Explanations

Helper doc that summarizes the work and contains conclusions.

Data

Python file ‘analyze\_data.py’ based on:

[ElephClass - Asian vs African Elephants Classifier (kaggle.com)](https://www.kaggle.com/code/nasruddinaz/elephclass-asian-vs-african-elephants-classifier)

How to distinguish African and Asian elephants:

[What's the Difference between African and Asian Elephants? | Thomson Safaris](https://thomsonsafaris.com/blog/difference-african-asian-elephant/)

What features may the network learn? Ears, skin, environment, head ‘dome’, …

Data characteristics:

1. Different image shapes – ranges from (100, 100) to (4992, 3328).
2. Train set contains less than 5% mislabeled images.
3. All images in the test set have the correct label.
4. Balanced distribution for the train and test sets.

Binary classification Task

best results achieved on this dataset (from Kaggle) using:

1. Machine Learning –

Extracting features using HoG (object detection method) and concatenate them to RGB values 🡪 reduce dimensions using PCA 🡪 apply SVM 🡪 72% accuracy.

[Image Classification using ML + Hog features(71%) (kaggle.com)](https://www.kaggle.com/code/yassineyahyaouii/image-classification-using-ml-hog-features-71)

2. Deep learning CNN –

Using transfer learning with ImageNet pretrained networks.

The reference: [ElephClass - Asian vs African Elephants Classifier (kaggle.com)](https://www.kaggle.com/code/nasruddinaz/elephclass-asian-vs-african-elephants-classifier) shows best results (91% accuracy) for MobileNet with 20 epochs, batch size of 32 and implementation with tensor flow.

Since the dataset is very small I didn’t separate train set to train and validation (I think that ~80 images for validation are not enough).

I tried different options (using torch) to find the best model:

Batch size = {32, 64, 128}

Epochs = {15, 20, 25}

Update all parameters or just final layer - Feature Extraction (FE) or Fine Tuning (FN)

Models = {MN\_v2, MN\_v3\_small, MN\_v3\_large, squeezenet, resnet50} (MN refers to MobileNet)

The results are summarized in the table below –

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Number | Batch size | Epochs | FT/FE | Pretrained model | Validation accuracy | Time [min.sec] | Use LoRA/DoRA | comments |
| 1 | 32 | 15 | FE | MN\_v2 | 0.51 | 2.16 | No |  |
| 2 | 32 | 15 | FE | MN\_v3\_small | 0.86 | 2.16 | No |  |
| 3 | 64 | 15 | FE | MN\_v3\_small | 0.87 | 2.14 | No |  |
| 4 | 64 | 25 | FE | MN\_v3\_large | 0.84 | 3.50 | No |  |
| 5 | 128 | 15 | FE | MN\_v3\_small | 0.87 | 2.18 | No |  |
| 6 | 128 | 15 | FE | squeezenet | 0.84 | 2.22 | No | Conv 1X1 instead of FC |
| 7 | 128 | 100 | FE | Resnet50 | 0.69 | ? | No | Hadar’s result using WikiArt code |
| 8 | 128 | 20 | FE | Resnet50 | ? | ? | No | Same network as for WikiArt |
| 9 | 128 | 15 | FT | MN\_v3\_small | ? | ? | No | May take some time |
| 10 | 128 | 15 | FE | MN\_v3\_small | ? | ? | LoRA |  |
| 11 | 128 | 15 | FE | MN\_v3\_small | 0.86 | 8.38 (CPU) | DoRA | Best model? |

I don’t have GPU on my computer, so I done it in colab’s GPU but ran out of time so need to complete cases #7-10.

To conclude I don’t achieve tensor flow accuracy but achieve pretty good results with low computations.

The best model was saved and plotted in the notebook.