CSC345/M45 Big Data and Machine Learning

Coursework

Policy

1. To be completed by students working individually.
2. Feedback: individual feedback is given on Blackboard within two weeks of deadline.
3. Learning outcome: The tasks in this assignment are based on both your practical work in the lab sessions and your understanding of the theories and methods. Thus, through this coursework, you are expected to demonstrate both practical skills and theoretical knowledge that you have learned through this module. You also learn to formally present your understandings through technical writing. It is an opportunity to apply analytical and critical thinking, as well as practical implementation.
4. Unfair practice: This work is to be attempted individually. You may get help from your lecturer, academic tutor and lab tutor, but you may not collaborate with your peers. **Copy and paste from the internet is not allowed. Using external code without proper referencing is also considered as breaching academic integrity.**
5. Submission deadline: The report and your implementation code in MATLAB need to be submitted electronically to Blackboard by **11AM Friday 28 April**.

1. The Task

The amount of image data is growing exponentially, due in part to readily available camera equipment. Teaching computers to recognise objects within a scene has tremendous application prospects, with applications ranging from medical diagnostics to robotics. Object recognition problems have been studied for decades in machine learning, however it is still a challenging and open problem. The following task is your first small step on this interesting question within machine learning.

You are provided with an image dataset, where there are 10 different categories of objects, each of which has 1000 images for training and 100 images for testing. Each image only contains one object. The task is to apply supervised learning algorithms to classify the testing images into 10 object categories. The code to compute image features and visualise the image is provided. You can use it to visualise the images, compute features, and transform them if necessary, e.g. using PCA and LDA. You will then perform supervised classification and report quantitative results; writing this up into a 4-page report. You don't have to use all the provided code or methods discussed in this module. You may add additional steps to the process if you wish.

2. Image Dataset – Subset of CIFAR-10

We select a sub-set of 10 object categories from the complete CIFAR-10 dataset. Each category contains 1000 training images and 100 testing images, which are stored in two 4D arrays. The corresponding category labels are also provided. The size of each image is fixed at 32x32x3, corresponding to height, width, and colour channel, respectively. The training images will be used to train your model(s), and the testing images will be used to evaluate your model(s). You can download the image dataset and relevant code for visualisation and feature extraction from the URL

link provided as follows: (http://csvision.swansea.ac.uk/BDML/CW2.zip). These are also made available on Blackboard.

There are four variables in the *CW2Data.mat* file, as follows:

- *trnImage*, 32x32x3x10000 matrix, training images (RGB image)
- *trnLabel,* 10000x1 matrix, training labels (1-10)
- *tstImage*, 32x32x3x1000 matrix, testing images (RGB image)
- *tstLabel,* 1000x1 matrix, testing labels (1-10)

The image data is stored in a 4D matrix, and for many of you this will be the first time seeing a high dimensionality matrix. Although this may seem intimidating, it is relatively straightforward. The first dimension is the height of the image, the second dimension is the width, the third dimension is the colour channels (RGB), and the fourth dimension is the samples. Indexing into the matrix is similar to as with any other numeric matrix in MATLAB, but now we deal with the additional dimensions. So, in a 4D matrix 'X', to index all pixels in all channels of the 5^{th} image, we use the index notation X(:,:,:,5). So in a generic form, if we want to index into the i,j,k,l^{th} element of X we use X(i,j,k,l).

Figure 1. 10 Categories of CIFAR-10 Dataset

3. Compute Features and Visualise Images

The following two functions are provided to compute image features and visualise the images of a specified category. A demo script is provided to showcase how to use those two functions.

- *RunMe.m*, Demo script of how to use two provided functions
- *ComputeFeature.m*, Compute image features
- *Visualize.m*, Show 100 random images from a specified category

You are NOT asked to understand how these features are extracted from the images, but feel free to explore the underlying code and the MATLAB API. You can simply treat the features as the same as the features you loaded from wine dataset in the Lab work.

