Computer Vision 2024
(CSE344/ CSE544/ ECE344/ ECE544)
Assignment-1

Max Marks (UG/PG): 85/ 91 + (11 Bonus) **Due Date**: 20-Feb-2024, 11:59 PM

Instructions

- Keep collaborations at high-level discussions. Copying/plagiarism will be dealt with strictly.
- Your submission should be a single zip file **Roll_Number_HW[n].zip**. Include only the **relevant files** arranged with proper names. A single **.pdf report** explaining your codes with relevant graphs, visualization and solution to theory questions.
- Remember to **turn in** after uploading on Google Classroom. No justifications would be taken regarding this after the deadline.
- Start the assignment early. Resolve all your doubts from TAs during their office hours **two days before the deadline.**
- Kindly **document** your code. Don't forget to include all the necessary plots in your report.
- All [**PG**] questions, if any, are **optional for UG** students but are **mandatory for PG** students. UG students will get BONUS marks for solving that question.
- All [**BONUS**] questions, if any, are optional for all the students. As the name suggests, BONUS marks will be awarded to all the students who solve these questions.
- Your submission **must include a single python (*.py*) file for each question**. You can submit *.ipynb* along with the *.py* files. Failing to follow the naming convention or not submitting the python files will incur a **penalty**.

1. (15(UG)/18(PG) points) **Theory**

   1. (*7(UG)/10(PG) points*) Many customers use the yellow color of the papaya skin to evaluate its level of sweetness. To help customers who aren't aware of this fact, you decide to build an image classifier to predict whether a papaya is sweet (label=1) or not (label=0).

      (a) (1 point) You've built your own labeled dataset, chosen a neural network architecture, and are thinking about using the mean squared error (MSE) loss to optimize model parameters. Give one reason why MSE might not be a good choice for your loss function.

(b) (1 point) You finally decide to use the binary cross-entropy (BCE) loss to optimize your network. Write down the formula for this loss (for a single example) in terms of the label y and prediction $\widehat{y}$.

(c) (1 point) You are still not sure, so you decided check your implementation of the BCE loss. What value does the loss take for a prediction of $\widehat{y} = 0.9$ on a negative ($y = 0$) example.

(d) (2 points) Compute the total loss, L, of the network averaged across the following dataset of 3 examples using the binary cross entropy loss. $Y = \{y_1, y_2, y_3\} = \{1, 0, 0\}$, and the corresponding predictions as $\widehat{Y} = \{\widehat{y}_1, \widehat{y}_2, \widehat{y}_3\} = \{0.1, 0.2, 0.7\}$).

(e) (2 points) You add L2 regularization to your loss function. For a particular trainable weight W, write the update formula for W when the standard gradient descent optimizer is used with L2 regularization. Write your answer in terms of the learning rate α, L2 regularization hyperparameter λ, and the gradient of binary cross entropy loss function L_{BCE}. You decide to train one model with L2 regularization (model A) and one without (model B). How would you expect model A's weights to compare to model B's weights?

(f) **[PG]** (3 points) Given two separate probability distributions $P(x)$ and $Q(x)$ over a random variable x. Explain **KL (Kullback-Leibler) divergence** $D_{KL}(P, Q)$, and **cross entropy** $H(P, Q)$. Also, Derive the relation between them. *Hint: refer to deep learning book by Ian Goodfellow.*

2. (*8 points*) The softmax function has the desirable property that it outputs a probability distribution, and is often used as an activation function in many classification neural networks. Consider a 2-layer neural network for K-class classification using softmax activation and cross-entropy loss, defined as: $z^{[1]} = W^{[1]}x + b^{[1]}$, $a^{[1]} = \text{LeakyReLU}(z^{[1]}, \alpha = 0.01)$[1], $z^{[2]} = W^{[2]}a^{[1]} + b^{[2]}$, and $\hat{y} = softmax(z^{[2]})$. Loss for this neural network is given as:

$$L = -\sum y_i \log(\hat{y}_i)$$

where the model is given input x of shape $D_x \times 1$, and *one-hot encoded* label $y \in \{0, 1\}^K$. Assume that the hidden layer has D_a nodes, i.e. $z^{[1]}$ is a vector of size $D_a \times 1$. Let z_k denote the k^{th} element of the vector z.

