The University of Melbourne
School of Computing and Information Systems
COMP90086 Computer Vision, 2022 Semester 2

Assignment 2: Convolutional neural networks for image classification

Due:	7pm, 9 Sept 2021
Submission:	Source code (in Jupyter Notebook) and written responses (as .pdf)
Marks:	The assignment will be marked out of 7 points, and will contribute 7% of your total mark.

In this assignment, you will use CNNs to classify a yoga pose dataset, `yoga32`. This dataset, based on a dataset created by Anastasia Marchenkova, includes 590 RGB colour images from 10 classes. Images have been downsampled to 32x32 pixels. We have provided a train/test split with 520 images for training/validation and 70 for testing – you should use the provided split throughout this assignment.

1. CNN implementation [3 pt]

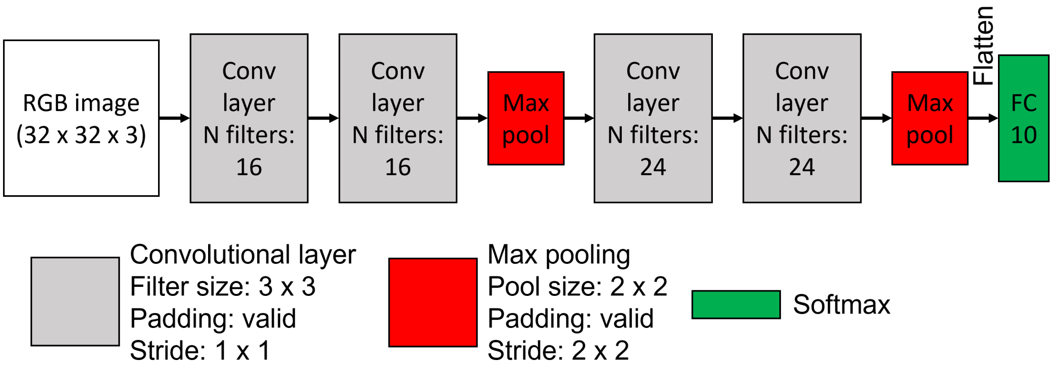

Figure 1: Network diagram

1.1. Basic architecture [1 pt]

Implement the CNN architecture shown above in Figure 1. Use ReLU activation functions for all layers except the final layer, which should use the Softmax activation function. Use the Adam optimiser and `SparseCategoricalCrossentropy` loss. Train this on the `yoga32` dataset – what do you observe?

1.2. Regularisation and data augmentation [2 pt]

Modify the basic architecture by adding some form of (a) regularisation and (b) data augmentation. Train your new network on the `yoga32` dataset – how does the training performance change?

Your write-up should include a brief description and justification of your choice of regularisation and data augmentation schemes. It should also show the plots of training and validation accuracy for the original network (without regularisation+data augmentation) and the network with these modifications and explain any differences that you observe in the training behaviour.

2. Error analysis [2 pt]

Evaluate your network from part 1.2 on the `yoga32` test set. In your write-up, present the overall classification accuracy and the average accuracy for each of the 10 classes. Explain the performance of the CNN model, using example images from the test set to illustrate your discussion. What classes/images were difficult for this model, and why?

3. Visualisation [2 pt]

Visualise the feature space that your network uses to classify images by implementing a nearest neighbour analysis. Use the embeddding from the last convolutional layer of your network from part 1.2 after it has been maxpooled (e.g., extract this layer at the point at which it is flattened and sent to the classification layer). To visualise how images are organised in this feature space, implement a nearest neighbour analysis. For each test image, find the 5 nearest neighbours in the training set. Use Euclidean distance to compare the feature vector from the test image to the feature vectors of the training images. In your write-up, show nearest neighbours for multiple test images to illustrate the feature space and explain your model's performance. Critically evaluate your model – has it learned a good feature space for this classification task?

Submission

You should make two submissions on the LMS: your code and a short written report explaining your method and results. The response to each question should be no more than 500 words.

Submission will be made via the Canvas LMS. Please submit your code and written report separately under the **Assignment 2: Code** and the **Assignment 2: Report** links on Canvas.

- Your **code** submission should include the Jupyter Notebook (please use the provided template) with your code and any additional files we will need to run your code, if any (do not include the `yoga32` dataset).
- Your written **report** should be a .pdf with your answers to each of the questions. The report should address the questions posed in this assignment and include any images, diagrams, or tables required by the question.

Evaluation

Your submission will be marked on the correctness of your code/method, including the quality and efficiency of your code and the appropriateness of design decisions. You should use built-in Python functions where appropriate and use descriptive variable names. Your written report should clearly explain your approach and any experimentation used to produce your results, and include all of the specific outputs required by the question (e.g., images, diagrams, tables, or responses to sub-questions).

Late submission

The submission mechanism will stay open for one week after the submission deadline. Late submissions will be penalised at 10% of the total possible mark per 24-hour period after the original deadline. Submissions will be closed 7 days (168 hours) after the published assignment deadline, and no further submissions will be accepted after this point.

Updates to the assignment specifications

If any changes or clarifications are made to the project specification, these will be posted on the LMS.

Academic misconduct

You are welcome — indeed encouraged — to collaborate with your peers in terms of the conceptualisation and framing of the problem. For example, we encourage you to discuss what the assignment specification is asking you to do, or what you would need to implement to be able to respond to a question.

However, sharing materials — for example, showing other students your code or colluding in writing responses to questions — or plagiarising existing code or material will be considered cheating. Your submission must be your own original, individual work. We will invoke University's Academic Misconduct policy (`http://academichonesty.unimelb.edu.au/policy.html`) where inappropriate levels of plagiarism or collusion are deemed to have taken place.