[Spring 2023] CS172 Assignment 1
Basics in computer vision

#NAME #StudentID

March 10, 2023

Acknowledgements

1. Deadline: **2023/4/17 23:59:00**. Late Policy shows as follows:
   - All students have 5 free late days with no penalty.
   - You may use up to 3 late days per assignment with no penalty.
   - You may not use late days for the final project.
   - Once you have exhausted your free late days, we will deduct a late penalty of 10% per additional late day.
2. Please submit your assignment in **Gradescope** in PDF format.
   - Giving your report in English, a report in Chinese is not accepted.
   - Handwritten homework is not accepted and we highly recommend using LaTeX. The LaTeX template has been uploaded to Blackboard.
   - Note that you **MUST** select pages for each question on Gradescope, and TA will **NEVER** select pages for you. Otherwise, you will lose **ALL** the points.
3. Please upload your code zip to the **ShanghaiTech cloud** disk.
   - `http://pan.shanghaitech.edu.cn/cloudservice/outerLink/decode?c3Vnb24xNjc4NDE2NjM3MzMyc3Vnb24=`
   - All source files and *readme* should be included but remember to remove the datasets.
   - Your zip should be named as CS172_NAME_ID_hw1.zip.
4. Plagiarism or cheating is strictly prohibited.
   - DO NOT share your assignment or code!
   - NO fake solution is allowed!
   - Make sure your codes can run and are consistent with your solutions.

1 [10 points] Preparing the image dataset

1. Download CIFAR-10 dataset from its Official website.
2. Write a function that can read an image from the dataset and display it with the corresponding label.
3. Select an image and apply at least two image augmentation methods (like filp, rotation, blur, mixup etc.) to it. Show the results.

Hint: You can use libraries like opencv-python, PIL or scikit-learn.

2 [15 points] Hand-crafted feature extraction

Historgram of oriented Gradient (HOG) is a classic feature descriptor in computer vision. The descriptor can be used for object detection and recognition tasks using machine learning algorithms such as support vector machines (SVM). **You are required to:**

1. Explain the steps how to extract HOG feature.
2. Use off-the-shelf function in python libraries to extract HOG feature from an image.
3. Describe the relationship between original image and the extracted HOG feature.

3 [20 points] SVM classification

A Support Vector Machine (SVM) is a machine learning algorithm that can be used for classification or regression tasks. In the context of classification, the SVM algorithm **finds a hyperplane** that separates the data points in different classes with the maximum margin. The hyperplane is defined by a set of weights and biases that are learned from the training data. In this section, **you are required to:**

1. Implement a SVM classifier with hinge loss.
2. Train your model with the extracted HOG features and try to improve the performance. Achieve 35% accuracy at least.
3. Explain the backward pass in the training of SVM classifier.

Hint: Principal component analysis (PCA) is useful to reduce feature dimension and make feature condensed. Try to add PCA function and compare these two versions.

4 [25 points] Build a simple neural network

With the development of deep learning, neural networks like fully connected network (FCN) and convolutional neural network (CNN) start to replace the hand-crafted feature extraction algorithms. A common architecture of CNN is a stack of **Conv2D** and **MaxPooling2D** layers followed by a few fully connected layers. Then these features are flattened and fed to other fully connected layers that determine the class of an image based on the presence of features. **Please try to build your own CNN in Pytorch.**

1. Learn to use pytorch to build a simple fully connected network (Quickstart of pytorch). Understand the class *NeuralNetwork* and know the *train* and *test* function.
2. Feel free to design your own network architecture, by adding convolutional layers, pooling layers or changing the number or sequence of these layers.
3. Feel free to modify the hyperparameters like convolution kernel size, padding size, stride size and so on. To achieve a better performance, you may iteratively repeat step 2 and 3.
4. Describe the structure of your network and analyse the performance on CIFAR-10.

Hint: You may change the way to read dataset by using dataloader such as torchvision.datasets.CIFAR10. Torch.utils.data.DataLoader

5 [20 points] Pre-train and fine-tune

The pre-trained models are trained on very large scale image classification problems. The convolutional layers act as feature extractor and the fully connected layers act as Classifiers. Since these models are very large and have seen a huge number of images, they tend to learn better and more discriminative features. We can either use the convolutional layers as a feature extractor or modify the already trained convolutional layers to adapt to our problem. The former approach is known as pre-training and the latter as Fine-tuning. Here are the steps:

1. Download a pre-trained deep learning model, such as VGG, ResNet, or Inception, from a repository like **PyTorch Hub**. Or use *torchvision.models*.
2. Fine-tune the pre-trained model on CIFAR-10 by replacing the last layer(s) of the pre-trained model with a new set of fully connected layers, and retraining the model on the dataset.
3. Evaluate the performance of the fine-tuned model on the dataset by computing the accuracy and comparing the results to the SVM classifier in last section.

6 [10 points] Visualization with t-SNE

t-SNE (t-Distributed Stochastic Neighbor Embedding) is a popular machine learning technique used for data visualization and dimensionality reduction. **The result is a low-dimensional representation of the data** that captures the underlying structure of the high-dimensional dataset. t-SNE is particularly useful for visualizing complex, high-dimensional datasets in two or three dimensions, where the structure of the data can be better understood and interpreted. It has been used in various applications such as image and text analysis, bioinformatics, and recommendation systems.In this section, **you may need to**:

1. Choose a subset of CIFAR-10, such as a specific class or set of classes.

2. Apply t-SNE dimensionality reduction to the subset of images using a library such as scikit-learn.
3. Visualize the t-SNE embedding of the subset of images in a two-dimensional scatter plot, where each point represents an image and the color represents the class label.
4. Interpret the visualization by analyzing the clusters and patterns in the scatter plot, and comparing them to the original dataset.