(a) (1 points) What are the shapes of $W^{[2]}$, $b^{[2]}$? If we were vectorizing across m examples, i.e. using a batch of m samples $X \in \mathbb{R}^{D_x \times m}$ as the input, what would be the shape of the output of the hidden layer?

(b) (1 points) What is $\frac{\partial \widehat{y}_k}{\partial z_k^{[2]}}$? Simplify your answer in terms of $\widehat{y}$.

(c) (1 points) What is $\frac{\partial \widehat{y}_k}{\partial z_i^{[2]}}$, for $i \neq k$? Simplify your answer in terms of $\widehat{y}$.

[1]See this page for a description of popular activation functions, including LeakyReLU.

(d) (3 points) Assume that the label y has 1 at its k^{th} entry, and 0 elsewhere. What is $\frac{\partial L}{\partial z_i^{[2]}}$? Simplify your answer in terms of $\widehat{y}_i$. *Hint: Consider both cases where* $i = k$ *and* $i \neq k$.

(e) (2 points) When implementing the softmax function, we often run into a numerical stability problem. Explain this numerical stability problem in softmax function, and the modified softmax implementation to resolve this issue.

2. (40(UG)/43(PG) points) **Image Classification**

1. (5 points) Refer to the Russian Wildlife Dataset.

(a) (1 point) Download the dataset and use the following mapping as the class labels: {'amur_leopard': 0, 'amur_tiger': 1, 'birds': 2, 'black_bear': 3, 'brown_bear': 4, 'dog': 5, 'roe_deer': 6, 'sika_deer': 7, 'wild_boar': 8, 'people': 9} Perform a stratified random split of the data in the ratio 0.7:0.1:0.2 to get the train, validation and the test sets. Create a custom Dataset class for the data. Initialize Weights & Biases (WandB)(Video Tutorial).

(b) (2 points) Create data loaders for all the splits (train, val and test) using PyTorch.

(c) (2 points) Visualize the data distribution across class labels for training and validation sets.

2. (10(UG)/13(PG) Points) Training a CNN from scratch (Tutorial):

(a) (3.5 points) Create a CNN architecture with 3 Convolution Layers having a kernel size of 3×3 and padding and stride of 1. Use 32 feature maps for the first layer, 64 for the second and 128 for the last convolution layer. Use a Max pooling layer having kernel size of 4×4 with stride 4 after the first convolution layer and a Max pooling layer having kernel size of 2×2 with stride 2 after the second and third convolution layers. Finally flatten the output of the final Max pooling layer and add a classification head on top of it. Use ReLU activation functions wherever applicable.

(b) (3 points) Train the model using the Cross-Entropy Loss and Adam optimizer for 10 epochs. Use wandb to log the training and validation losses and accuracies.

(c) (0.5 points) Look at the training and validation loss plots and comment whether the model is overfitting or not.

(d) (3 points) Report the Accuracy and F1-Score on the test set. Also, log the confusion matrix using wandb.

(e) [**PG**] (3 points) For each class in the test set, visualize any 3 images that were misclassified along with the predicted class label. Analyze why the model could possibly be failing in these cases. Is this due to the fact that the image does not contain the ground truth class or it looks more similar to the predicted class or something else? Can you think of any workaround for such samples?

3. (10 points) Fine-tuning a pretrained model

(a) (3.5 points) Train another classifier with a fine-tuned Resnet-18 (pre-trained on ImageNet) architecture using the same strategy used in Question 2.2.(b) and again use wandb for logging the loss and accuracy.

(b) (0.5 points) Look at the training and validation loss plots and comment whether the model is overfitting or not.

(c) (3 points) Report the Accuracy and F1-Score on the test set. Also, log the confusion matrix using wandb.

(d) (3 points) For deep neural networks, typically, the backbone is the part of a model (initial layers) that is used to extract *feature representations* (or simply features) from the raw input data, which can then be used for classification or some other related task. These features are expressed as an n-dimensional vector, also known as a feature vector and the corresponding vector space is referred to as the feature space. As the training progresses and the classifier learns to classify the input, the data samples belonging to the same class lie closer to each other in the feature space than other data samples. For input samples from the training and validation sets, extract the feature vectors using the backbone (ResNet-18 in this case) and visualize them in the feature space using the tSNE plot in a 2-D Space. Also, visualize the tSNE plot of the validation set in a 3D-Space.

