CV Assignment 2, Image Segmentation

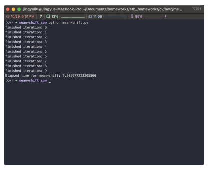
Jingyu Liu

2.4 loop-based version:

First we just calculate the distance vector using standard vector operation $torch.sqrt(torch.sum((x - X) ** 2, dim = -1))$. Gaussian kernel is also simply $torch.exp(-0.5 * (dist / bandwidth) ** 2)$ with bandwidth as the parameter. Then we update the intensity of the pixel based on weighted average where we first calculate the normalizer, and then calculate the weighted sum, and finally get the result by dividing the normalizer.

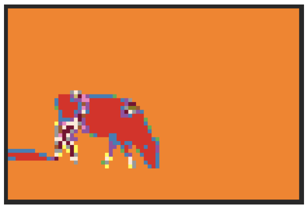
batch_based version:

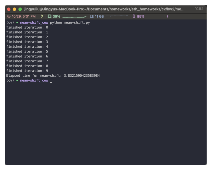
The distance function takes in the data matrix, and calulate a distance matrix which is a symmetric matrix with the entry i, j being the norm of difference between data_i and data_j. To do so, we take advantage of the broadcasting feature of torch vector addition to get a difference matrix $X.reshape(-1, 1, 3) - X.reshape(1, -1, 3)$, which will give us a $N \times N \times 3$ matrix that describes the pair-wise difference in 3 channels. ,We square, sum (over the third dimension), and sqrt to get the norm matrix, which is $N \times N$. Then we pass this to the gaussian function to get the weight matrix W. Finally, the we get an unnormalized weighted sum, $\widetilde{X} = WX$, which gets normalized to get $X' = \frac{1}{Z} \odot \widetilde{X}$ where Z is found by summing W over the second dimension and the multiplication is element wise with Z expanded to match $\widetilde{X}$ in dimension.



```
jingyuliu@Jingyus-MacBook-Pro:~/Documents/homeworks/eth_homeworks/cv/hw2/me...
10/29, 5:31 PM | 13% | 11 GB | 85%
(cv) → mean-shift_cow python mean-shift.py
finished iteration: 0
finished iteration: 1
finished iteration: 2
finished iteration: 3
finished iteration: 4
finished iteration: 5
finished iteration: 6
finished iteration: 7
finished iteration: 8
finished iteration: 9
Elapsed time for mean-shift: 7.585677223205566
(cv) → mean-shift_cow _
```

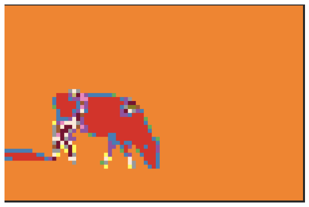
Figure 1: Naive Implementation





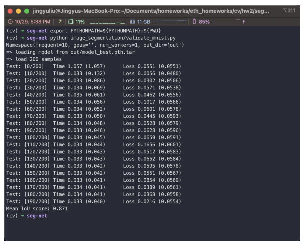
```
jingyuliu@Jingyus-MacBook-Pro:~/Documents/homeworks/eth_homeworks/cv/hw2/me...
10/29, 5:31 PM | 39% | 11 GB | 85%
(cv) → mean-shift_cow python mean-shift.py
finished iteration: 0
finished iteration: 1
finished iteration: 2
finished iteration: 3
finished iteration: 4
finished iteration: 5
finished iteration: 6
finished iteration: 7
finished iteration: 8
finished iteration: 9
Elapsed time for mean-shift: 3.8321598423583984
(cv) → mean-shift_cow _
```

Figure 2: Batch Implementation



Both versions are experimented with CPUs. The speed up is $(7.5 - 3.8) / 7.8 \approx 50\%$.

3.2



```
jingyuliu@Jingyus-MacBook-Pro:~/Documents/homeworks/eth_homeworks/cv/hw2/seg...
10/29, 5:38 PM | 11% | 11 GB | 85%
(cv) → seg-net export PYTHONPATH=${PYTHONPATH}:${PWD}
(cv) → seg-net python image_segmentation/validate_mnist.py
Namespace(frequent=10, gpus='', num_workers=1, out_dir='out')
=> loading model from out/model_best.pth.tar
=> load 200 samples
Test: [0/200]   Time 1.057 (1.057)      Loss 0.0551 (0.0551)
Test: [10/200]  Time 0.033 (0.132)      Loss 0.0056 (0.0480)
Test: [20/200]  Time 0.033 (0.086)      Loss 0.0302 (0.0506)
Test: [30/200]  Time 0.034 (0.069)      Loss 0.0571 (0.0538)
Test: [40/200]  Time 0.035 (0.061)      Loss 0.0462 (0.0556)
Test: [50/200]  Time 0.034 (0.056)      Loss 0.1017 (0.0566)
Test: [60/200]  Time 0.034 (0.052)      Loss 0.0601 (0.0578)
Test: [70/200]  Time 0.033 (0.050)      Loss 0.0445 (0.0593)
Test: [80/200]  Time 0.034 (0.048)      Loss 0.0528 (0.0579)
Test: [90/200]  Time 0.033 (0.046)      Loss 0.0628 (0.0596)
Test: [100/200] Time 0.034 (0.045)      Loss 0.0659 (0.0591)
Test: [110/200] Time 0.034 (0.044)      Loss 0.1656 (0.0601)
Test: [120/200] Time 0.033 (0.043)      Loss 0.0512 (0.0583)
Test: [130/200] Time 0.033 (0.043)      Loss 0.0652 (0.0584)
Test: [140/200] Time 0.033 (0.042)      Loss 0.0595 (0.0578)
Test: [150/200] Time 0.033 (0.042)      Loss 0.0551 (0.0567)
Test: [160/200] Time 0.033 (0.041)      Loss 0.0854 (0.0569)
Test: [170/200] Time 0.034 (0.041)      Loss 0.0389 (0.0561)
Test: [180/200] Time 0.034 (0.041)      Loss 0.0368 (0.0558)
Test: [190/200] Time 0.033 (0.040)      Loss 0.0216 (0.0554)
Mean IoU score: 0.871
(cv) → seg-net
```

Figure 3: Validation Results

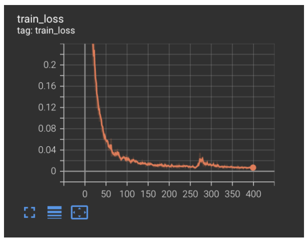


Figure 4: Train Curve

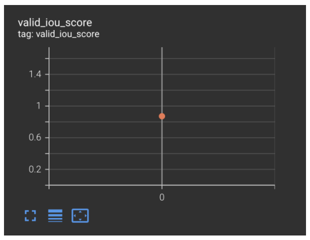


Figure 5: Validation Curve