A3_Q2-2024

August 14, 2024

0.1 Question 2: Animal classification (15 marks)

For this question, we will use the Animal (https://cloudstor.aarnet.edu.au/plus/s/cZYtNAeVhWD6uBX) dataset. This dataset contains images of 151 different animals.

The dataset contains a total of 6270 images corresponding to the name of animal types.

All images are RGB images of 224 pixels wide by 224 pixels high in .jpg format. The images are separated in 151 folders according to their respective class.

The task is to categorize each animal into one of 151 categories.

We provide baseline code that includes the following features:

- Loading and Analysing the dataset using torchvision.
- Defining a simple convolutional neural network.
- How to use existing loss function for the model learning.
- Train the network on the training data.
- Test the trained network on the testing data.

The following changes could be considered:

1. "Transfer" Learning (ie use a model pre-trained another dataset)
2. Change of advanced training parameters: Learning Rate, Optimizer, Batch-size, Number of Max Epochs, and Drop-out.
3. Use of a new loss function.
4. Data augmentation
5. Architectural Changes: Batch Normalization, Residual layers, etc.
6. Others - please ask us on the Discussion Forums if you're not sure about an idea!

Your code should be modified from the provided baseline. A pdf report of a maximum of two pages is required to explain the changes you made from the baseline, why you chose those changes, and the improvements they achieved.

0.1.1 Marking Rules:

We will mark this question based on the final test accuracy on testing images and your report.

Final mark (out of 50) = acc_mark + efficiency mark + report mark

Acc_mark 10:

We will rank all the submission results based on their test accuracy. Zero improvement over the baseline yields 0 marks. Maximum improvement over the baseline will yield 10 marks. There will be a sliding scale applied in between.

Efficiency mark 10:

Efficiency considers not only the accuracy, but the computational cost of running the model (flops: https://en.wikipedia.org/wiki/FLOPS). Efficiency for our purposes is defined to be the ratio of accuracy (in %) to Gflops. Please report the computational cost for your final model and include the efficiency calculation in your report. Maximum improvement over the baseline will yield 10 marks. Zero improvement over the baseline yields zero marks, with a sliding scale in between.

Report mark 30:

Your report should comprise: 1. An introduction showing your understanding of the task and of the baseline model: [10 marks]

2. A description of how you have modified aspects of the system to improve performance. [10 marks]

A recommended way to present a summary of this is via an "ablation study" table, eg:

Method1	Method2	Method3	Accuracy
N	N	N	60%
Y	N	N	65%
Y	Y	N	77%
Y	Y	Y	82%

3. Explanation of the methods for reducing the computational cost and/or improve the trade-off between accuracy and cost: [5 marks]

