EE452 Computer Vision (Spring 2021)

Assignment 1

Training Neural Networks for Image Classification

Release date: Thursday, 25 February 2021

Due date: Friday, 19 March 2021

Maximum Marks: 100 (Weight in Final grade: 8%)

Objectives:

- To understand the usage of numpy for vectorized coding and perform preprocessing of data set
- To perform feature extraction and apply Support Vector Machine (SVM) classifier on extracted features
- To create dense and sparse neural network classifiers from scratch and apply on CIFAR-10 dataset
- To perform regularization to improve the trained networks
- To create neural network classifier using transfer learning (trained on ImageNet) and apply on CIFAR-10 dataset
- To perform hyperparameter tuning

Instructions:

- This assignment is individual task where only collaboration in terms of discussing and idea sharing is allowed.
- A report is to be submitted on LMS describing the different phases of the work also including what worked, what did not work, what you learned, what extra things you tried and how to proceed with this etc.

Tasks:

1. To get started, look at TensorFlow Datasets and specifically the malaria dataset. This is a fun application that would introduce you to the world of medical applications of Computer Vision. You need to do the following: **[10]**

a. Load data: You can use the tfds package to load the dataset. Look at a few samples from different categories, do you see any visual difference? If yes, what is that difference?
b. Build a binary classification model for this task. Briefly describe what binary classification is and why we are using binary classification for this task.
c. Pre-process the data: Describe all your steps and your rationale behind it.
d. Use sklearn to build a LogisticRegression model. You can look at sklearn.linear_model.LogisticRegression documentation to understand how it works. [Note: To train the model and predict using the model, you would first need to flatten the image. Generally the idea is to extract some features, build a feature vector and then use that to classify, but for this task you can use the flattened image as a feature vector.]

e. Split your data into train/test splits using sklearn. Train on the train split and evaluate on the test split. You're not required to get accuracy above a certain threshold, but it would be nice to have > 70% test accuracy.
f. [Optional] Repeat tasks 4 and 5 below for this dataset. Notice and understand how a feature extractor and a neural network helps you with this task.

2. In a different notebook, load the CIFAR-10 Dataset (this can be done using the following command in keras: from keras.datasets import cifar10) and briefly describe what the dataset is about. Also mention some of the preprocessing steps which can be applied while dealing with image data in general. [**5**]

3. Write a function "filt" whose aim would be to perform the task of filtering. The function should take the following arguments: input, filter, padding, normalization. The 'input' would be the numpy representation of an image over which filtering is to be performed - this will either have a single channel or three channels, 'filter' would be a 2D numpy array representing the filter (for example: np.array([[1, 1, 1], [1, 1, 1], [1, 1, 1]]), 'padding' would be a boolean value which would determine as to whether the output image should have the same size as the input image, and 'normalization' would again be a boolean value which would determine as to whether the filtering operation needs to be normalized or not. The output of this function should be the filtered image. Also note that the only external library which can be used in this function is "numpy". [**10**]

4. Extract HOG features from the CIFAR-10 dataset, and train a Linear SVM using those features. You are expected to reach around 45% accuracy on test data using this method. [**15**]

5. This part requires you to use either tensorflow, pytorch, or keras. In each of the following subparts (a, b, and c) you are required to do the following: [**20x3**]

i. Create a network architecture
ii. Load the CIFAR-10 dataset
iii. Preprocess the dataset accordingly
iv. Train your architecture on the dataset for a reasonable number of epochs (maximum is 100) using the testing data as your validation data
v. Report both your training accuracy as well as validation accuracy
vi. Plot both the training loss as well as the validation loss on the same plot
vii. Plot both the training accuracy as well as the validation accuracy on the same plot

For each subpart you are encouraged to experiment both with the hyperparameters as well as different architecture designs for your network

a. Create a feed forward neural network consisting of only fully connected layers; the network should have at least two hidden layers. You are allowed to experiment with regularization techniques and are expected to reach above 55% accuracy on test data.

b. Create a convolutional neural network consisting of at least two convolutional layers. You are allowed to experiment with regularization techniques and are expected to reach above 80% accuracy on test data.

c. Use the pre-trained VGG-16 network (trained on ImageNet) as a feature extractor, and connect it to a feed forward network consisting of fully connected layers. You should try to fine tune the network and are expected to reach above 70% accuracy on test data.

Submission Guidelines:

The entire assignment is to be done as a python notebook. Once you are done, you should upload the notebook to LMS. Create sections within the notebook in order to answer each question. For the second question, you should also display some results of using your function on custom images (which you may obtain from the web). For the fourth question, you should only submit one model architecture for each part, and in case you have experimented with other architectures, you should mention those in a dedicated paragraph.

It is preferred that you do this assignment on google colab: https://colab.research.google.com/. Colab allows you to utilize its GPU, which can be accessed as follows: Runtime => Change runtime type => Change Hardware accelerator to GPU. Once you have completed your assignment on colab, you can download it as a notebook as follows: File => Download.ipynb