Assignment 2: Deep learning

This assignment should be carried out using Jupyter Notebooks, TensorFlow 1, TensorBoard, NumPy, Pandas and Matplotlib. ***You are not allowed to use Keras for this assignment!***

Goal

The deep learning assignment consists of two parts. At first you will experiment with fully-connected networks. You are going to build multiple neural networks with varying structures and activation functions for a given dataset.

In the second assignment you are going to build a convolutional neural network for image classification purposes.

Part 1: Fully Connected Layers

The provided dataset contains possible combinations for the game Yathzee. If you don't know the rules for this dice game, please have a look at: `https://en.wikipedia.org/wiki/Yahtzee`.

You will need to build a neural network that is able to predict the label for 5 thrown dice. The following labels are available:

Name	**Description**
3-of-a-kind	Three dice the same.
4-of-a-kind	Four dice the same.
Full-house	Three of one number and two of another.
Small-straight	Four sequential dice (1-2-3-4, 2-3-4-5, or 3-4-5-6)
Large-straight	Five sequential dice (1-2-3-4-5 or 2-3-4-5-6)
Yathzee	All five dice the same
Nothing	None of the above combinations has been thrown.

Goal of the assignment

The overall goal is to experiment with deep learning and find out what gives you the best results. Don't forget to compare the results and write a conclusion!!! Experiments we expect you to carry out:

- Playing around with different networks sizes
  - different amounts of layers
  - different amount of neurons per layer
  - at least 5 different networks with a minimum of 1 hidden layer per network
- Comparison of different activation functions.
  - Sigmoid
  - Tanh
  - ReLu
- Difference with and without dropout.

The notebook should contain:

- For each neural network that you train graphs showing the accuracy and the loss for train set, test set and validation set.
- Your observations and conclusions.
- An export of your best trained network and a way to run this exported model.

Hints

In order to build proper neural networks, keep in mind:

- Convert the labels into one-hot-encoded values.
- Use cross-entropy as loss function for classification.
- Create a proper output layer that uses SoftMax activation.
- Use the accuracy metric to measure your classification performance.
- Avoid overfitting by using dropout, a validation set and proper cross-validation.
- Use Minibatch gradient descent, to speed up the training process.
- Export all the important variables (such as the loss) to TensorBoard, monitor these variables during your testing. Save the images and add them to your notebook with description, observations and conclusion per network.

Export your best model (see `https://www.tensorflow.org/guide/saved_model`) and add a cell to your notebook that loads the model and is able to validate your model, by loading in a dataset from file and feeding it into the network. This cell should show the accuracy of the classifier.

Part 2: Convolutional Neural Networks

For this assignment you are going to use the zipcode-dataset from Introduction to Machine Learning (see blackboard).

There are plenty of examples of how to build convolutional neural networks. We advice you, however, to reuse the code from your first assignment. This time you also need to use convolutional layers and pooling layers.

Goal of the assignment

You should experiment with different network structures and learn how to train and test neural networks. The goal is to understand what is going on in your network and visualizing that as well.

Don't forget to compare the results and write a conclusion!!! Experiments we expect you to carry out:

- Playing around with different networks sizes (different amount of layers, different amount of neurons per layer, at least 3 different networks with a minimum of 2 convolutional layers per network).
- Difference with and without dropout.
- Visualize metrics such as accuracy and cross entropy for both the trainingset and testingset (to prove that you are not overfitting the network).

Export your best model (see `https://www.tensorflow.org/guide/saved_model`) and add a cell to your notebook that loads the model and is able to validate your model, by loading in a dataset from file and feeding it into the network. This cell should show the accuracy of the classifier.

In order to build proper neural networks, keep in mind:

- Use at least 2 convolutional layers in each network.
- Create a proper output layer using SoftMax activation.
- Convert the labels into one-hot-encoded values.
- Use cross-entropy as loss function for classification.
- Avoid overfitting by using dropout, a validation set and proper cross-validation.

- Use Minibatch gradient descent, to speed up the training process.
- Export all the important variables (such as the loss) to TensorBoard, so that you can monitor your training/testing. (or plot them using Matplotlib in your notebooks).

Extra work (additional points for your grade)

- Use other regularization techniques to prevent overfitting.
- Load an existing popular neural network and retrain (transfer learning) for your problem.
- Test with different optimizers.
- Use literature to find best layer size and network structure.
- Use mini batch stochastic gradient descent.
- Use part of the API of TensorFlow to improve your code.
- More ideas are welcome... Please discuss with your lab session teacher.

During the assessment we will discuss your code, the way you dealt with overfitting and the comparison of your networks.