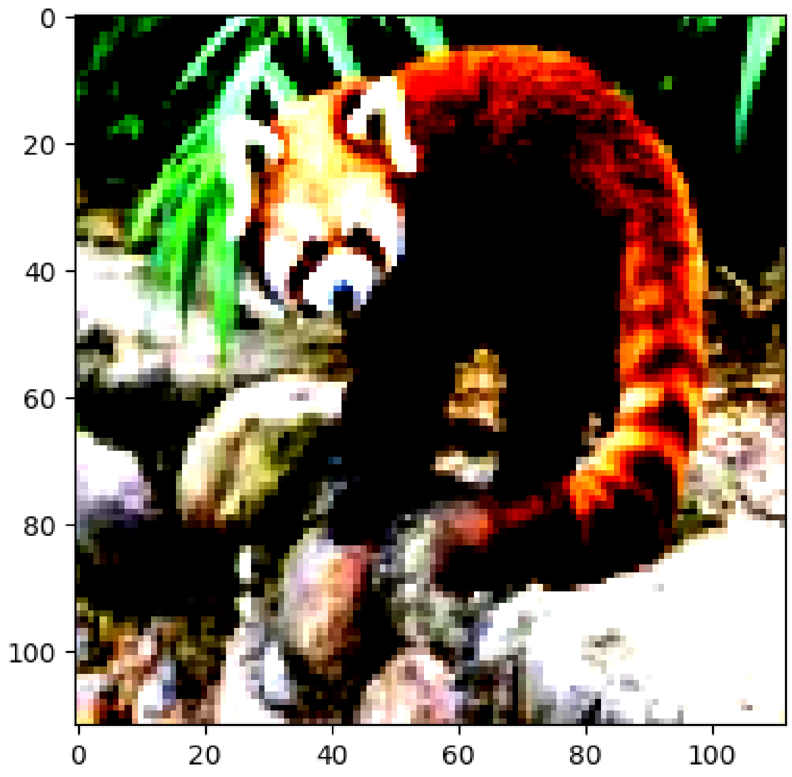
4. Limitations/Conclusions: [5 marks]

```
Size of training dataset : 6270
```

```
torch.Size([3, 112, 112])
```

```
Clipping input data to the valid range for imshow with RGB data ([0..1] for
floats or [0..255] for integers). Got range [-2.1745567..2.3031092].
```

```
Label:  ailurus-fulgens (5)
```



```
(5330, 313, 627)
```

```
Clipping input data to the valid range for imshow with RGB data ([0..1] for
floats or [0..255] for integers). Got range [-2.2467773..2.3572743].
```

```
cuda
```

```
ConvolutionalNetwork(
  (conv1): Conv2d(3, 64, kernel_size=(5, 5), stride=(1, 1))
  (conv2): Conv2d(64, 128, kernel_size=(3, 3), stride=(1, 1))
  (conv3): Conv2d(128, 128, kernel_size=(3, 3), stride=(1, 1))
  (conv4): Conv2d(128, 128, kernel_size=(3, 3), stride=(1, 1))
  (fc1): Linear(in_features=3200, out_features=151, bias=True)
)
```

```
DOConvolutionalNetwork(
  (conv1): Conv2d(3, 64, kernel_size=(5, 5), stride=(1, 1))
  (conv2): Conv2d(64, 128, kernel_size=(3, 3), stride=(1, 1))
  (conv3): Conv2d(128, 128, kernel_size=(3, 3), stride=(1, 1))
  (conv4): Conv2d(128, 128, kernel_size=(3, 3), stride=(1, 1))
  (drop): Dropout2d(p=0.2, inplace=False)
  (fc1): Linear(in_features=3200, out_features=151, bias=True)
)

SmallerDOConvolutionalNetwork(
  (conv1): Conv2d(3, 64, kernel_size=(5, 5), stride=(1, 1))
  (conv2): Conv2d(64, 128, kernel_size=(3, 3), stride=(1, 1))
  (conv3): Conv2d(128, 128, kernel_size=(3, 3), stride=(1, 1))
  (drop): Dropout2d(p=0.2, inplace=False)
  (fc1): Linear(in_features=3200, out_features=151, bias=True)
)
```

```
images.shape: torch.Size([16, 3, 112, 112])
out.shape: torch.Size([16, 151])
out[0]: tensor([-5.0230, -4.9953, -4.9554, -5.0577, -5.0879, -4.9579, -5.0188,
-4.9894,
        -5.0599, -5.0163, -5.0539, -4.9913, -4.9857, -5.0409, -5.0310, -5.0157,
        -5.0245, -5.0169, -5.0365, -5.0875, -5.0270, -5.0374, -5.0112, -4.9949,
        -4.9988, -4.9719, -4.9550, -5.0776, -5.0188, -4.9772, -5.0024, -5.0198,
        -5.0638, -5.0260, -5.0068, -4.9895, -5.0360, -5.0093, -4.9352, -5.0104,
        -5.0441, -5.0583, -5.0315, -5.0521, -4.9614, -5.0777, -4.9949, -5.0993,
        -5.0077, -5.0149, -5.0059, -5.0456, -4.9595, -5.0615, -5.0447, -5.0477,
        -5.0479, -5.0515, -5.0259, -4.9723, -5.0579, -4.9977, -5.0147, -5.0159,
        -5.0510, -4.9851, -5.0430, -5.0367, -5.0584, -5.0355, -5.0562, -5.0008,
        -5.0364, -5.0926, -5.0348, -5.0814, -5.0266, -4.9945, -5.0116, -4.9721,
        -5.0838, -4.9965, -5.0464, -4.9896, -5.0034, -4.9707, -5.0356, -5.0457,
        -5.0201, -5.0624, -5.0166, -5.0077, -4.9982, -4.9898, -5.0005, -5.0049,
        -5.0133, -4.9981, -5.0875, -5.0521, -5.0416, -4.9875, -4.9918, -4.9795,
        -4.9953, -4.9941, -5.0496, -4.9952, -4.9475, -5.0382, -5.0724, -5.0452,
        -5.0809, -5.0133, -5.0103, -5.0372, -4.9514, -4.9338, -4.9830, -5.0384,
        -5.0050, -5.0667, -5.0189, -4.9990, -5.0019, -4.9941, -4.9857, -5.0004,
        -4.9441, -5.0051, -4.9554, -4.9784, -5.0311, -5.0086, -5.0350, -5.0331,
        -5.0215, -5.0902, -5.0069, -5.0139, -4.9984, -5.0382, -5.0160, -4.9698,
        -4.9807, -4.9903, -4.9908, -5.0896, -4.9861, -5.0555, -4.9645],
       device='cuda:0', grad_fn=<SelectBackward0>)
images.shape: torch.Size([16, 3, 112, 112])
out.shape: torch.Size([16, 151])
out[0]: tensor([-4.9902, -5.0081, -5.0600, -5.0530, -5.0296, -4.9609, -4.9967,
-5.0261,
        -5.0063, -5.0800, -5.0384, -5.0035, -5.0735, -5.0201, -5.0061, -4.9693,
        -5.0172, -4.9353, -5.0316, -5.0683, -4.9832, -5.0173, -4.9584, -4.9613,
        -5.0234, -5.1189, -5.0464, -5.0057, -5.0162, -4.9257, -4.9495, -5.0450,
        -5.0043, -5.0264, -5.0149, -5.1097, -5.0439, -5.0193, -5.0353, -5.0409,
        -5.0508, -5.0441, -5.0081, -4.9794, -5.0003, -5.0187, -5.1316, -5.0494,
        -5.0324, -5.0245, -5.0338, -5.0314, -4.9725, -4.9989, -5.0436, -5.0130,
        -5.0046, -5.0744, -4.9959, -5.0081, -5.0688, -4.9437, -5.0349, -4.9831,
        -4.9887, -5.1062, -5.0362, -4.9701, -4.9577, -5.0467, -5.0080, -4.9374,
        -5.0556, -5.0479, -5.0388, -5.0516, -5.0012, -5.0287, -4.9453, -4.9909,
        -5.0261, -4.9571, -5.0503, -5.0512, -4.9410, -4.9535, -5.0920, -4.9930,
        -5.0528, -5.0432, -4.9743, -5.0628, -4.9948, -5.0548, -5.0857, -5.0600,
        -4.9722, -4.9664, -4.9808, -4.9959, -5.0155, -5.0338, -5.0825, -5.0124,
        -5.0315, -4.9854, -4.9828, -5.0224, -5.0216, -5.0314, -5.0752, -5.0465,
        -5.0171, -4.9847, -4.9854, -5.0832, -4.9835, -4.9480, -4.9874, -5.0470,
        -4.9943, -5.0214, -5.0210, -5.0449, -4.9939, -5.0275, -5.0275, -5.0455,
        -5.0731, -4.9981, -5.0926, -5.0459, -5.0205, -4.9826, -5.0558, -5.0384,
        -5.0025, -5.0306, -5.0569, -5.0245, -5.0504, -5.0174, -4.9724, -4.9903,
        -4.9991, -4.9287, -5.0236, -4.9795, -4.9843, -5.0675, -4.9594],
       device='cuda:0', grad_fn=<SelectBackward0>)
images.shape: torch.Size([16, 3, 112, 112])
out.shape: torch.Size([16, 151])
out[0]: tensor([-5.1607, -5.0587, -5.0836, -5.0919, -5.0264, -5.1140, -4.9258,
-5.0981,
        -5.0320, -4.9561, -4.9348, -4.9151, -5.1087, -5.0263, -5.0944, -5.0623,
        -4.9975, -4.9832, -4.9561, -4.9462, -5.0844, -5.0230, -4.8776, -5.0206,
        -5.0525, -5.2297, -4.8701, -5.0183, -4.8798, -5.0598, -5.0335, -4.9774,
        -4.8095, -5.1403, -4.9945, -5.0210, -5.1242, -5.0299, -5.0697, -5.1245,
        -5.0545, -4.9492, -4.9583, -5.0503, -5.0448, -5.0097, -5.0260, -4.9391,
        -5.0115, -5.0575, -5.0568, -4.9893, -5.0315, -4.9576, -4.8958, -4.9636,
        -5.0183, -5.0524, -4.9802, -5.0332, -5.1174, -5.0664, -4.9505, -4.9341,
        -5.0903, -4.9573, -5.2270, -5.0280, -5.0678, -5.0608, -5.0002, -4.9936,
        -4.9913, -4.9673, -5.0312, -5.1029, -4.9332, -4.9581, -4.9466, -5.0456,
        -5.0173, -5.0429, -5.0543, -4.9141, -5.0370, -5.1372, -5.0900, -4.9994,
        -4.9074, -4.9997, -5.0877, -4.9177, -5.0246, -5.0713, -5.0183, -5.1100,
        -4.9007, -4.9632, -5.0251, -4.8202, -4.9423, -5.0390, -5.0769, -4.9304,
        -5.2012, -4.9551, -4.9051, -4.9683, -5.0835, -5.0496, -5.0464, -5.0310,
        -5.0125, -5.0651, -5.0243, -5.0164, -5.0550, -5.1438, -5.0607, -4.9869,
        -4.9686, -5.1282, -4.9994, -4.9154, -5.0288, -4.8743, -5.0695, -4.9287,
        -5.0409, -5.0249, -5.0642, -5.1578, -4.9837, -4.9013, -5.0942, -5.1189,
        -5.1839, -5.0537, -4.9076, -4.8411, -4.8051, -5.3101, -5.0561, -4.9631,
        -4.8471, -5.1329, -5.1931, -4.9445, -5.0662, -5.1504, -5.1154],
       device='cuda:0', grad_fn=<SelectBackward0>)
images.shape: torch.Size([16, 3, 112, 112])
out.shape: torch.Size([16, 151])
out[0]: tensor([-5.0852, -4.9648, -5.1995, -5.2560, -5.0407, -5.2802, -4.8265,
-4.8388,
        -4.9967, -5.1484, -5.1654, -5.0979, -4.8123, -5.0436, -5.1637, -4.8440,
        -4.7746, -5.0610, -5.2121, -4.6985, -4.9423, -5.1221, -5.0025, -4.9018,
        -5.2633, -5.0981, -4.9795, -5.1394, -5.0343, -4.9789, -5.1657, -4.7976,
        -4.8428, -4.6316, -4.8905, -5.0221, -4.8042, -5.1657, -5.0406, -5.0719,
        -5.3497, -5.0020, -5.0742, -4.9149, -5.1639, -4.9434, -5.0562, -4.8945,
        -5.0835, -4.9176, -4.5704, -4.8886, -5.2589, -5.1508, -5.2251, -5.2332,
        -5.0083, -5.2899, -5.0045, -4.7461, -5.2313, -4.7975, -4.8277, -5.1003,
        -5.0156, -4.9453, -5.1930, -4.8382, -5.1290, -4.9679, -4.9207, -5.0480,
        -5.1628, -4.9940, -5.1021, -5.2223, -5.0514, -4.9608, -5.1815, -4.9528,
        -5.2064, -4.8329, -5.0557, -5.0839, -5.1217, -4.9232, -5.3418, -5.0347,
        -4.8277, -5.0385, -5.0155, -5.1693, -5.0269, -4.8522, -4.9634, -4.9905,
        -5.1372, -4.9899, -5.5450, -5.1915, -5.0705, -4.9674, -5.2769, -5.1791,
        -5.3866, -5.1854, -4.8707, -4.9977, -4.7099, -5.2200, -4.9159, -5.0970,
        -4.8775, -5.1767, -5.1747, -4.9962, -4.7719, -5.0108, -5.0305, -5.0176,
        -4.9523, -4.6892, -4.7284, -5.0978, -4.8331, -5.2294, -4.9283, -5.0867,
        -4.9595, -5.1687, -5.1785, -4.9567, -5.2002, -4.9276, -4.9570, -4.8785,
        -5.0459, -5.1782, -5.0305, -4.8938, -5.0696, -5.0399, -5.0956, -5.0303,
        -4.8860, -5.1053, -5.1094, -5.1468, -4.7071, -5.2171, -5.1440],
       device='cuda:0', grad_fn=<SelectBackward0>)
```

```
Starting evaluation for baseline (10 epochs): [{'val_loss': 5.019043922424316,
'val_acc': 0.015625}]
Starting evaluation for baseline (50 epochs): [{'val_loss': 5.019298553466797,
'val_acc': 0.012500000186264515}]
Starting evaluation for dropout model: [{'val_loss': 5.01693058013916,
'val_acc': 0.02500000037252903}]
Starting evaluation for smaller dropout model: [{'val_loss': 5.0107622146606445,
'val_acc': 0.015625}]
```

```
627
```

```
Enabling notebook extension jupyter-js-widgets/extension…
      - Validating: OK
```

```
  0%|          | 0/334 [00:00<?, ?it/s]
```

```
Epoch [0], train_loss: 4.7812, val_loss: 4.5188, val_acc: 0.0813
```

```
  0%|          | 0/334 [00:00<?, ?it/s]
```

```
Epoch [1], train_loss: 4.2279, val_loss: 4.1728, val_acc: 0.1705
```

```
  0%|          | 0/334 [00:00<?, ?it/s]
```

```
Epoch [2], train_loss: 3.7033, val_loss: 3.9535, val_acc: 0.2771
```

```
  0%|          | 0/334 [00:00<?, ?it/s]
```

```
Epoch [3], train_loss: 3.2529, val_loss: 3.8748, val_acc: 0.3035
```

```
  0%|          | 0/334 [00:00<?, ?it/s]
```

```
Epoch [4], train_loss: 2.7991, val_loss: 3.8950, val_acc: 0.3597
```

```
  0%|          | 0/334 [00:00<?, ?it/s]
```

```
Epoch [5], train_loss: 2.4509, val_loss: 4.0764, val_acc: 0.3684
```

```
  0%|          | 0/334 [00:00<?, ?it/s]
```

```
Epoch [6], train_loss: 2.1071, val_loss: 4.0610, val_acc: 0.3615
```

```
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [7], train_loss: 1.7828, val_loss: 4.4330, val_acc: 0.3410
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [8], train_loss: 1.5228, val_loss: 4.8786, val_acc: 0.3840
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [9], train_loss: 1.2671, val_loss: 5.3668, val_acc: 0.3653
627
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [0], train_loss: 4.9357, val_loss: 4.9695, val_acc: 0.0556
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [1], train_loss: 4.4560, val_loss: 4.2928, val_acc: 0.1792
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [2], train_loss: 3.9413, val_loss: 4.0844, val_acc: 0.2316
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [3], train_loss: 3.5854, val_loss: 4.0363, val_acc: 0.2441
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [4], train_loss: 3.2344, val_loss: 3.9848, val_acc: 0.2823
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [5], train_loss: 2.9493, val_loss: 4.2111, val_acc: 0.2972
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [6], train_loss: 2.6812, val_loss: 4.1448, val_acc: 0.2760
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [7], train_loss: 2.3886, val_loss: 4.3971, val_acc: 0.2910
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [8], train_loss: 2.1225, val_loss: 4.9992, val_acc: 0.2823
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [9], train_loss: 1.8942, val_loss: 5.0507, val_acc: 0.2823
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [10], train_loss: 1.6371, val_loss: 5.3860, val_acc: 0.2847
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [11], train_loss: 1.3909, val_loss: 5.6965, val_acc: 0.2674
  0%|          | 0/334 [00:00<?, ?it/s]
```

```
Epoch [12], train_loss: 1.2448, val_loss: 6.1128, val_acc: 0.3042
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [13], train_loss: 1.0730, val_loss: 6.4605, val_acc: 0.2691
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [14], train_loss: 0.8992, val_loss: 7.1500, val_acc: 0.2878
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [15], train_loss: 0.8060, val_loss: 7.7612, val_acc: 0.2580
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [16], train_loss: 0.7000, val_loss: 7.9697, val_acc: 0.2823
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [17], train_loss: 0.6071, val_loss: 8.8796, val_acc: 0.2559
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [18], train_loss: 0.5288, val_loss: 9.1917, val_acc: 0.2785
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [19], train_loss: 0.5121, val_loss: 9.4462, val_acc: 0.2667
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [20], train_loss: 0.3974, val_loss: 9.6197, val_acc: 0.2840
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [21], train_loss: 0.3882, val_loss: 10.7523, val_acc: 0.3003
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [22], train_loss: 0.3736, val_loss: 11.2352, val_acc: 0.2892
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [23], train_loss: 0.3375, val_loss: 11.9034, val_acc: 0.2792
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [24], train_loss: 0.3076, val_loss: 12.4391, val_acc: 0.2917
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [25], train_loss: 0.2720, val_loss: 13.2115, val_acc: 0.2830
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [26], train_loss: 0.2997, val_loss: 12.7151, val_acc: 0.2854
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [27], train_loss: 0.2445, val_loss: 13.0876, val_acc: 0.2885
  0%|          | 0/334 [00:00<?, ?it/s]
```

```
Epoch [28], train_loss: 0.2383, val_loss: 13.5954, val_acc: 0.2691
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [29], train_loss: 0.2713, val_loss: 13.3136, val_acc: 0.2667
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [30], train_loss: 0.2412, val_loss: 14.6833, val_acc: 0.2653
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [31], train_loss: 0.1954, val_loss: 15.4177, val_acc: 0.2941
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [32], train_loss: 0.3060, val_loss: 14.6058, val_acc: 0.2910
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [33], train_loss: 0.2451, val_loss: 14.6765, val_acc: 0.2566
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [34], train_loss: 0.2165, val_loss: 16.2358, val_acc: 0.2573
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [35], train_loss: 0.1998, val_loss: 16.0646, val_acc: 0.2691
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [36], train_loss: 0.2003, val_loss: 16.1693, val_acc: 0.2597
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [37], train_loss: 0.1904, val_loss: 15.5741, val_acc: 0.2934
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [38], train_loss: 0.1457, val_loss: 17.5492, val_acc: 0.2573
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [39], train_loss: 0.1306, val_loss: 17.3114, val_acc: 0.2660
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [40], train_loss: 0.1556, val_loss: 17.9774, val_acc: 0.2760
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [41], train_loss: 0.2717, val_loss: 15.4870, val_acc: 0.2417
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [42], train_loss: 0.1772, val_loss: 18.4256, val_acc: 0.2760
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [43], train_loss: 0.2047, val_loss: 16.5789, val_acc: 0.2597
  0%|          | 0/334 [00:00<?, ?it/s]
```

```
Epoch [44], train_loss: 0.1943, val_loss: 17.2899, val_acc: 0.2542
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [45], train_loss: 0.1947, val_loss: 18.0521, val_acc: 0.2510
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [46], train_loss: 0.1354, val_loss: 17.7582, val_acc: 0.2635
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [47], train_loss: 0.1517, val_loss: 17.5030, val_acc: 0.3104
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [48], train_loss: 0.1742, val_loss: 17.5753, val_acc: 0.2885
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [49], train_loss: 0.2826, val_loss: 17.6750, val_acc: 0.2785
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [0], train_loss: 4.9231, val_loss: 4.7034, val_acc: 0.0712
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [1], train_loss: 4.5586, val_loss: 4.4102, val_acc: 0.1212
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [2], train_loss: 4.2871, val_loss: 4.2268, val_acc: 0.1698
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [3], train_loss: 4.0394, val_loss: 4.1092, val_acc: 0.2403
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [4], train_loss: 3.8210, val_loss: 3.9268, val_acc: 0.2465
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [5], train_loss: 3.6018, val_loss: 3.7164, val_acc: 0.3083
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [6], train_loss: 3.4449, val_loss: 3.7051, val_acc: 0.3615
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [7], train_loss: 3.3092, val_loss: 3.5231, val_acc: 0.3597
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [8], train_loss: 3.1817, val_loss: 3.4779, val_acc: 0.4076
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [9], train_loss: 3.0645, val_loss: 3.3370, val_acc: 0.4326
  0%|          | 0/334 [00:00<?, ?it/s]
```

```
Epoch [10], train_loss: 2.9605, val_loss: 3.3863, val_acc: 0.4045
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [11], train_loss: 2.8576, val_loss: 3.1452, val_acc: 0.4476
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [12], train_loss: 2.7556, val_loss: 3.1400, val_acc: 0.4764
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [13], train_loss: 2.6703, val_loss: 3.0496, val_acc: 0.4726
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [14], train_loss: 2.5747, val_loss: 3.0365, val_acc: 0.4569
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [15], train_loss: 2.4708, val_loss: 3.2646, val_acc: 0.4545
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [16], train_loss: 2.4151, val_loss: 3.0858, val_acc: 0.4826
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [17], train_loss: 2.3539, val_loss: 3.0687, val_acc: 0.4670
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [18], train_loss: 2.2659, val_loss: 2.9950, val_acc: 0.5045
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [19], train_loss: 2.1914, val_loss: 3.0674, val_acc: 0.4733
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [20], train_loss: 2.1553, val_loss: 3.1107, val_acc: 0.4771
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [21], train_loss: 2.0928, val_loss: 2.9320, val_acc: 0.5351
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [22], train_loss: 2.0091, val_loss: 2.9467, val_acc: 0.5288
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [23], train_loss: 1.9743, val_loss: 2.8457, val_acc: 0.5389
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [24], train_loss: 1.9131, val_loss: 2.8077, val_acc: 0.5563
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [25], train_loss: 1.8790, val_loss: 2.8335, val_acc: 0.5351
  0%|          | 0/334 [00:00<?, ?it/s]
```

```
Epoch [26], train_loss: 1.8285, val_loss: 2.8233, val_acc: 0.5389
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [27], train_loss: 1.7243, val_loss: 2.9385, val_acc: 0.5382
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [28], train_loss: 1.7336, val_loss: 2.8578, val_acc: 0.5569
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [29], train_loss: 1.6545, val_loss: 2.9931, val_acc: 0.5264
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [30], train_loss: 1.6137, val_loss: 2.6508, val_acc: 0.5819
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [31], train_loss: 1.5784, val_loss: 2.6916, val_acc: 0.5976
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [32], train_loss: 1.5171, val_loss: 2.6744, val_acc: 0.6007
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [33], train_loss: 1.4834, val_loss: 2.9805, val_acc: 0.5694
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [34], train_loss: 1.4249, val_loss: 2.9482, val_acc: 0.5663
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [35], train_loss: 1.4058, val_loss: 2.7303, val_acc: 0.5819
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [36], train_loss: 1.3300, val_loss: 2.7255, val_acc: 0.5826
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [37], train_loss: 1.3190, val_loss: 2.7577, val_acc: 0.5788
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [38], train_loss: 1.2642, val_loss: 2.8431, val_acc: 0.5569
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [39], train_loss: 1.2281, val_loss: 2.8646, val_acc: 0.5569
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [40], train_loss: 1.2130, val_loss: 2.8495, val_acc: 0.5858
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [41], train_loss: 1.1494, val_loss: 2.7654, val_acc: 0.5976
  0%|          | 0/334 [00:00<?, ?it/s]
```

