Hands-on Machine Learning Training

Session 7 – Deep Learning

Theoretical Preparation

This session focuses on deep learning. We will steer away from implementing complex networks by hand and instead use the PyTorch framework to setup a network architecture and automatically optimize the parameters without explicitly defining every gradient function ourselves. As an established network structure we will use VGG16 - a 16 layer deep neural network, with a simple structure that is still the basis for many other architectures.

Deep Learning typically requires larger datasets than classical machine learning approaches. In this session, we will use the PascalVOC dataset which provides different forms of annotations for various supervised learning setups. This Session uses the class labels in the set for a multi-label classification, while in the next session you will use the semantic annotation maps for a different learning task.

- **General Deep Learning and VGG16 Architecture**
  - Read through the introduction of the stanford lecture CS231n `http://cs231n.github.io/convolutional-networks/`. Herein, pay attention to the different layer types, in particular the Convolutional Layer (including computations of the feature map sizes) and the Pooling Layer. At the end of the introduction, the idea of Layer Patterns is introduced and VGG16 is discussed as a case example.
  - Zissermann et al. introduce VGG in `https://arxiv.org/pdf/1409.1556.pdf`, read through section 1, 2 and 3 to understand the "how and why" of the network. In Table 1 of the paper, different setups for VGG networks are given. For this session, we will implement the network as specified in column D.
- **PascalVOC 2012 dataset**
  Browse `http://host.robots.ox.ac.uk/pascal/VOC/voc2012/` to obtain an overview of the dataset. Relevant are the Introduction and Data section.
- **PyTorch** To familiarize yourself with the framework, take a look at the extensive documentation:
  - `http://pytorch.org/about/` gives a very high level overview.

- The pyTorch Cheat Sheet gives you an overview over different classes in pyTorch.
- Now, take a look at some of the tutorial examples `http://pytorch.org/tutorials/beginner/pytorch_with_examples.html#nn-module` and `http://pytorch.org/tutorials/beginner/pytorch_with_examples.html#pytorch-custom-nn-modules` to understand how the framework replaces your handwritten code of previous session and modularizes large, potentially complicated networks.

Am I Prepared?

Since this session has quite a number of different sources you might find the following remarks and questions helpful for preparation:

- **General Deep Learning and VGG16 Architecture**
  Do you know the following network layers?: Convolutional layer, Dense/Linear layer, Max-Pooling, Batch-Norm, Dropout
  How is a convolutional layer different from a dense layer?
  What is the influence of padding and how might it affect the network?
  Which layers appear in a VGG network?
  Are there layer-patterns in a VGG net?
- **PascalVOC 2012**
  How many classes appear are present in the dataset?
  How many objects may appear per image?
  Define the classification problem in your own words as accurately as possible.
- **PyTorch Framework**
  How do I write my own module?
  What is the difference between layers appearing in the `__init__()` and `forward()` method of a module?
  What is a tensor?
  How is a tensor for image data shaped (= what are its dimensions)?
  Why do I have wrap (virtually) everything in `Variables`?
  How many non-linearities can you name?