Warm Up Assignments

Rishitosh Kumar Singh
Arizona State University
rksing18@asu.edu

Abstract

This report is part of the assignment that is due on 21st Jan 2024. This report consists of two practice problems and by completing these problems I got to know how datasets and dataloaders work in Pytorch, and how to train and test a neural network with Pytorch.

1. Introduction

This report documents the warm-up assignments and will be refined with each subsequent task completion. Additionally, it will incorporate details of new experiments undertaken in each assignment. The overarching goal is to achieve State-of-the-Art (SOTA) performance by employing a diverse array of techniques. A dedicated section labeled "To-Do" will delineate pending tasks for discussion and action.

2. Backbone models

Training a model from scratch consumes substantial resources, including data and time, which may not always be feasible. In scenarios where large datasets are lacking, transfer learning emerges as a practical solution. Transfer learning involves leveraging a pre-trained network with existing weights and fine-tuning it on a new dataset. Typically, backbone models are utilized in the initial layers as they excel at capturing fundamental features. Following these backbone networks, additional layers such as a linear layer or another classifier can be incorporated to adapt to the nuances of the new dataset.

2.1. ResNet

ResNet, an abbreviation for Residual Network, was pioneered by Kaiming He et al. [1]. Its hallmark innovation lies in the incorporation of residual connections, a breakthrough technique that remains integral in contemporary state-of-the-art networks, transcending beyond the realm of computer vision tasks. ResNet manifests in various versions, distinguished by the number of layers, including ResNet18, ResNet50, and beyond. Its widespread adoption stems from its capacity to facilitate the utilization of extremely deep networks, a feat made possible by the introduction of residual connections.

ResNet models are trained on ImageNet dataset, thus it have 1000 output nodes. In order to use it for our problems, we have to append few linear layers after this backbone layer.

3. Read/Write Images

3.1. Problem

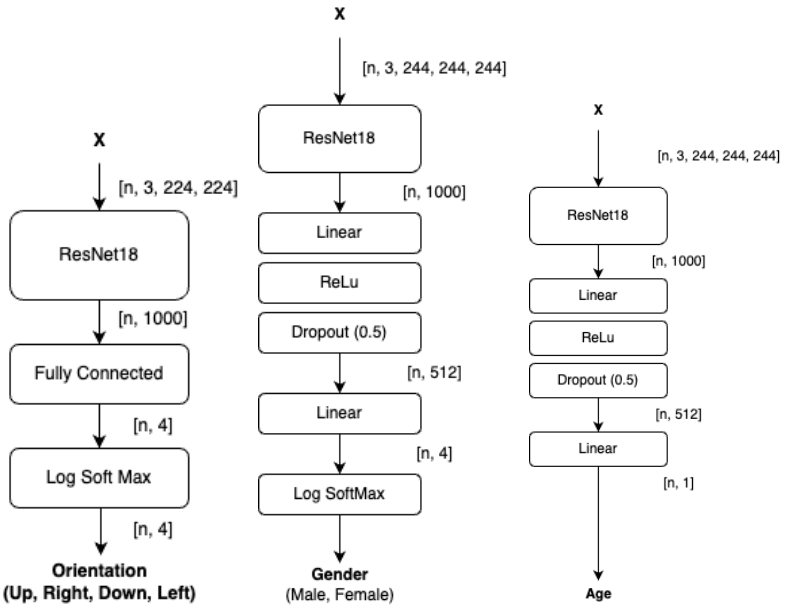
In this exercise, we were provided with two datasets. The objective was to proficiently load the data using PyTorch's dataloaders and dataset APIs, and subsequently, successfully generate matplotlib plots by writing the images.

3.2. Solution

To load the provided dataset, custom PyTorch dataset classes were crafted. These classes offer flexibility to handle datasets of various types and formats. The torchvision.io.read_image function is employed for reading PNG/JPG images, while the pydicom library is utilized for reading DICOM data.

Transformations

- When PNG/JPG images were loaded, a resize transformation was applied to change the images to a size of 256x256. Figure 2a displays a batch of 4 images from the PNG/JPG dataset.
- After loading DICOM data, a sequence of transformations was employed. At first, the image type is converted from uint16 to uint8, because PyTorch resize function does not support uint16 dtype. Subsequently, the tensor is converted to a PIL image, and the image is resized. Following the dataset loading, all the images are plotted in a matplotlib plot. Figure 2b displays a batch of 4 images from the DICOM dataset.