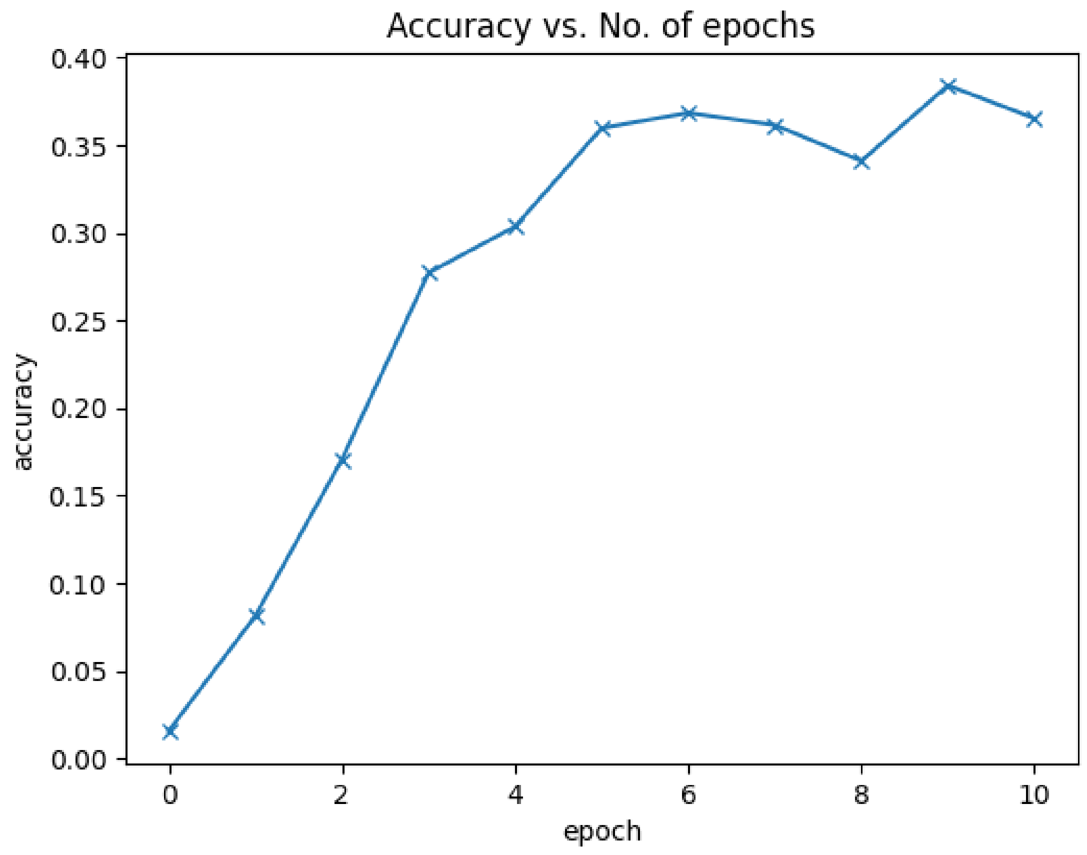
```
Epoch [42], train_loss: 1.1291, val_loss: 2.6940, val_acc: 0.6226
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [43], train_loss: 1.0931, val_loss: 2.8511, val_acc: 0.5944
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [44], train_loss: 1.0664, val_loss: 2.7750, val_acc: 0.6226
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [45], train_loss: 1.0220, val_loss: 2.8061, val_acc: 0.5913
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [46], train_loss: 0.9887, val_loss: 2.7541, val_acc: 0.6108
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [47], train_loss: 0.9792, val_loss: 2.8887, val_acc: 0.5851
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [48], train_loss: 0.9706, val_loss: 2.8030, val_acc: 0.6132
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [49], train_loss: 0.9332, val_loss: 2.9605, val_acc: 0.5920
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [0], train_loss: 4.8640, val_loss: 4.6040, val_acc: 0.0906
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [1], train_loss: 4.5031, val_loss: 4.3287, val_acc: 0.1674
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [2], train_loss: 4.1797, val_loss: 4.0585, val_acc: 0.2410
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [3], train_loss: 3.9109, val_loss: 3.8555, val_acc: 0.2583
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [4], train_loss: 3.6774, val_loss: 3.6797, val_acc: 0.3521
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [5], train_loss: 3.4965, val_loss: 3.5270, val_acc: 0.4170
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [6], train_loss: 3.3338, val_loss: 3.5719, val_acc: 0.3663
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [7], train_loss: 3.1685, val_loss: 3.3173, val_acc: 0.4288
  0%|          | 0/334 [00:00<?, ?it/s]
```

