CMP 719 - Computer Vision Assignment 1

Due Date : December 27th 2019

Classifying Dog Images

In this assignment, you are going to implement a classifier using Convolutional Neural Networks(CNN)[4]. As a dataset, you will use a subset of Stanford Dogs Dataset[1], which have the following 20 classes; *Chihuahua, Irish_terrier, Kerry_blue_terrier, Australian_terrier, Norfolk_terrier, Yorkshire_terrier, African_hunting_dog, Pug, Rottweiler, Redbone, English_foxhound, Shih-Tzu, Sussex_spaniel, German_shepherd, French_bulldog, malamute, Eskimo_dog, Siberian_husky, Chow, Dingo.*

The assignment contains 3 parts. In the first part, you are going to split the dataset into train, validation and test sets and implement your own data loader. In the second part, you will train a classifier from scratch. In the last part, you will fine tune a classifier. *It is important to note that in each part you are expected to give your reasonings in detail.*

Goals of this Assignment

1. Implement a data loader in Pytorch
2. Exp1: Design a network from scratch
3. Exp2: Fine-tune a pre-trained model
4. Analyze the findings of each experiment

The Dataset :

The dataset can be accessed via the link.

Framework :

You are advised to use the recent stable version of Pytorch[2], but if you will more confident on other platforms, you are welcome to use them as well.

Part 1 - Preparing your Data

In this part you are expected to split your dataset into 3 sets: train, validation, and test sets. Give the details on how you split dataset, and explain your reasoning. The distribution in your train and validation sets will impact your results. At the end, your dataset will have 20 classes and train/validation/test splits.

After that, you need to implement your own data loader for the dataset[3]. How did you implement it? Do you think you need to shuffle your training data in the training process? Explore *data augmentation*. Are you using any *data augmentation* techniques?

Part 2 - Training a Classifier from Scratch

In this part you are expected to train a network from scratch. You should first define the components of your model;

1. How many layers are there in your network? How did you choose the number of layers? Do not forget to give parameter details of the layers such as; in_channels, out_channels, stride etc..
2. Which loss function did you use and why did you choose it?
3. Which optimization algorithm are you using? How did you decide on the optimization technique?

A simple architecture might contain 3 convolutional layers and 2 fully connected layers. After each convolutional layer you might add a *batch normalization layer*,or *pooling layer*, and apply an activation function. You might also consider adding *dropout layer* in to your network. Please don’t forget to indicate your reasonings when you add a component to your network.

Now it is time to select hyperparameters for training. Please indicate how you tune hyperparameters like; *number of epochs*, *learning rate*, *batch-size* etc. **Please don’t forget to report your model’s behavior with respect to the selected hyperparameters.**

You need to report your train and validation loss curves, classification accuracy and confusion matrix. Do you think you can improve the performance [7, 8] of your model? How could you do that?

Part 3 - Transfer Learning in Convolutional Neural Networks

In this part, you are expected to take a pre-trained model on ImageNet[5] and finetune it using Dogs dataset. Do you think it is a good idea to finetune a pre-trained model? What might be the possible advantages of this approach? On the other hand, why did we choose to use a pre-trained model specifically in ImageNet dataset?

For this assignment, you will finetune ResNet-18[6] model. Download the pretrained model from [6]. What are the differences between the two networks? Report your reasoning on hyperparameter selection process for training, as given in Section 2.

Evaluate your trained model in the test set and plot the confusion matrix. What approaches do you think might help to improve the performance?

In the end, you are expected to compare your results in Part-2 and Part-3. Please state your observations clearly and precisely. In your quest, have you faced overfitting and/or underfitting problem? If so, how did you solve it?

Important Note : Please do not forget to use the validation set for tuning and give your accuracy results on the test set. Also note that analyzing your results is the main consideration of this assignment. Plotting loss/accuracy graphs might also help you in the hyperparameter tuning process.

What to Hand In

1. README.txt (includes details of your implementation, version of your environment etc.)
2. src/ (includes all of your codes)
3. report.pdf

Archive this folder as <student_name>.zip and send it to nazli@cs.hacettepe.edu.tr

Academic Integrity

All work on assignments must be done individually unless stated otherwise. You are encouraged to discuss with your classmates about the given assignments, but these discussions should be carried out in an abstract way. That is, discussions related to a particular solution to a specific problem (either in actual code or in the pseudocode) will not be tolerated. In short, turning in someone else's work, in whole or in part, as your own will be considered as a violation of academic integrity. Please note that the former condition also holds for the material found on the web as everything on the web has been written by someone else. You have to keep your implementation until it is evaluated.

Useful Links

1. https://pytorch.org/tutorials/beginner/transfer_learning_tutorial.html
2. https://github.com/pytorch/examples/blob/master/imagenet/main.py
3. https://www.deeplearningbook.org/

References

1. http://vision.stanford.edu/aditya86/ImageNetDogs/
2. https://pytorch.org/
3. https://pytorch.org/tutorials/beginner/data_loading_tutorial.html
4. https://www.deeplearningbook.org/contents/convnets.html
5. Olga Russakovsky*, Jia Deng*, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg and Li Fei-Fei. (* = equal contribution) **ImageNet Large Scale Visual Recognition Challenge**. *IJCV*, 2015.
6. https://pytorch.org/docs/stable/torchvision/models.html
7. https://www.deeplearningbook.org/contents/optimization.html
8. https://www.deeplearningbook.org/contents/regularization.html
9. https://cs230-stanford.github.io/