CSCM77 Computer Vision and Deep Learning Coursework

Released: 9 a.m., 13th March, 2020.
Due: 11 a.m., 27th April, 2020.

This coursework specification changed on 17th March, 2020. Please ensure you have fully read the updated sections and understand the new requirements.

1 Coursework Assignment: Pac-Man

1. To be completed by students working individually.
2. **Feedback:** You will receive feedback after your four-page report has been marked.
3. **Learning outcome:** The tasks in this assignment are based on your practical work in the lab sessions and understanding of the theories and methods. Thus, through this coursework, you are expected to demonstrate both practical skills and theoretical knowledge of several computer vision techniques.
4. **Unfair practice:** This work is to be attempted individually. You may request help from your lecturer, academic tutor and lab tutor, but *you may not collaborate with your peers.*
5. **Submission deadline:** The code and your four-page report must be submitted on Blackboard **by 11 a.m. on Friday, 27th April, 2020**. You **must** submit your work **to Blackboard** before the deadline. Late submissions without extenuating circumstances will receive zero marks.
6. You will be accessed on your four-page report. Your code will stand as supporting material to your report.
7. This coursework constitutes 20% of your final mark for this module.

2 Task Description

In this coursework, you are given a set of 3D point-clouds with appearance features (i.e. RGB values). These point-clouds were collected using a Kinect system in our old PhD lab. Several virtual objects are also positioned among

those point clouds. Your task is to write a Python programme that can automatically detect those objects from an image and use them as anchors to navigate through the 3D scene. A set of example images that contain those virtual objects are provided. These example images are used to train a random-forest classifier (or a classifier of another kind explored in the labs) in order detect the objects. Some Python code is provided to help you complete the task. Code demonstrating how to obtain a 2D image by projecting 3D point-clouds onto the camera image-plane, and how to re-position and rotate the camera, is provided as well.

You will also write a four-page report on your work, which you will submit to Blackboard alongside your code. As of 17th March, 2020, you will be accessed on this report with your code as supporting material. The report must be structured as an academic paper, with the following structure:

Introduction (10%). Contextualise the machine-learning problem and introduce the task and the hypothesis. Make sure to include a few references to previous work. You should demonstrate an awareness of the research-area.

Methodology (50%). The model(s) you trained to undertake the task. Any decisions on hyperparameters must be stated here, including motivation for your choices where applicable. If the basis of your decision is experimentation with a number of parameters, then state this.

Results (30%). Describe, compare and contrast the results you obtained on your model(s). Any relationships in the data should be outlined and pointed out here. Only the most important conclusions should be mentioned in the text. By using tables and confusion-matrices to support the section, you can avoid describing the results fully.

Discussion and Conclusion (10%). Restate the task and hypothesis/-ses concisely. Reiterate the methods used. Describe the outcome of the experiment and the conclusion that you can draw from these results in respect of the hypothesis/-ses.

The following materials from lectures are relevant to this task:

1. Camera translation and orientation.

2. Feature descriptors, e.g. histograms of pixel intensity, histograms of oriented gradients, etc.

3. Supervised learning using random forests, convolutional neural networks, region-proposal networks.

All the software and data are available on Blackboard.

2.1 Demo Code

Demo code is provided for orienting the camera view and obtaining an image from the current camera view. The software to generate and visualise the point clouds are also provided. The Python functions that are required to train and test random forests are explained below.

Python provides a suite of methods for random-forest classification, namely the `sklearn.ensemble.RandomForestClassifier` class, which is able to train an ensemble for classification given some training observations and their labels. Key uses of `sklearn.ensemble.RandomForestClassifier` include:

- To instantiate a random-forest object:
  `classifier = sklearn.ensemble.RandomForestClassifier(n_estimators=n_trees)`
- Fit the Random Forest to the data and labels:
  `classifier.fit(data, labels)`
- Get predicted class probabilities given data:
  `prediction_probability = classifier.predict_proba(data)`

The use of these functions will be further explained in the lab sessions.

Random-forest classification is required to detect those artificial objects among the point clouds. Training should be carried out on the provided example images. The detection requires a sliding-window-based evaluation that is same as in human detection (HoG, covered in lectures). The sliding window should be the same size as the training images.

You can use any features to train the classifier, for example, a histogram of pixel values or histogram of oriented gradients. Once a virtual object is detected, you need to move your camera to where the virtual object is located in space and start your search for the next one until all virtual objects are found. In the event that multiple virtual objects are detected in a single view, the nearest virtual object should be selected.

2.2 Viva Voce

Following the termination of all face-to-face lectures at Swansea University on Tuesday, 17th March, 2020, there will be no viva voces. You will be accessed remotely based on your four-page report. You must however submit your code alongside your report.

2.3 Marking Criteria

Your report work will be assessed on its structure, content and presentation. We expect it to be read as an academic paper, with the explanation appropriately divided as per the structure described in the Task Description above. You should demonstrate your knowledge of the field, along with any conclusions you can draw from your results. We expect at least the same standard of work as in CSC345, the prerequiste module for CSCM77.

2.4 Assessment

You will be accessed on the four-page report that you submit to Blackboard. You must submit your code alongside your report, as this will act as supporting material. This assignment is worth 20% of the total credit.