UNIVERSITÄT HEIDELBERG
Zukunft. Seit 1386.
IWR Interdisciplinary Center for Scientific Computing

Artificial Intelligence
Wintersemester 17/18
Prof. Björn Ommer
ommer@uni-heidelberg.de

Exercise sheet 10: Exercise 10

Due on *19.01.2018, 2 pm*, 2pm.

Question 1: Image Compression using k-Means clustering (2P)

During the lecture, we learnt an application of k-Means for image compression.

1. Write a function that takes a color image and applies the k-Means algorithm to cluster the colors of each pixel. In this case the feature vectors are three-dimensional, one for each of the three color channels: (R,G,B). For the clustering of the colours you can make use of the k-means function of the *scikit-learn* library. (1P)

2. Apply your function on the image 'grass2.jpg' and display your results using different number of centroids. (1P)

Question 2: Feature Learning and Sample Synthesis via Variational Auto-Encoder (8P)

Last exercise sheet you already implemented a standard Auto-Encoder framework. This time you will implement a Variational Auto-Encoder which extends the normal Auto-Encoder with a generative component, allowing for synthesizing new, unseen samples of your data.

1. Implement a Variational Auto-Encoder framework based on the architecture shown in figure 1. It should support functionalities for training the network, extracting features ($= \mu$) for a given set of images and generating new samples by sampling from the learnt latent space. For computing the VAE loss and the reparametrization trick you can use of the provided code. (3P)

2. Train your network on the *MNIST* dataset using a batchsize of 128 and an ADAM solver with learning rate of 0.001. After $t \in \{1, 5, 10\}$ epochs visualize 10 images plus reconstructions and 10 generated images. (1P)

3. After training your model...

   - Extract features for a random subset ($\sim$ 400 images) of your dataset. (1P)
   - Cluster the subset using the k-means method with $k =$ 'number of classes present in the dataset'. Visualize the cluster centers, respectively their corresponding images. (1P)

- Visualize the learnt data manifold approximated by the extracted features. To do so you can use the provided function *apply_tsne_img*. It computes first a projection of the features into the 2D-plane using the tSNE algorithm. Then at each mapped feature location the corresponding image is visualized. (1P)

4. Comment on the results of (3) and describe your observations. (1P)

Encoder	
Linear	$H \times W \times N_c \rightarrow 512$
Relu	
Linear	$512 \rightarrow 256$
Relu	
Linear	$256 \rightarrow 64$ ($\times 2$: μ and σ)

Decoder	
Linear	$64 \rightarrow 256$
Relu	
Linear	$256 \rightarrow 512$
Relu	
Linear	$512 \rightarrow H \times W \times N_c$
Tanh	

Figure 1: Architecture of the encoder and decoder network. H, W denote the image width and height and N_c denotes to number of image channels.

NOTE:
If you are using the CIP-Pool servers you need to install some libraries in order to use *scikit-learn* and the provided function from Question 2. To do so, please run the following commands with activated ai-environment:

- conda install scikit-learn
- conda install scikit-image

Note: Submit exactly one ZIP file and one PDF file via Moodle before the deadline. The ZIP file should contain your executable code. Make sure that it runs on different operating systems and use relative paths. Non-trivial sections of your code should be explained with short comments, and variables should have self-explanatory names. The PDF file should contain your written code, all figures, explanations and answers to questions. Make sure that plots have informative axis labels, legends and captions.