To compute the image features from an image, you can use the following code *"Feature = ComputeFeature(Image);"*, where the input variable *Image* is a 32x32x3 matrix that can be indexed from the provided 4D image dataset. A feature vector of 1x324 is then obtained, which can be simply considered as 324 measurements for a given image. You can construct a *for* loop to compute the image features for all the images and store the features in two separate 2D matrices for the training set and the testing set, respectively. These 2D matrices can be organised as follows: each row represents one image, and each column corresponds to one measurement. Therefore, the sizes of feature matrices for training and testing should be 10000x324 and 1000x324, respectively. The corresponding category labels for training and testing images are provided in the following two variables, *trnLabel, tstLabel,* which are required for supervised learning and accuracy evaluation. Please note that you are only allowed to build your models using the training set, and evaluate the learnt models using the testing set.

4. Learning Algorithms

You can find all relative learning algorithms in the lab sheets and lecture notes. You can use the following algorithms (MATLAB built-in functions) to analyse the data and carry out the classification task. Please note that if you feed the learning algorithm with a large chunk of data, it may take a while to train.

- K-Mean: Lab sheet 1
- Gaussian Mixture Model: Lab sheet 1
- Principal Component Analysis: Lab sheet 2
- Linear Discriminative Analysis: Lab sheet 2
- Support Vector Machine: Lab sheet 3
- Neural Networks: Lab sheet 3

5. Benchmark and Discussion

Your proposed method should be trained on the training set alone, and then evaluated on the testing set. To evaluate, you should count, for each category, the percentage of correct recognition (i.e. classification), and report the confusion matrix. The benchmark to compare against is 44.60%, averaged across all 10 categories. Note, this is only a reference, not a target.

An example confusion matrix is given below:

		Target									
		airplane	automobile	bird	cat	deer	dog	frog	horse	ship	truck
Predict	airplane	51	6	10	1	9	2	2	2	12	2
	automobile	4	38	1	4	7	3	4	1	14	11
	bird	9	0	33	16	6	10	8	3	6	3
	cat	3	1	8	18	16	14	7	3	2	1
	deer	6	6	4	12	37	8	8	19	0	5
	dog	1	2	13	13	2	43	9	12	0	3
	frog	5	15	16	19	7	10	60	1	1	1
	horse	1	3	8	10	11	9	1	49	1	3
	ship	17	22	5	4	3	1	1	2	55	9
	truck	3	7	2	3	2	0	0	8	9	62

Report

You are required to write a 4-page report to summarize your proposed method and the results. Your report should contain the following sections:

1. **Introduction**. Provide an overview of the problem, your proposed solution, and your experimental results. [10%]
2. **Method.** Present your proposed method in detail. This should cover how the features are extracted, any feature processing you use (e.g. clustering and histogram generation, dimensionality reduction), which classifier(s) is/are used, and how they are trained and tested. This section may contain multiple sub-sections. [50%]
3. **Results.** Present your experimental results in this section. Explain the evaluation metric(s) you use and present the quantitative results (including the confusion matrix). If you have tried multiple solutions, present all the results. [30%]
4. **Conclusion.** Provide a summary for your method and the results. Also, provide your critical analysis; that is the shortcomings of your method and how they may be improved. [10%]
5. **References.** Include references where appropriate. References are included in the page limit.

Page Limit: The report should be no more than **4 pages**. Font size should be no smaller than 10, and the text area is approximately 9.5x6 inches. You may use images but do so with care; do not use images to fill up the pages. You may use an additional cover sheet, which has your name and student number. Reports that exceed the specified page limit will result in penalties: **10% deduction for every over-length page**.

Source Code: Submit your MATLAB source code to Blackboard, together with your report, in **a Single Zip file**.

Assessment

The percentages listed above indicate the distributions of marks. This assignment is worth 15% of the total credit. Submitted work without an implementation will lose the entire mark for section 4 and portions of marks for other sections.

Submission

Submit your work electronically to Blackboard. **Your report should be in PDF format only**. Compress your MATLAB source code and report into **a Single Zip file**. The deadline for this coursework is **11AM Friday 28 April**.