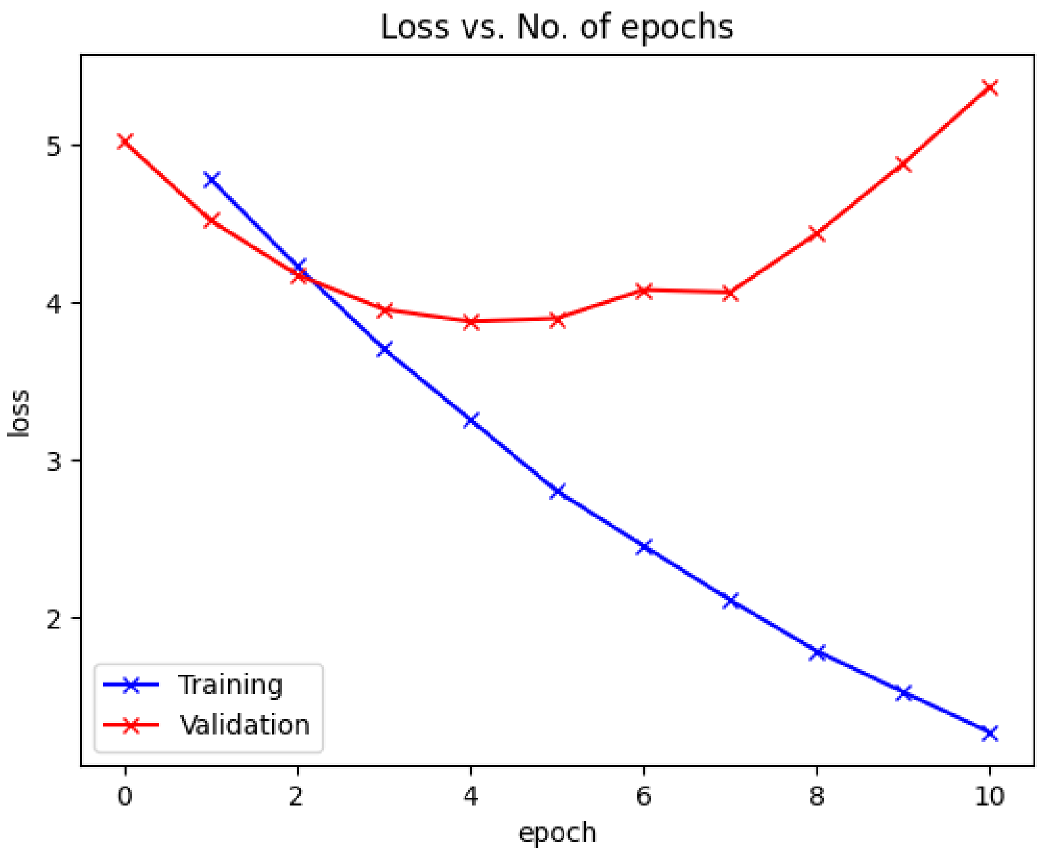
```
Epoch [8], train_loss: 3.0445, val_loss: 3.3981, val_acc: 0.4358
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [9], train_loss: 2.9122, val_loss: 3.8872, val_acc: 0.3458
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [10], train_loss: 2.8055, val_loss: 3.1880, val_acc: 0.4420
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [11], train_loss: 2.7232, val_loss: 3.3467, val_acc: 0.4288
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [12], train_loss: 2.6488, val_loss: 3.3861, val_acc: 0.4208
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [13], train_loss: 2.5505, val_loss: 3.1105, val_acc: 0.4663
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [14], train_loss: 2.4817, val_loss: 3.0594, val_acc: 0.4944
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [15], train_loss: 2.4052, val_loss: 3.0028, val_acc: 0.5132
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [16], train_loss: 2.3351, val_loss: 3.0496, val_acc: 0.4976
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [17], train_loss: 2.2660, val_loss: 3.2916, val_acc: 0.4781
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [18], train_loss: 2.2286, val_loss: 2.9210, val_acc: 0.5382
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [19], train_loss: 2.1439, val_loss: 3.0677, val_acc: 0.5233
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [20], train_loss: 2.1041, val_loss: 2.8755, val_acc: 0.5351
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [21], train_loss: 2.0522, val_loss: 2.9087, val_acc: 0.5076
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [22], train_loss: 1.9796, val_loss: 2.9033, val_acc: 0.5326
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [23], train_loss: 1.9656, val_loss: 3.0076, val_acc: 0.5163
  0%|          | 0/334 [00:00<?, ?it/s]
```

```
Epoch [24], train_loss: 1.9425, val_loss: 3.0451, val_acc: 0.5007
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [25], train_loss: 1.9119, val_loss: 2.9573, val_acc: 0.5170
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [26], train_loss: 1.8218, val_loss: 2.7627, val_acc: 0.5538
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [27], train_loss: 1.7915, val_loss: 2.8448, val_acc: 0.5601
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [28], train_loss: 1.7729, val_loss: 2.8722, val_acc: 0.5663
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [29], train_loss: 1.7190, val_loss: 2.9020, val_acc: 0.5569
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [30], train_loss: 1.7032, val_loss: 2.7985, val_acc: 0.5576
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [31], train_loss: 1.6288, val_loss: 2.8299, val_acc: 0.5538
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [32], train_loss: 1.6053, val_loss: 2.7596, val_acc: 0.5594
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [33], train_loss: 1.5711, val_loss: 2.8595, val_acc: 0.5483
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [34], train_loss: 1.5486, val_loss: 2.7680, val_acc: 0.5750
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [35], train_loss: 1.5177, val_loss: 2.8348, val_acc: 0.5427
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [36], train_loss: 1.4685, val_loss: 2.8126, val_acc: 0.5444
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [37], train_loss: 1.4609, val_loss: 2.6831, val_acc: 0.5844
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [38], train_loss: 1.4412, val_loss: 2.6478, val_acc: 0.5976
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [39], train_loss: 1.3726, val_loss: 2.6826, val_acc: 0.5750
  0%|          | 0/334 [00:00<?, ?it/s]
```

```
Epoch [40], train_loss: 1.3544, val_loss: 2.6511, val_acc: 0.5608
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [41], train_loss: 1.3155, val_loss: 2.9899, val_acc: 0.5351
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [42], train_loss: 1.2683, val_loss: 2.7803, val_acc: 0.5764
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [43], train_loss: 1.2792, val_loss: 2.7069, val_acc: 0.5813
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [44], train_loss: 1.2477, val_loss: 2.6604, val_acc: 0.5781
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [45], train_loss: 1.2199, val_loss: 2.8445, val_acc: 0.5819
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [46], train_loss: 1.1720, val_loss: 2.6801, val_acc: 0.6257
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [47], train_loss: 1.1515, val_loss: 2.6145, val_acc: 0.5913
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [48], train_loss: 1.1455, val_loss: 2.7587, val_acc: 0.5882
  0%|          | 0/334 [00:00<?, ?it/s]
Epoch [49], train_loss: 1.1239, val_loss: 2.8798, val_acc: 0.5701
```

