CAP 4630 – Intro to AI – Dr. Oge Marques – Summer 2019

Final Project: Cats vs. Dogs

In this project you will design and implement a deep learning solution for image classification based on images of cats and dogs.

Goal: to build a deep learning solution in MATLAB that is capable of predicting whether an input color image containing an animal represents a *cat* or a *dog*.

You can choose whatever preprocessing image processing techniques, machine learning algorithms, and MATLAB toolboxes you wish, but keep in mind that the most important points are:

- To demonstrate the use of a sound **methodology** for every step of your work
- To **document** each step (and associated design decisions, criteria, references, algorithms, strengths and weaknesses) of your work in a clear and readable manner.

The expected deliverables are:

- Self-contained MATLAB code to run your solution (and details on how to install and run it)
- A descriptive report containing the most important steps, building blocks, design decisions, examples of incorrect prediction results, and a ***summary table*** that should contain overall accuracy rates for, **at least, three different approaches (and their variants, if applicable).**

Learning objectives:

- Learn how to implement a complete, fully functional, deep learning solution for a contemporary computer vision challenge using MATLAB.
- Get acquainted with representative contemporary datasets, challenges and problems in computer vision and machine learning.
- Learn how to use deep neural networks under the paradigm of *transfer learning*.
- Demonstrate the ability to perform model selection, fine-tuning, and performance evaluation of different solutions to the same problem.

Starter package

- **Dataset**: *cats and dogs* dataset from Kaggle
  (https://www.kaggle.com/c/dogs-vs-cats/data)

- **Reference links**:
  - https://www.kaggle.com/c/dogs-vs-cats (more details about the Kaggle challenge associated with the dataset)
  - https://www.mathworks.com/help/deeplearning/deep-learning-with-images.html (deep learning with images using MATLAB)
  - https://www.mathworks.com/help/deeplearning/gs/get-started-with-transfer-learning.html (transfer learning using MATLAB)
  - https://www.mathworks.com/help/deeplearning/ug/pretrained-convolutional-neural-networks.html (pre-trained networks available in MATLAB)

- https://www.mathworks.com/matlabcentral/fileexchange/59133-deep-learning-toolbox-model-for-alexnet-network (AlexNet in MATLAB)

- **MATLAB starter code**: ***final_project_starter_cap4630.m*** (and associated data) (available on Canvas)

Instructions:

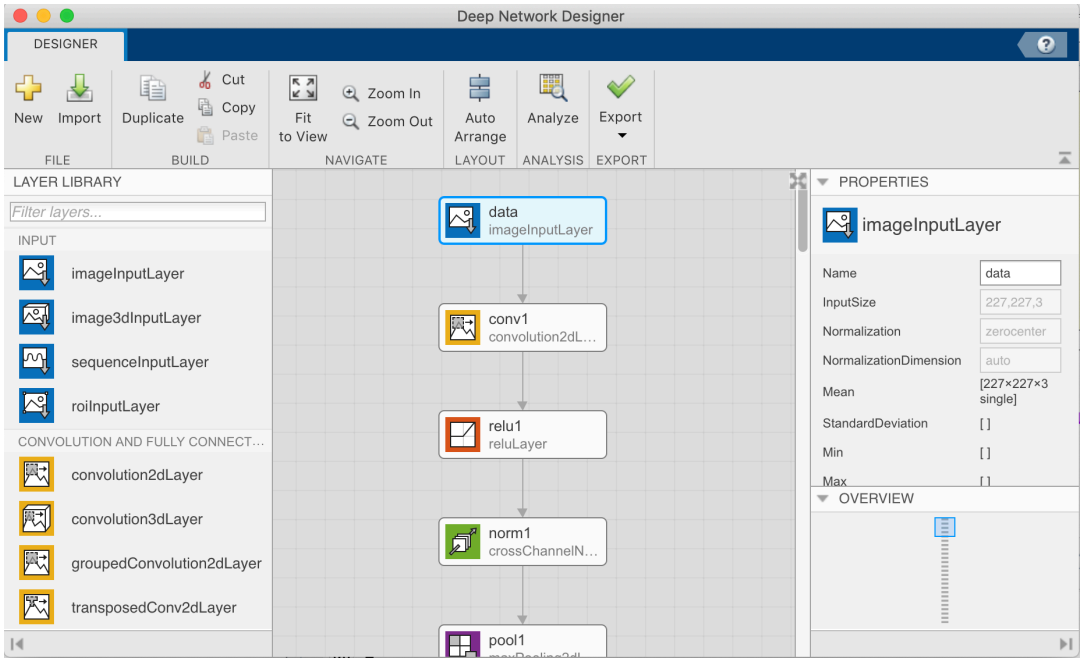
- **This is a group activity. Groups of 1-3 students are allowed.**
- Open collaboration among groups is **not** allowed, except to help fellow classmates solve technical questions that may be causing difficulties and delays.

