SGN-13000/SGN-13006 Introduction to Pattern Recognition and Machine Learning
Department of Signal Processing, Tampere University of Technology
Joni Kamarainen and Katariina Mahkonen
Exercise 3
ImageNet Classification

To get the exercise points you should show your solutions to the problems to the assistant during an exercise session. During the exercise sessions the assistant is also available for your questions concerning the problems.

1. **ImageNet evaluation** (30 points)

   A "tiny" version of the original ImageNet (see http://image-net.org/) dataset is put into shared linux drive on (*only-5):

   ```
   proffa.cc.tut.fi:/work/sgn_mlintro/
   ```

   You should now have copied at least one of the tiny direction to your local working directory.

   The same directory contains another sub direction, *ILSVRC2014_devkit*, that contains scripts for evaluating visual class detection methods. To evaluate your method, you should output an ascii file of the following form (don't mind about the "bbox" entries):

   ```
   classNum bbox_xmin bbox_ymin bbox_xmax bbox_ymax classNum bbox_xmin ...
   classNum bbox_xmin ...
   ...
   ```

   There are 50,000 validation images and in the detection file there should be 50,000 lines to test against the validation images of the whole dataset. The state-of-the-art results are available at: http://image-net.org/challenges/LSVRC/2014/results. The winning top-5 error in 2015 was (Task 2a - ordered by classification error) 6.66%. Top-5 means that you can give up to 5 "guesses" out of 1,000 possible classes for each test image.

   Tiny-5 – Instead of 1,000 classes let's start with only 5 classes. Run the evaluation:

   ```
   ~:> cd <DATA_DIR>ILSVRC2014_devkit/evaluation
   ~:> nice matlab -nodesktop
   >> demo_eval_clsloc_tiny_5
   ```

   The script is rather slow (keep waiting), but should provide you the result using a "perfect classifier" and by giving only one candidate for every image. There is a corresponding script for evaluating all 1,000 classes: *demo_eval_clsloc* (don't run until you want to test your classifier to beat your lecturer ;-)

   Your task – Make a classifier "imagenet_random.m" that assigns a random class for every validation image and saves results to file *demo-tiny-5-RAND.val.pred.loc.txt*. Then copy the above evaluation script to your local directory, edit its paths variables correctly and re-run for your random classifier results. Verify that the result corresponds to the theoretical limit!

   Task 2: Instead of a single candidate, generate 5 random candidates. Verify that the result corresponds to the theoretical limit!

   Hints: The evaluation script loads a structure from the file "../data/meta_clsloc.mat" - see what it contains and figure out what are the names of the classes included to ImageNet-tiny-5. The correct validation list file numbers are in "../data/ILSVRC2014-tiny-5-filelist.txt" and these are the test images in all following experiments.

2. **ImageNet feature extraction and k-NN classifier** (70 points)

   Using your Matlab script to plot ImageNet-tiny example images write another Matlab script *imagenet_tiny_5_train_feature_extraction.m* that extracts and saves features for all training examples.

   Your script should read every training image, form a single feature vector $\vec{f}_{1\times 192}$ (length=width×height×3 (RGB)). The script should return a training data matrix $\mathbf{P}$ where each line is one image and the output vector $\vec{t}$ where each line is the number of the correct class (remember the synsets from the previous exercise).

Next, write another classifier script,*imagenet_nearestneighbor.m*, that reads every test image, forms a feature vector, computes distance to all stored training examples and assigns a class of the closes match (nearest neighbor classifier). Results should be stored to the file *demo-tiny-5-1NN.val.pred.loc.txt*. Run the evaluation code for this output and check whether it is any better than the random classifier.