Università di Genova | DIBRIS DIPARTIMENTO DI INFORMATICA, BIOINGEGNERIA, ROBOTICA E INGEGNERIA DEI SISTEMI

Gina vs. Shiro

A image recognition project about cats

Stefano Amalberti, Lorenzo Mantero, Gina, Shiro

Gina

Shiro

Motivation

Motivation

- We wanted to verify the effectiveness of **transfer learning** on a **very limited dataset**.
- We love **cats**.

Description of the dataset

Description of the dataset

We put together a dataset of about 300 images.

- about 25 were used for testing (8%)
- about 50 were used for validation (16%)

The dataset is far from IID (since all images were taken by us).

Data Agumentation

To enrich the dataset we applied some data augmentation:

- `randomfFlip`
- `ramdomRotation`

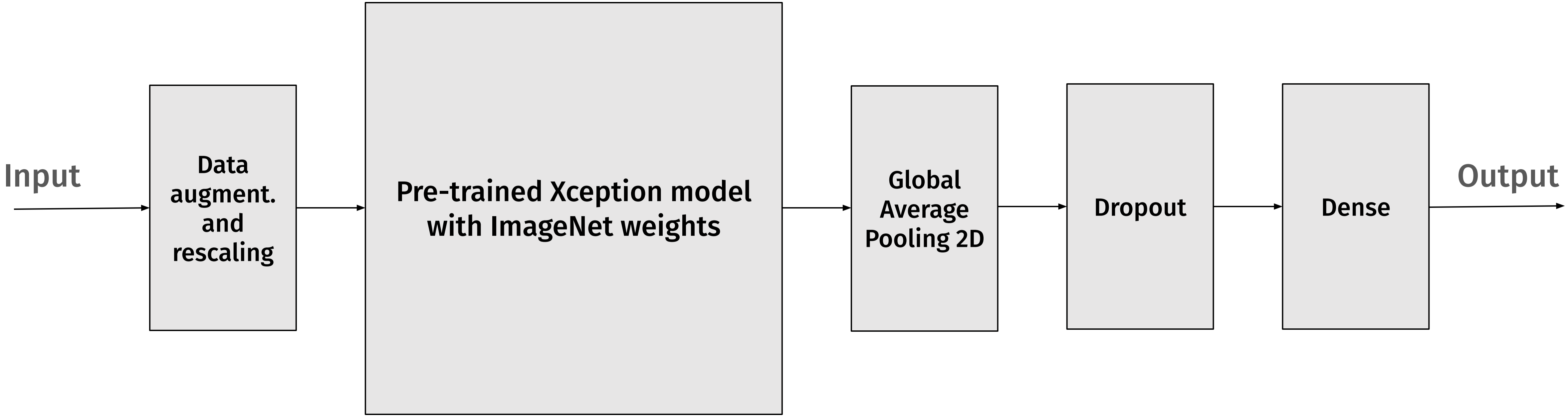
We also tried adding

- `randomSaturation`
- `randomContrast`
- `randomZoom`

and the results were about the same

Model architecture

Model architecture



Model architecture

```python
pretrained_model = keras.applications.Xception(
    weights='imagenet',        # pre traines weights
    input_shape=image_sizes,  # shape of input
    include_top=False         # do not include top fully connected layer
)

# use or not fine tuning
pretrained_model.trainable = apply_fine_tuning

# define input
input = keras.Input(shape=image_sizes)

# apply data agumentation
x = data_augmentation(input)

# normalize and scale imput (from [0 255] to [-1 1])
x = keras.layers.Rescaling(scale=1./(255/2),offset=-1)(x)

# apply the pretrained model
x = pretrained_model(x)

# apply global average pooling to reduce number of parameters
x = keras.layers.GlobalAveragePooling2D()(x)

# apply dropout
x = keras.layers.Dropout(dropout_rate)(x)

# apply a dense layer
output = keras.layers.Dense(1)(x)

# define the model
model = keras.Model(inputs = input, outputs = output)
```

`apply_fine_tuning` is a variable that we can set to `True` or `False`, to evaluate the effectiveness of fine tuning.

Fine Tuning

Fine Tuning:

```
Epoch 1/4
204/204 ━━━━━━━━━━━━━━━━━━━━ 62s 162ms/step - binary_accuracy: 0.6112 - loss: 0.6846 - val_binary_accuracy: 0.6275 - val_loss: 9.9360
Epoch 2/4
204/204 ━━━━━━━━━━━━━━━━━━━━ 19s 66ms/step - binary_accuracy: 0.6330 - loss: 0.6656 - val_binary_accuracy: 0.3922 - val_loss: 1.2087
Epoch 3/4
204/204 ━━━━━━━━━━━━━━━━━━━━ 13s 65ms/step - binary_accuracy: 0.6334 - loss: 0.6408 - val_binary_accuracy: 0.6863 - val_loss: 22.1347
Epoch 4/4
204/204 ━━━━━━━━━━━━━━━━━━━━ 20s 64ms/step - binary_accuracy: 0.6486 - loss: 0.6532 - val_binary_accuracy: 0.5882 - val_loss: 80.5406
<keras.src.callbacks.history.History at 0x7d5f4cd6b490>
```