Procedure:

1. Download MATLAB starter code (and associated data) and add to your working folder, adjusting the MATLAB path if necessary.

2. Run "Part 1" of the starter code and ensure that it works as intended. Inspect the contents of variable **model**.

3. (OPTIONAL) Used the "Deep Network Designer" app (within MATLAB) to explore the model interactively (see screenshot on the next page).



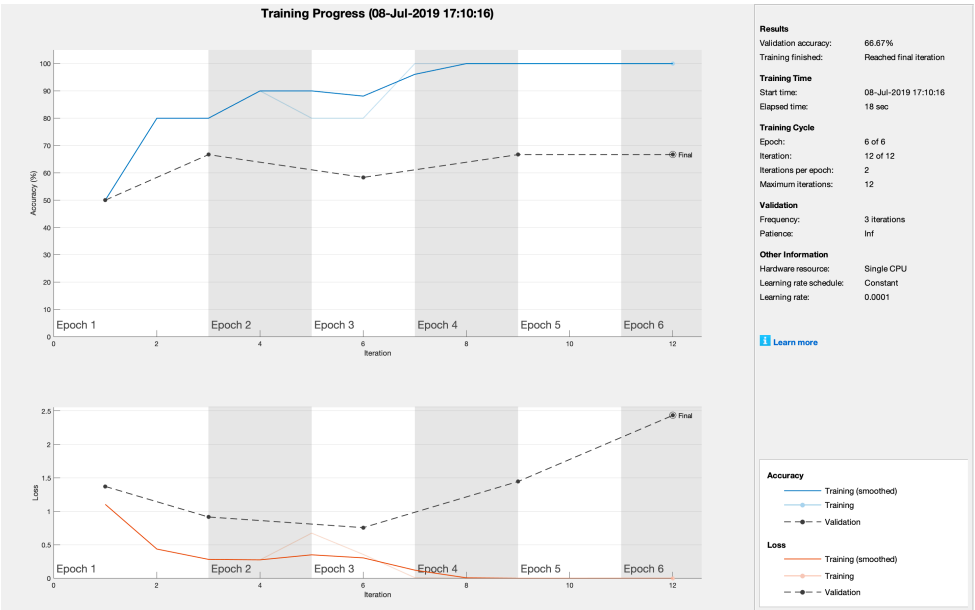
Question 1: What type of preprocessing is performed by the auxiliary function *readAndPreprocessImage*?

Question 2: What can you say about the montage with network weights for the second convolutional layer (above)?

4. Run "Part 2" of the starter code and ensure that it works as intended.

Question 3: How many images are there in each set (training / validation)?

5. Run "Part 3" of the starter code and ensure that it works as intended. You should see learning curves similar (but not identical!) to the ones below.

