimization at the end of the next exercise. Your net will be tested on images not published on the website.

- `Hint:` the training process needs a lot of resources and time. If possible let the net train on a computer with a good graphics card and/or use the parallel computing toolbox MATLAB addon to accelerate training.