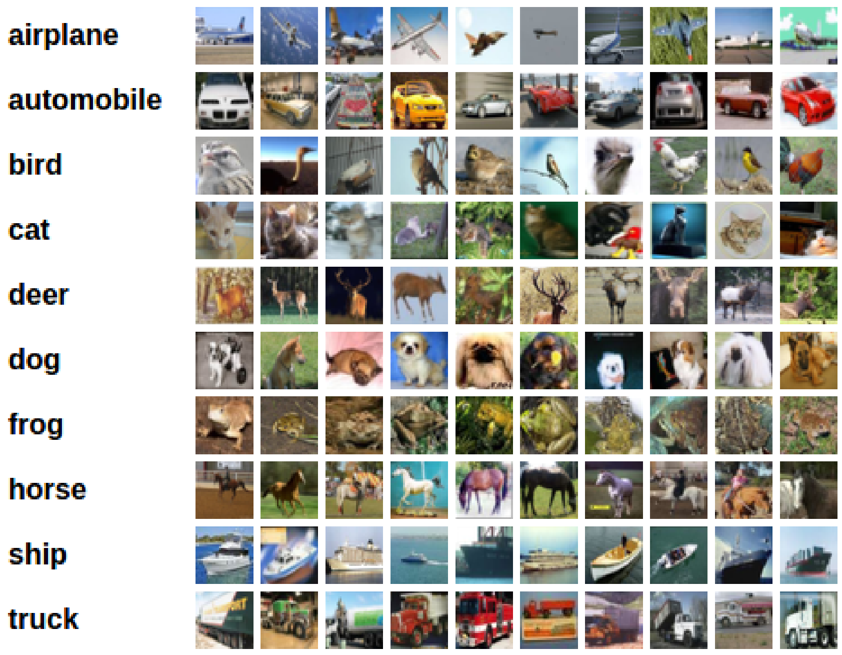
Assignment 10

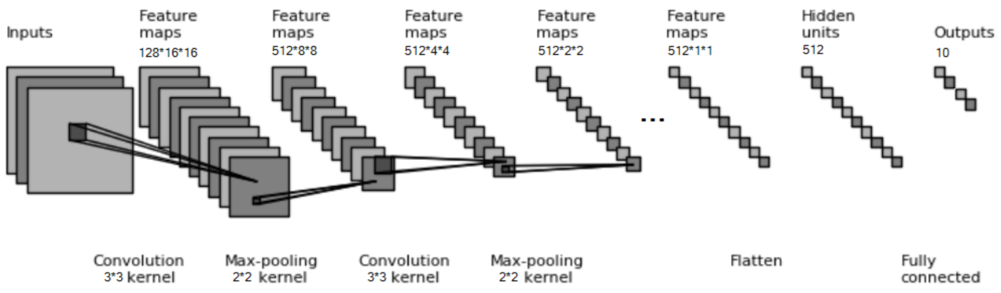
Welcome to the tenth assignment of the lecture *2D Vision and Deep Learning*. **Please read all instructions carefully!** This assignment covers the concepts of batch normalization and early stopping in deep convolutional neural networks.

Submission is due on Monday, February 1th, 2021 at 2pm. Please note that late assignments will receive zero (0) marks, so you are strongly encouraged to start the assignment early. If you have any questions please contact Tim Heußler (`theussle@students.uni-mainz.de`).

Exercise 1 (20 points)**.** The CIFAR-10 dataset is a common dataset consisting of 50000 training images in 32x32 resolution with 10 object classes, namely airplanes, cars, birds, cats, deer, dogs, frogs, horses, ships and trucks (see images below). It is a good starting point to experiment with deep learning approaches. Implement a five-layered convolutional neural network architecture as well as an appropriate loss function using PyTorch to classify the images from the CIFAR-10 dataset.



1. (10 points) Implement a class `ConvNet` that takes 32×32 color images as inputs, has 5 hidden layers with 128, 512, 512, 512, 512 filters, and produces a 10-class classification. The model should be composed of five convolution blocks (see the figure above for the overall architecture). Each block should consist of convolution, max pooling and ReLU operation in that order.



Use 3×3 kernels in all convolutional layers. Set the padding and stride of the convolutional layers so that they maintain the spatial dimensions. Max pooling operations should be done with 2×2 kernels, with a stride of 2. This halves the spatial resolution each time. Stacking these five blocks leads to a $512 \times 1 \times 1$ feature map. Finally, the classification is achieved by a fully connected layer. Train the above model on th CIFAR-10 dataset and report the training and validation accuracies.

2. (2 points) Implement a function `PrintModelSize` which calculates and prints the number of parameters of a neural network. This gives a measure of model capacity. Report the number of parameters for the model implemented in `ConvNet`.

3. (5 points) Extend the `ConvNet` class to `ConvNetBatchNorm` by adding batch normalization to the model. Keep all the other hyperparameters same, but add only batch normalization. In each block, the computations now should be in the order of convolution, batch normalization, pooling, ReLU. Compare the loss curves and accuracy of `ConvNetBatchNorm` using batch normalization to its counterpart `ConvNet` without batch normalization.

4. (3 points) Early stopping is a method to reduce overfitting. Instead of reporting the performance of the final model, early stopping also saves the best model on the validation set during training. Increase the training epochs to 50 in `ConvNet` and `ConvNetBatchNorm`, and compare the best model and latest model on the training set. Due to the randomness, you should train multiple times to verify and observe overfitting and early stopping.