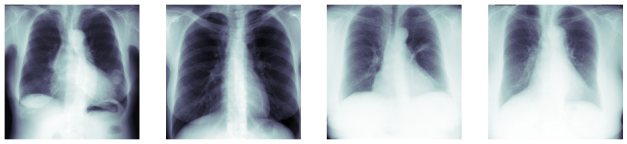
(a) Orientation Model (b) Gender Model (c) Age Model

4. Orientation

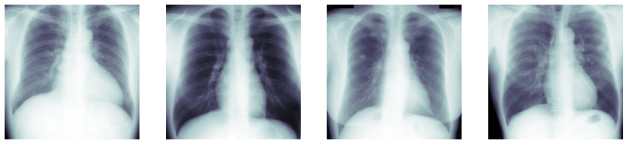
In this introductory practice exercise, chest X-rays were provided, and the task involved training/fine-tuning a model to determine the orientation of these X-rays.

4.1. Dataset

The provided dataset consists of RGB X-ray images with 948 training samples depicting various orientations and an additional 40 samples for testing. All images are of size 128x128 and are oriented either as up (0), right (1), down (2), or left (4). Due to the absence of a dedicated validation set in the given dataset, I have randomly partitioned the training samples into training and validation sets in 8:2 ratio.



(a) PNG/JPG dataset



(b) DICOM dataset

Figure 2. XRay images batch

4.2. Model Used

For this task, I utilized ResNet18 as the backbone model with pre-trained weights from ImageNet. Since ResNet18 was originally trained on ImageNet, its output tensor has a size of (n, 1000). However, in the given problem, we are specifically dealing with 4 classes (up, right, down, left). Therefore, I introduced a fully connected layer followed by a log-softmax layer to obtain probabilities for each class. Refer to Figure 1a for a graphical representation of the model architecture.

4.3. Training

The model is trained using the categorical cross-entropy loss function with stochastic gradient descent as the optimizer. As the Log-Softmax layer is already included in the model, the torch.nn.NLLLoss function is employed instead of a torch.nn.CrossEntropyLoss. Four experiments were conducted with learning rates of 0.001, 0.003, 0.01, and 0.03, and the results were averaged. Refer to Figure 3 for a demonstration of the learning curve from one of the experiments.

4.4. Results

After fine-tuning the model with the ResNet18 backbone network, the model was transitioned to testing and evaluated on the provided test samples. On average, it achieved an accuracy of 96.87%, and in some experiments, 100% accuracy was also attained. Figure 4 showcases some of the test samples along with their model predictions and target classes.

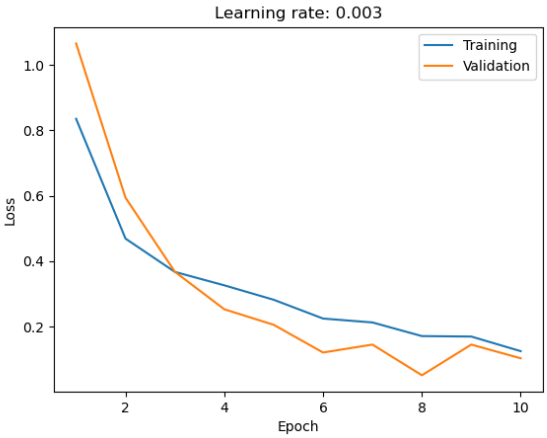


Figure 3. Learning curve

5. Gender

In this preliminary practice activity, we were given chest X-ray images of patients using which we have to either train or fine-tune a model to determine the gender of patients.

5.1. Dataset

We were given three datasets for classifying gender of the patients; grayscale, RGB, and index. Out of these, I have used Gender01_RGB for the experiments. The provided dataset 154 training samples of patient X-Rays and an additional 93 samples for testing trained model. The images in this dataset were color image with shape 256x256. Each training samples have either label "male", or "female". Due to the absence of a dedicated validation set in this dataset too, I have randomly partitioned the training samples into training and validation sets in 9:1 ratio.