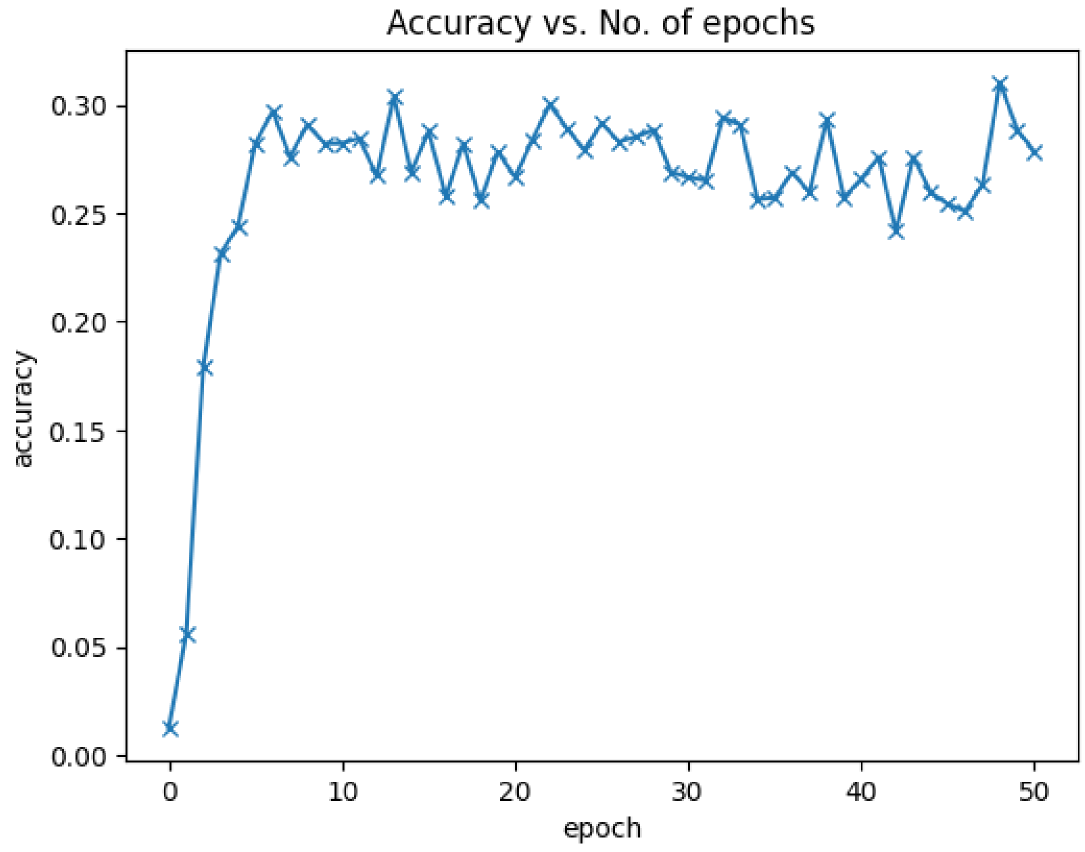
0.2 Baseline accuracy and loss (10 epochs)

