Deep Metric Learning Course - Programming assignment

Deadline - 21st February 2022 EOD

1 Tasks

In this assignment, you are expected to study the behaviour and effect of different experimental settings on the performance of the model on 2 datasets - CUB200-2011 and CARS-196. We will be using Resnet-50 as our model pretrained from ImageNet and will be training it for a total of 40 epochs only. You will studying the following

- Effect of embedding dimension - 25 pts
- Effect of margin - 25 pts
- Effect of batch size - 25 pts
- Use of triplet loss - 25 pts

For each of these settings, you are expected to first tabulate the results and technically explain why do you think there is an increase or decrease in performance. The evaluation metric you will be reporting is Recall@1, Recall@2, MAP@R and NMI. The evaluation metric in the code I have shared with you only reports Recall@K, where $K = 1, 2, 4, 8, 16, 32$. You will find the code and compute MAP@R and NMI from pytorch-metric-learning. For each of this study, 10 pts will be given for tabulating the result and 15 pts for the explanation.

Effect of embedding dimension You can choose minimum of 3 embedding dimensions such that $d = 2^n$ and $5 \leq n \leq 10$.

Effect of margin You can choose minimum of 3 value for margin including the default $m = 0.5$ as provided in the code. Again, you need to technically explain why increasing or decreasing the value of the margin increases/ decreases performance.

Effect of batch size The default batch size is 100, so instead of understanding the performance for different batch sizes, you can report the minimum value of batch size required to run an experiment. I also want an explanation of why do you think the experiment fails if we use a batch size below this minimum value.

Triplet loss In the class, we covered both contrastive and triplet loss functions. The code I have provided evaluates Resnet-50 using contrastive loss. So, you can find the code from pytorch-metric-learning for triplet loss. Instead of repeating triplet loss for all these settings, you can use the best hyper-parameter that you obtained from studying embedding dimension, margin and batch size to evaluate Resnet-50 on triplet loss.

2 Bonus - for extra credit

We studied proxy based losses in class. You can attempt to just produce the results of Proxy Anchor using the code from from pytorch-metric-learning only on CUB200 for any one hyper-parameter i.e. choose any embedding dimension and batch size. You can train for only 30 epochs and report the result. No need for explanation, just summarize your result.