Assignment-1: Introductory Vision

Instructions:

[1] Any plagiarism will lead to award of **F grade** STRICTLY

[2] Use python only for the implementation of all the assignments

[3] Use NumPy to represent the vector and array

[4] Do not use the inbuilt functionality of any library including NumPy until suggest so.

[5] PyTorch must be used to implement the deep learning-based methods.

[6] One mark will be deducted for each late day.

[7] Submit via Moodle only. Email submissions won't be considered.

No.	Question	Marks
1	Data source: Any standard single Face image-based dataset Write a program to	10
	I. Implement the Histogram of Local Binary Patterns (LBP) from screech using python. Read images from a directory Train/Val/Test and generate a Train.csv/Val.csv/Test.csv	2
	II. Use KNN classifier or One-vs-Rest SVM classifier (choose best distance metrics and K in KNN and best kernel and hyperparameters in SVM case using Val.csv) report the accuracy on Test.csv for best case of each	2
	III. Collect non-face images from any source with the same number as the face and train a binary SVM for face vs. non-face (*you can use the sklearn library for SVM*)	2
	IV. Implement a sliding window-based detection approach for any images having multiple faces and detect the face using the trained SVM model	2
	V. Show the scatter plots using PCA and TSNE for the generated LBP features	2
4	Data source: Any standard Human/Person detection dataset Write a program to	10
	I. Implement the Histogram of Oriented Gradients (HOG) descriptor from screech using python. Read images from a directory Train/Val/Test and generate a Train.csv/Val.csv/Test.csv	2 2
	II. Use KNN classifier or One-vs-Rest SVM classifier (*choose best distance metrics and K in KNN and best kernel and hyperparameters in SVM case using Val.csv and report the accuracy on Test.csv for best case of each*)	2
	III. Collect non-Human images from any source with the same number as the Humans and train a binary SVM for Human vs. non-Human (*you can use the sklearn library for SVM*)	2 2
	IV. Implement a sliding window-based detection approach for any images having multiple persons and detect the persons using the trained SVM model	
	V. Show the scatter plots using PCA and TSNE for the generated LBP features	

4	Data source: MNIST	16
	I. Implement a 1-hidden layer neural network in NumPy, use sigmoid function as activation in all layers and train it using MSE and SGD with manually calculated gradients.	2
	II. Implement a 1-hidden layer autoencoder using tanh activation in hidden layer and linear in output layer in PyTorch.	2
	III. Implement a 3 layer MLP using PyTorch use sigmoid function as activation in all layers and train it using MSE and SGD with autograd.	2
	IV. Implement a 3 layer MLP using PyTorch use sigmoid function as activation in all layers and initialize the first 2-layers with pretrained weights as per the deep belief network using autoencoders then fine tune it using MSE and SGD with autograd. Use appropriate learning rate or a momentum on weights. Compare the results of the III and IV for training vs validation losses and accuracies. Plot the train and validation curves in a single figure.	2
	V. Implement LeNet and compare with the best of above.	2
	VI. Create the train loss, validation loss, and validation accuracy curves in same figure, with iteration number or number of epochs on x axis.	2
	VII. Make sure via loss and accuracy curves that your model does not overfit (try different learning rates, other hyperparameters to avoid overfitting). Decide the best model weights (out of model weights at different iterations/epochs, based on your loss and accuracy curves.	2
	VIII. What do you observe in the final figure in previous part? In case where the point of first global minima (from left) of validation loss is at a different iteration/epoch than the first global maxima (from left) of validation accuracy, give justification about which model weights would be better (out of validation loss first global minima and validation accuracy first global maxima) and in which scenario?	2
5	Data source: CIFAR10 Write a program in PyTorch to	14
	I. Implement a 3 CONV + 1 FC layers CNN for classification and train with CIFAR10 dataset. Fine tune it for the filter size, activation function, pooling, batch normalization, data augmentation, and other model and optimization hyperparameters.	5
	II. Implement a program to train AlexNet, VGG16, GoogLeNet, ResNet152, EfficientNet-b1 from scratch on CIFAR10. Fine tune each for best hyperparameter. Compare the results losses and accuracies in two single figures each for train and val.	5
	IX. Create the train loss, validation loss, and validation accuracy curves in same figure. Make sure via loss and accuracy curves that your model does not overfit (try different learning rates, other hyperparameters to avoid overfitting). Decide the best model (iteration/epoch) based on such curves. Give proper justification for your choice.	4