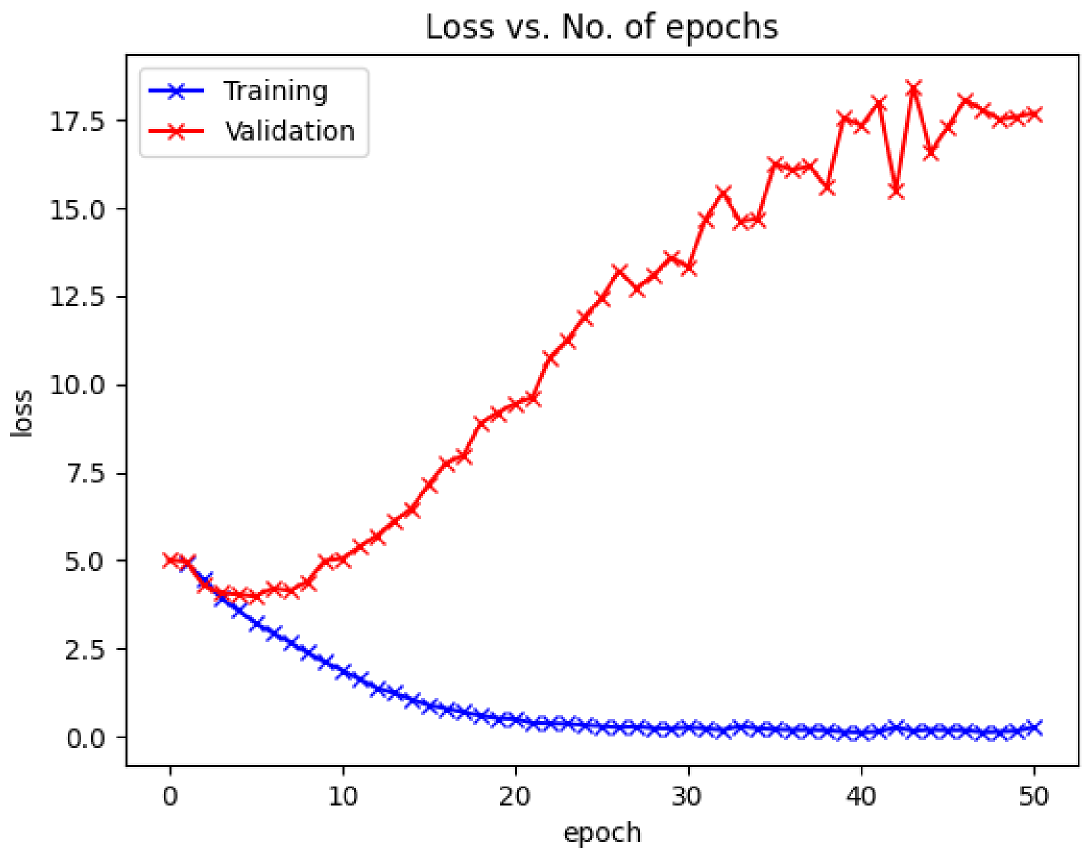




```
 + Number of FLOPs: 0.69G
Evaluation for baseline (10 epochs): {'val_loss': 5.366818904876709, 'val_acc':
0.3652777671813965, 'train_loss': 1.2670550346374512}
```

