Intro to Image Understanding (CSC420)

Assignment 2

Posted: Feb 5, 2024 **Submission Deadline : Feb 13, 11.59pm, 2024**

Instructions for submission: Please write a document (either pdf, doc, etc) with your solutions (include pictures where needed). Include your code inside the document. Please submit through MarkUs. You are expected to work on the assignment **individually**.

Max points: 15

1. [**7 points**] Implement seam carving:
   (a) [**1 point**] Compute magnitude of gradients of an image
   (b) [**4 points**] Find the connected path of pixels that has the smallest sum of gradients. A path is **valid** if it is connected (the neighboring points in the path are also neighboring pixels in the image), it starts in the first row of the image and in each step continues one row down. It finishes in the last row of the image.
   (c) [**1 point**] Remove the pixels in the path from the image. This gives you a new image with one column less.
   (d) [**1 point**] Remove a few paths with the lowest sum of gradients. Create a few examples and include them in your document.

   You can find more details about seam carving in this paper:

   S. Avidan and A. Shamir, *Seam Carving for Content-Aware Image Resizing*, SIGGRAPH 2007, `http://www.win.tue.nl/~wstahw/edu/2IV05/seamcarving.pdf`

   and this lecture:

   `http://www.cs.toronto.edu/~fidler/slides/2022Winter/CSC420/lecture4.pdf`

2. [**2 points**] Image upscaling:
   Write down the mathematical form of the convolution filter that performs upscaling of a 1D signal by a factor d. You do not need to write code. Please plot this filter (you can plot it by hand).

3. [**6 points**] Neural Network:
   (a) [**3 points**] In the main.py script we provide a training pipeline for image classification on the MNIST dataset (digit classification). Please follow the readme file to download the MNIST dataset (50MB). Implement the forward pass for One-layer MLP and the cross entropy loss function with the requirement specified in the mlp.py. In particular, you are expected to implement cross_entropy_loss_function, sigmoid, softmax functions and the __init__, forward functions for OneLayerNN class in mlp.py. You are only allowed to use numpy for this exercise.

(b) **[3 points]** Implement the backpropagation_with_gradient_descent functions for OneLayerNN class in mlp.py and running main.py to train OneLayerNN. The function is used to first compute the gradient of the loss function with respect to the weight matrix and biases, and then run gradient descent using the specified learning rate. Please submit the training curves (the accuracy for every epoch) and the best accuracy you obtained. You can play with different initializations and learning rates. You are also only allowed to use numpy for this exercise.

Additional credit [2 points]: Implement the functions __init__, forward and backpropagation_with_gradient_descent for TwoLayerNN class in mlp.py, and submit the new training curve and the accuracy you get with a two layer MLP.

You are expected to submit main.py and mlp.py for this problem.