5.2. Training

In this task, we are dealing with two labels: male and female. Therefore, we constructed a model with a graph described in Fig. 1b. A log-softmax layer is employed to obtain probabilities for each class. The model is trained using the categorical cross-entropy loss function with Adam optimizer having a learning rate of 0.003. We utilize the torch.nn.NLLLoss function, which is commonly used for classification problems, as our loss function. Additionally, early stopping with a patience of 3 epochs is implemented to halt the model training if the validation loss continues to increase for 3 consecutive epochs. Figure 5 illustrates the learning curve of the training and validation sets.

5.3. Results

After fine-tuning the model, it was evaluated on the provided test samples on which it achieved an accuracy of 90.32%, and AUC of 90.34. Figure 6 showcases some of the test samples along with their model predictions and target classes.

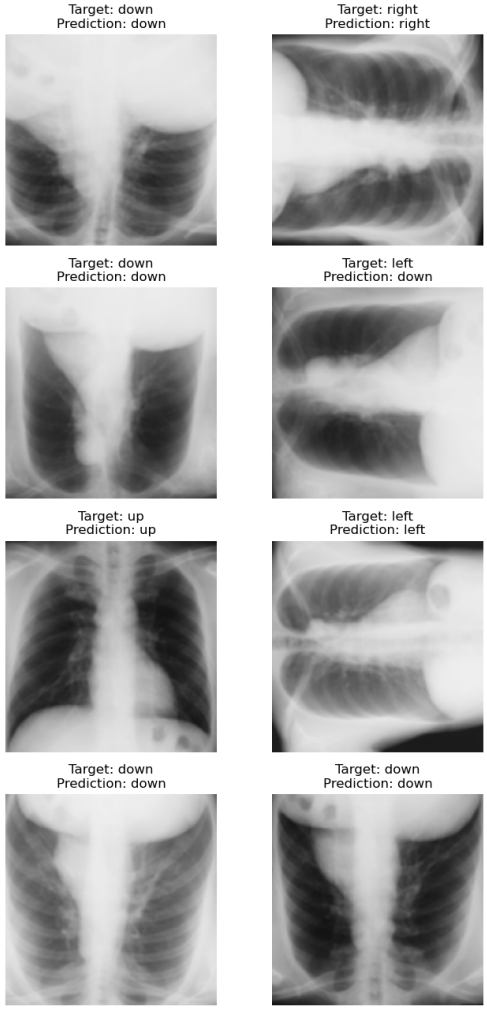


Figure 4. Predictions on few samples of test data

6. Age

We were given chest X-ray images of patients using which we have to fine-tune a model to determine the age of patients.

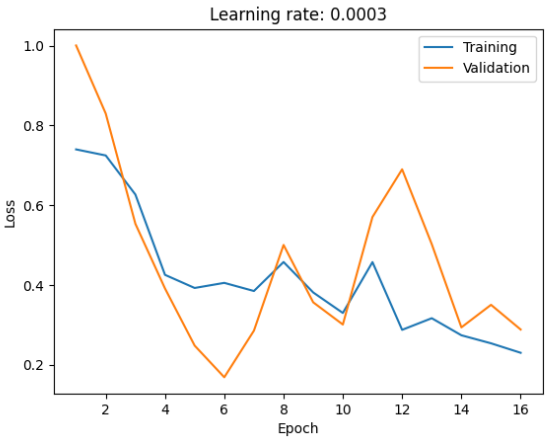


Figure 5. Gender model learning curve on training and validation sets

6.1. Dataset

The provided dataset have 80 training x-ray images and 165 testing images. As there was no validtion set, I have randomly splited the training dataset into validation and training set in ration 1:9 respictively. The images in this dataset were color image with shape 2048x2048, which is resized to image of size 224x224.

6.2. Training

This is a regression problem, thus a few changes is required to the model used in classifying gender of patients. The last Log Softmax layer is not required, thus eliminated from the model graph as shown in Fig. 1c. The model is trained using mean ssquare error function with Adam optimizer having a learning rate of 0.003. Additionally, in this experiment too, early stopping with a patience of 3 epochs is implemented to halt the model training. Figure 5 illustrates the learning curve of the training and validation sets.

6.3. Results

A total of 2 experiments were conducted with same hyperparameters and results are averaged. After fine-tuning the model, it was evaluated on the provided test samples on which it achieved a MAE of 5.57 Figure 8 showcases some of the test samples along with their model predictions.

7. To-Do

In the latest assignment, we have to use GradCAM to visualize the feature maps and see how our model is able to classify images into different classes. This task is pending, and I will complete this by next assignment.