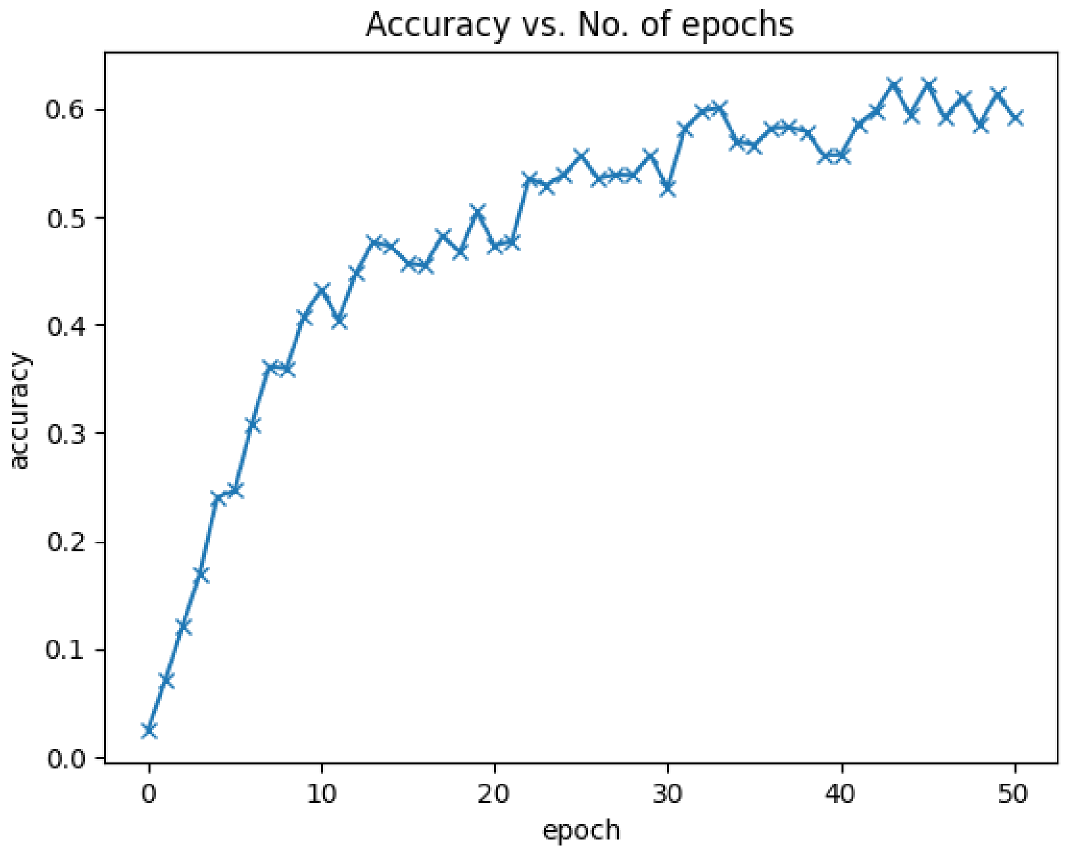
0.3 Baseline accuracy and loss (50 epochs)

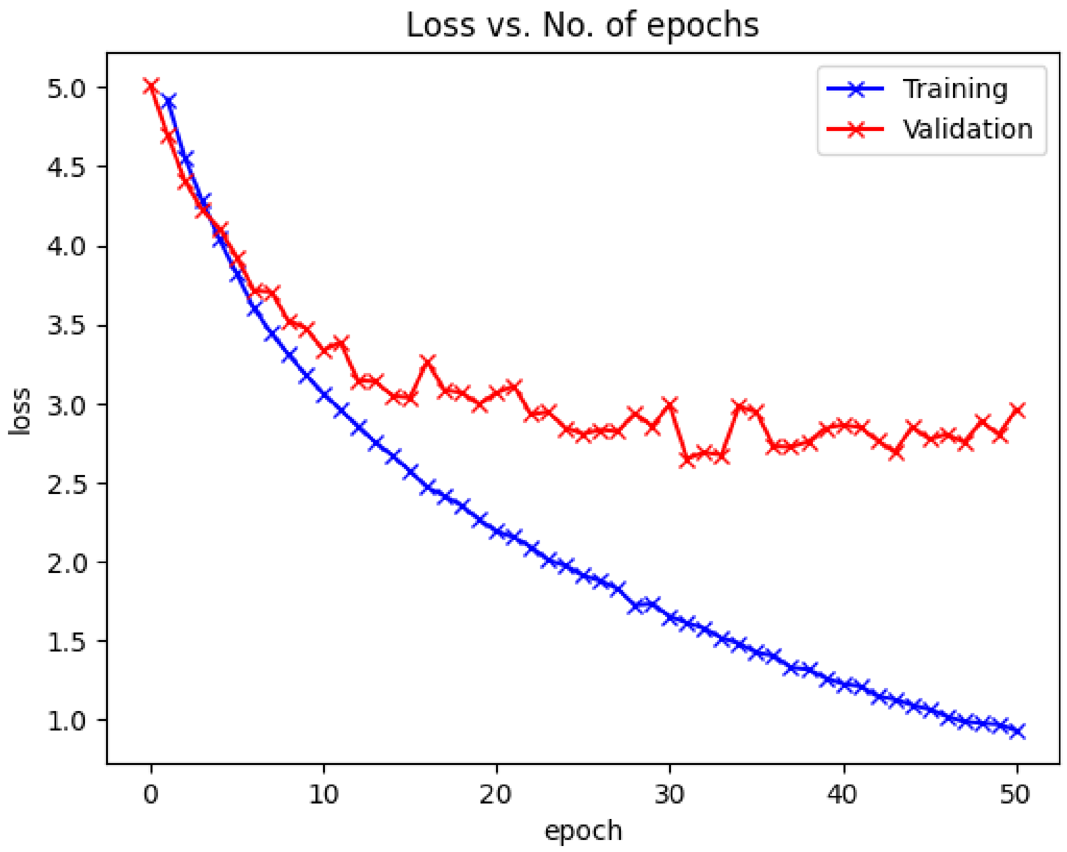




```
 + Number of FLOPs: 0.69G
Starting evaluation for baseline (50 epochs): {'val_loss': 17.674999237060547,
'val_acc': 0.27847224473953247, 'train_loss': 0.28257250785827637}
```

