Assignment 4: Autoencoders/LSTM networks

Deep Networks for Machine Learning — Spring 2019

Brief

• Due date: 04 May, 2019 11:59 PM
• Required files: Submit your ipython notebook along with a Report.pdf in a single zip file with name IDnumber_A04.zip.
• Submission: iLearn + hard copy of the report

Overview

The goal of this assignment is to design, implement and apply autoencoders and LSTM networks in Keras

Task 1 Autoencoders and dimensionality reduction (10 points)

The goal of this part to develop an autoencoder for the purpose of dimensionality reduction, while at the same time keeping the error in reconstruction small. You will use the fashio mnist dataset available in Keras for this purpose. The dataset can be imported using the following code instance. In, this assignment you will keep the images but will discard the labels.

```
from keras.datasets import fashion_mnist

(train_images, train_labels), (test_images, test_labels) = fashion_mnist.load_data()
```

Before you do the training or testing in this or the next task, make sure that the pixel values are floating point numbers and are scaled between 0 and 1.

Now train a densely connected autoencoder with one hidden layers (e.g. the kind of structure shown on slide 19 in Week11_01_02.pdf in the lecture nodes) on the training data. You can experiment with having 32 nodes and 64 nodes in this layer with a rectified linear unit as the activation function. In the output layer have a signmoid activation function. Train the autoencoder under squared error reconstruction loss function. The output of the hidden layer are the 'codes' that represent your input images in reduced dimensions.

Now test the networks with the images in the testing data. Report the average reconstruction error for different number of nodes in the hidden layer.

Repeat the same with autoencoders with three hidden layers (e.g. the kind of structure shown on slide 20 in Week11_01_02.pdf in the lecture nodes). Here, too use 32 and 64 as the dimenionality of the codes (that is train and test two networks).

Task 2 Denoising autoencoders (10 points)

Here, you will see how autoendoers can be used for denoising applications. You will use the same data as in Task 1 for training and testing. However, before applying as input you will corrupt the images with 0 mean additive gaussian noise with some standard deviation value. You can experiment with a structure

similar to the slide 12 in Week11_01_04.pdf in the lecture notes. Use 32 and 64 as the dimenionality of the codes. Experiment with the noise standard deviation level of 0.1, 0.2 and 0.3. Note that, in denoising autoencoders, you use clean training images at the output during training.

Corrupt the testing images in similar way as you did during training. Now compare the reconstructed images when noisy testing images are provided as input to the original testing images with no added noise. Plot a few reconstructed examples alongwith the corresponding noiy images and the original images for the three levels of noise. Comment on the quality and report the root mean square error between the reconstructed testing images and clean testing images for the three noise levels and the two architectures with different dimensionality of the reprenstative codes.

Task 3 LSTM networks (15 points)

In this task you will have a chance to work with a subset of the data collected by our lab, here at AUS with an Inertial Measurement Unit (IMU) for human activity understanding. You will be using this data for speed estimation. The data that is given to you are sequences of accelerometer and gyroscope data from an IMU attached to the shoe of three subjects. The subjects are asked to walk/run on a treadmill at 12 different speeds. The data can be downloaded from the following link:

https://drive.google.com/open?id=1pGMALM7S07YbnH5HvDDtML6TMse-y-LL

You can load the data for each of the three subjects using the following code snippent:

```
with open('Subject 1.pickle', 'rb') as f:
    [sequences, ground_truth] = pickle.load(f)
```

The 'seuqnces' tensor has the dimensions n sequences x length of sequence x 6 features. The 6 features has 3 for accelerometer data and 3 for gyroscope data. Each seuqnce is approximately six seconds long. 'The ground_truth' is a vector with values representing the speed of the subjects in km/hr. Note that this speed estimation problem is a regression problem.

You will use the data for subject 1 as training and the data for subject 2 as validation. Build your own LSTM network, you can experiment with different parameters/hyper-parameters. Pick the structure that gives you the best performance.

Now combine the data for subject 1 and subject 2. That is, use the sequences from both of theese subjects as the new training data. Retraing the LSTM network with the best parameters/hyper-parameters from the previous step.

Now test on the sequences from Subject 4. Report the average root mean squated error for speed estimation, both on the training data and testing data.

Task 4 Variational autoencoders

Explain in one page, what are variational autoencoders. You can essentially summarize this tutorial on variational autoencoders in one page (https://towardsdatascience.com/intuitively-understanding-variational-autoencoders-1bfe67eb5daf).

Honor code

• Honor code of AUS will be followed.
• Always give reference for code or ideas.
• Do not copy code from the web or from any other person.

Submission

• Please note that the zip file must be named in following manner IDnumber_A04.zip (e.g. g00012345_A04.zip).
• Please also submit a hard copy of the report.
• No late submissions will be accepted.