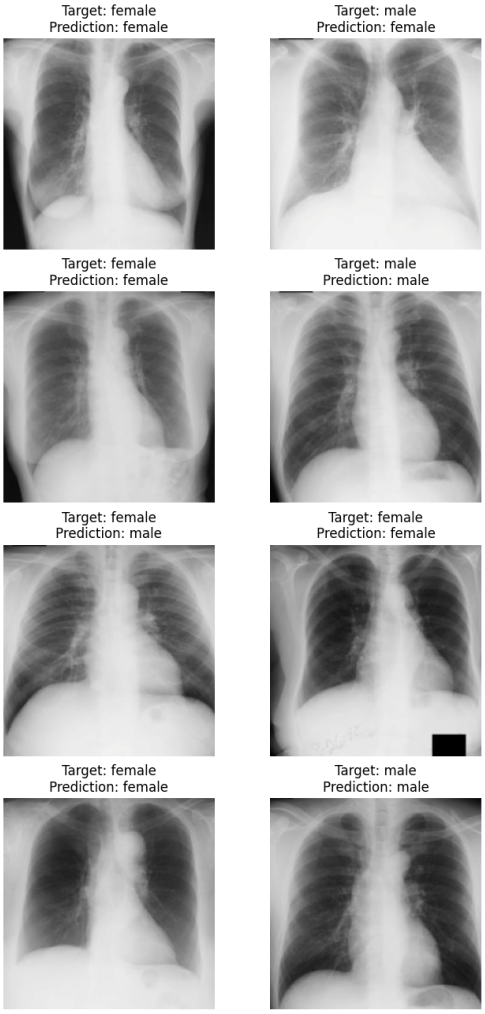


Figure 6. Gender predictions on few samples of test data

References

[1] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 1

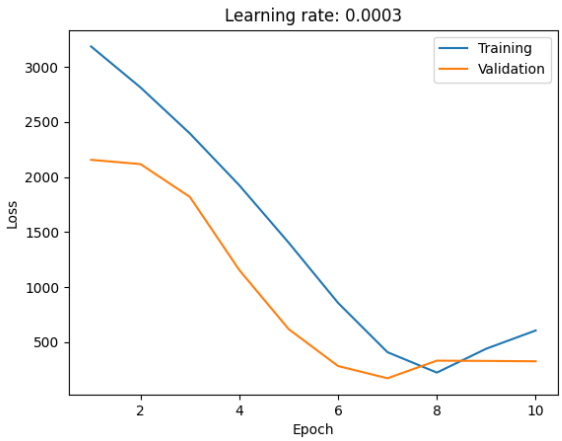


Figure 7. Age model learning curve on training and validation sets

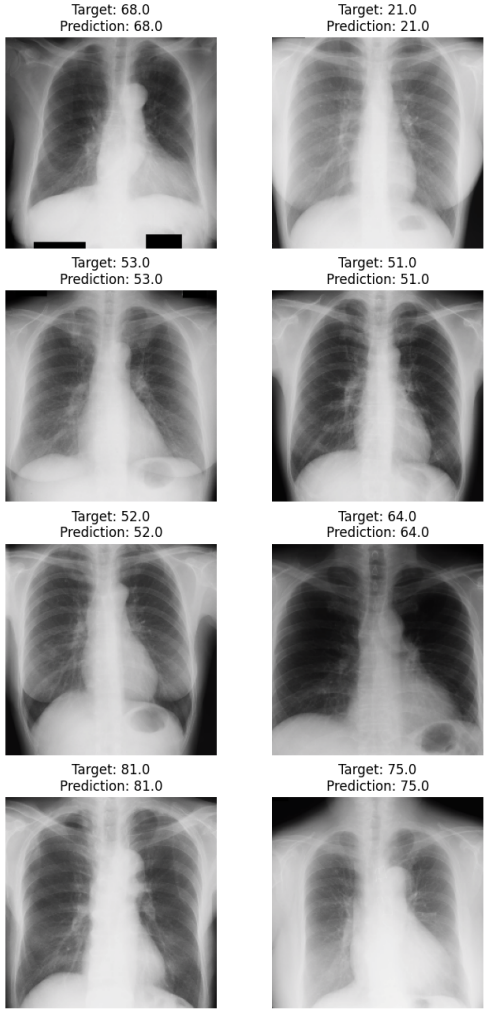


Figure 8. Age predictions on few samples of test data