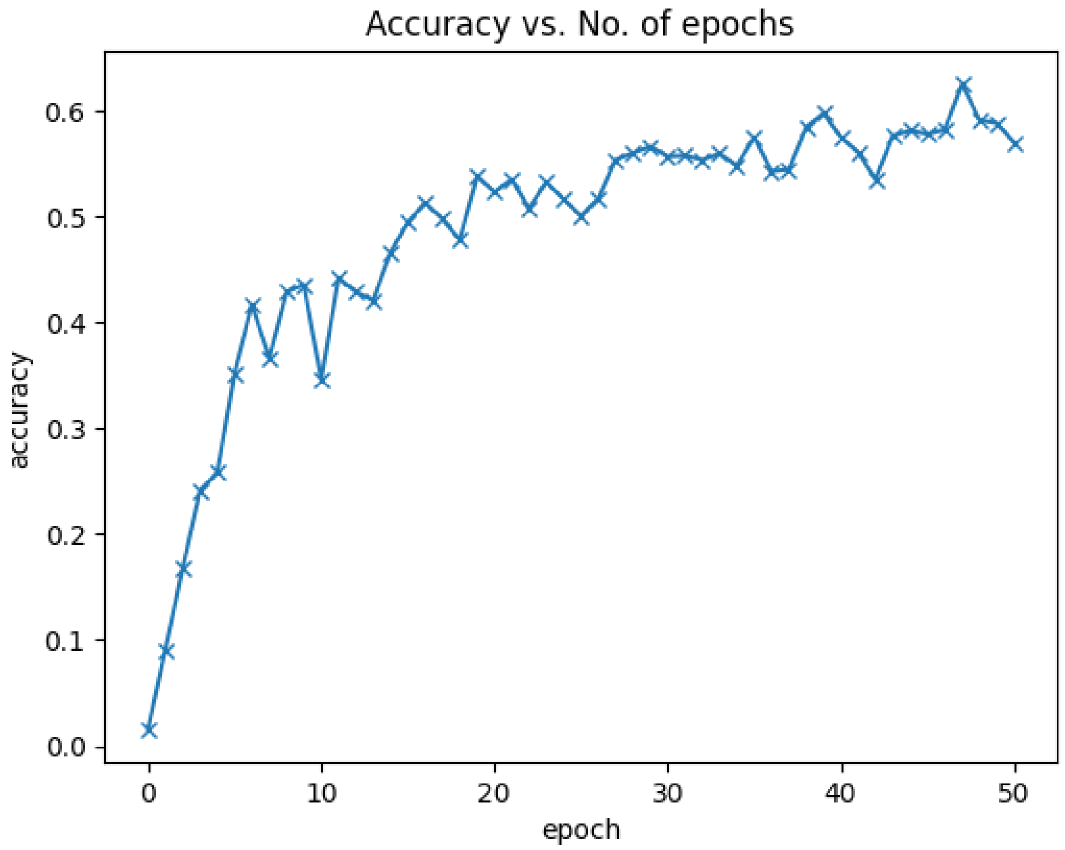
0.4 Dropout model accuracy and loss

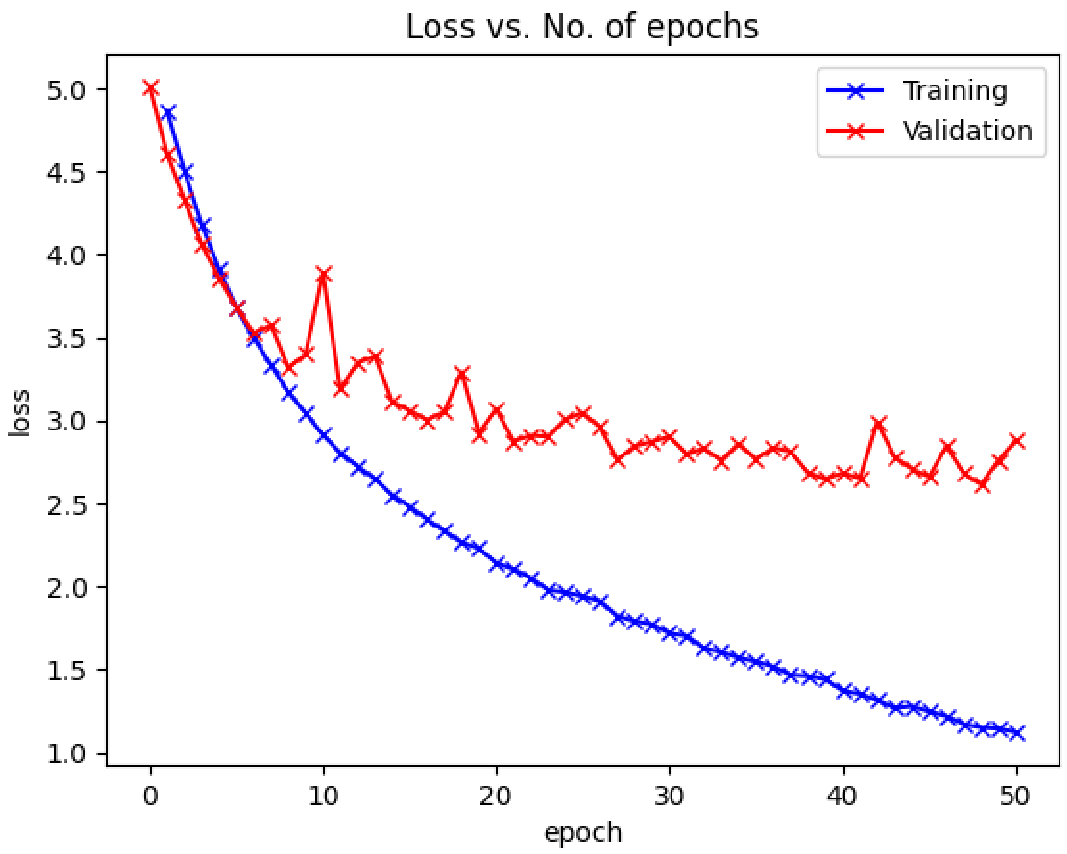




```
 + Number of FLOPs: 0.69G
Starting evaluation for dropout model: {'val_loss': 2.9605469703674316,
'val_acc': 0.5920138955116272, 'train_loss': 0.9332290291786194}
```

0.5 Smaller dropout model evaluation





```
 + Number of FLOPs: 0.53G
Starting evaluation for smaller dropout model: {'val_loss': 2.8797903060913086,
'val_acc': 0.5701388716697693, 'train_loss': 1.123948097229004}
```