4. (10 points) Data augmentation techniques

   (a) (3.5 points) Use any 3 (or more) Data Augmentation techniques that are suitable for this problem. Remember that data augmentation techniques are used for synthetically adding more training data so that the model can train on more variety of data samples.

   (b) (3 points) Follow the same steps as in Question 2.3.(a) to train the model.

   (c) (0.5 point) Look at the training and validation loss plots now and comment if the problem of overfitting is getting resolved or not.

   (d) (3 points) Report the Accuracy and F1-Score on the test set. Also, log the confusion matrix using wandb.

5. **[BONUS]** (5 points) For each class in the test set, pick any 3 images that were misclassified using the basic CNN architecture (question 2.2). Report the euclidean distances in n-dimensional space (feature representation) of each misclassified sample from the centroid (mean) of the ground truth class as well as the predicted class. Using the same approach, analyze the euclidean distances of the same samples using the other two trained models. If the samples are correctly classified using the other models, report the distance from the centroid of the predicted class as NA. Specify the names of the ground truth and predicted class.

6. (5 points) Compare and comment on the performance of all three models.

3. (30 points) **Image Segmentation**

   1. (5 points) Download the Indian Driving Dataset. Refer to the Indian Driving Dataset Home Page or Indian Driving Dataset Paper to read about the label structure.

IDD and Cityscapes dataset masks contain multiple classes, out of which you are suggested to work for the classes "Road, Sidewalk, Person, Rider, Motorbike, Bicycle, Car, Truck, Bus, Train, Wall, Fence, Traffic Sign, Traffic Light, Pole, Building, Vegetation, Sky".

(a) (1 point) Download the dataset and create a dataloader for the same. You can ignore downloading the dataset by importing the data from the drive directly into your Colab notebook if you use Google Colab.

(b) (2 points) Visualize the data distribution across the provided dataset.

(c) (2 points) Visualize two images along with their mask for each class. Color code the classes present in the mask properly.

2. (15 points) Evaluating a segmentation model:

   (a) (4 points) Use a pre-trained DeepLabv3Plus-Pytorch model trained over the cityscapes dataset and perform inference using this model on the 30% of the given Indian Driving Dataset as the test set. The image size in the dataset is of (1920, 1080) shape; better if you resize the image to (512, 512).

   (b) (6 points) Report the classwise performance of the test set in terms of pixel-wise accuracy, dice coefficient, mAP, and IoU (Intersection Over Union). Also, report precision, and recall. Use the IoUs within the range [0, 1] with 0.1 interval size for computation of the above metrics. You may refer to this article to learn more about the evaluation of segmentation models. Include all your findings in the submitted report.

   (c) (5 points) For each class in the test set, visualize any three images with IoU ≤ 0.5 and the predicted and ground truth masks; the visualization of masks should be in proper color code same as done in the part-1 of this question. Comment on why the model could possibly be failing in these cases with the help of IoU visualizations. Is the object occluded, is it being misclassified, or is it due to the environment (surroundings) where the object is present?

3. (10 points) Analysis of results

   (a) (4 points) Create a confusion matrix, visualize it as a heat-map, and comment on anything you can infer from it.

   (b) (6 points) Analyze performance metrics (precision, recall, F1 score) for each individual class. Identify classes where the model performs well and classes where improvement is needed.

4. **[BONUS]**(6 points) Cross-Domain Analysis: In the above parts, the DeepLabV3Plus-Pytorch model is trained over the cityscapes dataset and evaluated over the IDD dataset. Now, download the Cityscapes Dataset validation set.

   (a) (2 points) Perform inference over the cityscapes dataset for the suggested classes and evaluate the confusion matrix for the same.

   (b) (2 points) Comment on the major possible differences between the confusion matrix obtained for the Cityscapes Dataset and India Driving Dataset.

   (c) (2 points) Comment on why the worst performing class performed badly, and the best class performed best using the confusion matrix.