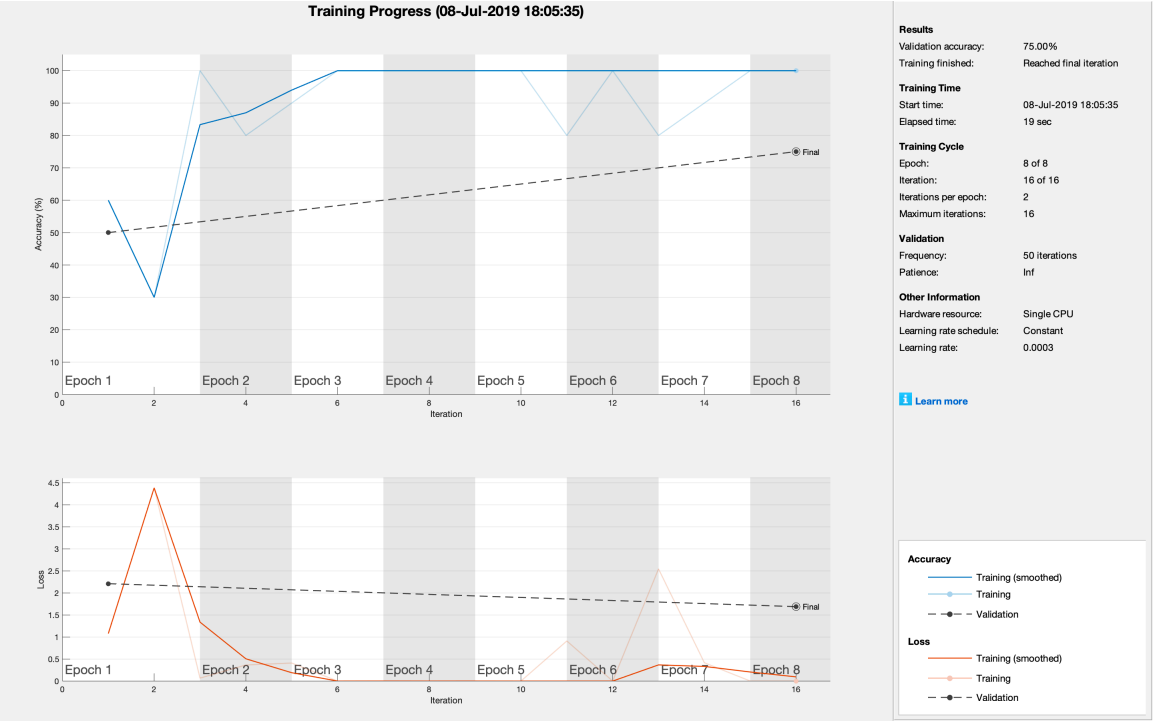


Question 4: Is the validation accuracy (in my example 66.67%) acceptable for a two-class classifier? Why (not)? If not, what could be the problem?

6. Finish the code for "Part 3.7".

Question 5: Did your classifier recognize 'Doge' as a dog? If not, can you tell why?

7. Run "Part 4" of the starter code and ensure that it works as intended. You should see learning curves similar (but not identical!) to the ones below.



Question 6: Is the validation accuracy (in my example 75%) better than before? What could be the reason(s) behind such (modest) improvement? How could you improve it even further?

8. The starter code used a very small subset of the Kaggle dataset (20 images of dogs and 20 images of cats) out of the 25,000 images (2 x 12,500) available. This has to be changed. Follow the steps below.

9. Download the **train.zip** data file from the dataset link indicated above (https://www.kaggle.com/c/dogs-vs-cats/data). You can disregard test1.zip (for now) and the sampleSubmission CSV file (forever).

Data Files

File Name	Available Formats
sampleSubmission	.csv (86.82 kb)
test1	.zip (271.15 mb)
train	.zip (543.16 mb)

10. Write code for "Part 5" of the starter code.

You should partition your dataset into:

- Training: 80 %
- Hold-out validation: 20 %

If you use different values (or "play" with different choices), make sure to document that.

Report your **accuracy on the validation set**.

Question 7: Did the validation accuracy improve as a result of using a much larger training dataset? How could you improve it even further?

11. Start a new MATLAB script containing your "final solution" to the assignment. Copy and paste all the relevant pieces from the starter code, but not more than absolutely necessary.

Your final solution script should contain three parts:

i. "Baseline classifier" (the one from "Part 5" above)
ii. "Improved classifier 1"
iii. "Improved classifier 2"

12. Perform **at least two significant changes** to the classifier used in Part 5 (pre-trained AlexNet with parameters from starter code), in order to produce the so-called "improved classifiers".

Examples of meaningful changes include:

i. different pre-trained networks (VGG-16, ResNet, Inception, etc.)
ii. data augmentation (if you haven't used it in Part 5 yet)
iii. different hyperparameters (learning rate, mini-batch size, optimizer, etc.)

13. Then, *and only then*, write code to test the three approaches using the Kaggle test dataset (test1.zip file from Kaggle).

14. Prepare a **summary table** containing – at least – three rows (one for each classifier) and three columns (model name, accuracy on training set, accuracy on test set)

15. Produce a meaningful technical **report**. In addition to the summary table include, if applicable, other tables and plots (e.g., ROC curves, confusion matrices, etc.), as well as representative samples of best and worst results.

Deliverables

You should submit **a single zip file** via Blackboard.

It should contain:

- All necessary m-files to properly reproduce your results.
- All relevant third-party MATLAB scripts and functions that were ***not*** provided in the starter package (with an acknowledgment of their source), if any.
- A ***detailed*** report (PDF) containing your comments, remarks, examples, answers to the questions above, relevant tables and plots, etc.
- A README file describing what is in the package, and (optionally) containing instructions on how to "install" (e.g., directory structure, dependencies, etc.) and run your code.

It should ***not*** contain:

- Any images or files that were already distributed as part of the starter package.
- The actual dataset.
- Older versions of your code or any leftovers that are not relevant for (and might even hurt) grading your work.