No Fine Tuning:

```
Epoch 1/4
204/204 ━━━━━━━━━━━━━━━━━━━━ 57s 241ms/step - binary_accuracy: 0.6848 - loss: 0.5456 - val_binary_accuracy: 0.9216 - val_loss: 0.1935
Epoch 2/4
204/204 ━━━━━━━━━━━━━━━━━━━━ 31s 12ms/step - binary_accuracy: 0.9581 - loss: 0.1362 - val_binary_accuracy: 0.9412 - val_loss: 0.1442
Epoch 3/4
204/204 ━━━━━━━━━━━━━━━━━━━━ 2s 11ms/step - binary_accuracy: 0.9902 - loss: 0.0688 - val_binary_accuracy: 0.9608 - val_loss: 0.1196
Epoch 4/4
204/204 ━━━━━━━━━━━━━━━━━━━━ 3s 13ms/step - binary_accuracy: 0.9640 - loss: 0.0781 - val_binary_accuracy: 0.9608 - val_loss: 0.1120
<keras.src.callbacks.history.History at 0x7d5fcc1a7b90>
```

Nota a piè di pagina

Training parameters

Observations

Based on our testing, it appeared that 4 **epochs** were enough to get good results.

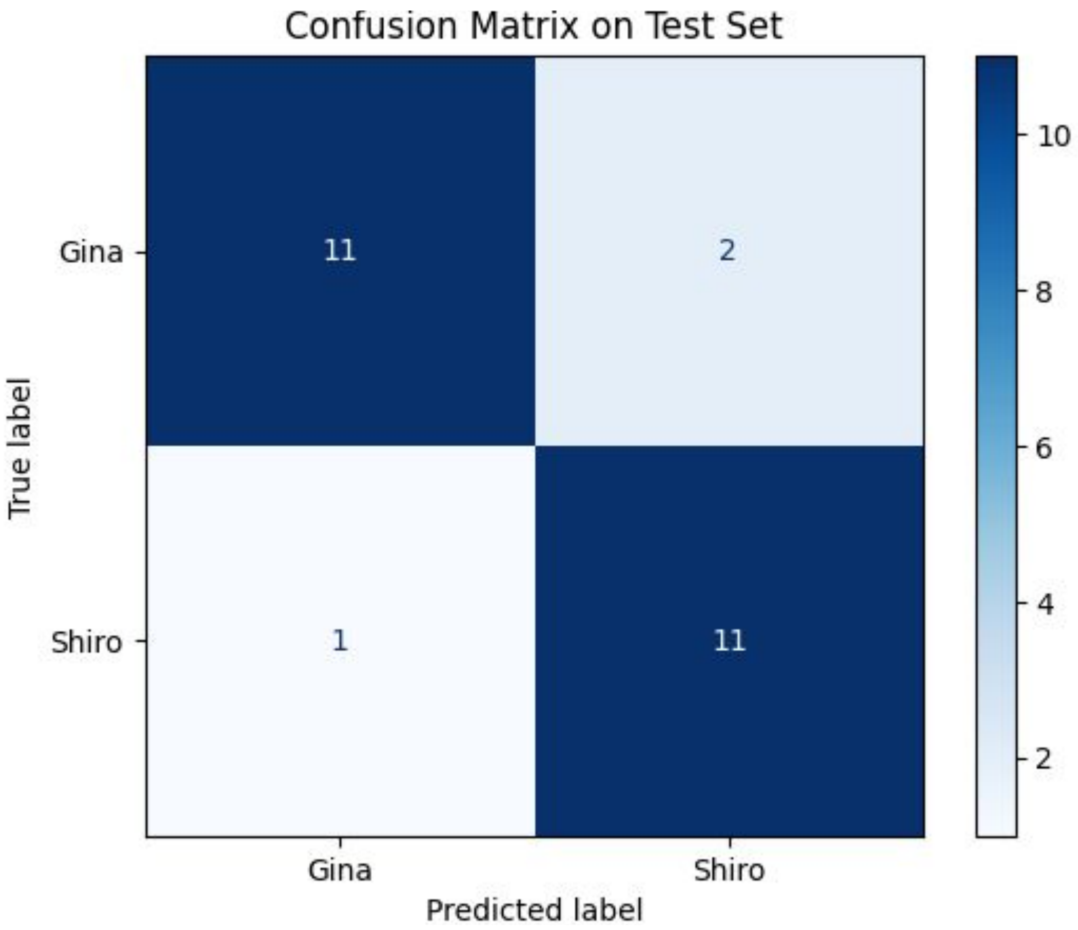
We used the **Adam optimizer**, and tested the following values for the learning rate:

- Learning rate = 10^{-5} gives a **worse** result.
- Learning rate = 10^{-3} is the **default value** and gives the **best** result.
- Learning rate = 10^{-2} causes **overfitting** (more noticeable with increasing epochs).

Nota a piè di pagina

Model test and conclusions

Error on the test



The error on the test ended up being pretty good with only 3 of the 25 samples being misclassified (about 12% error).

To avoid bias, the testing dataset was only used once.

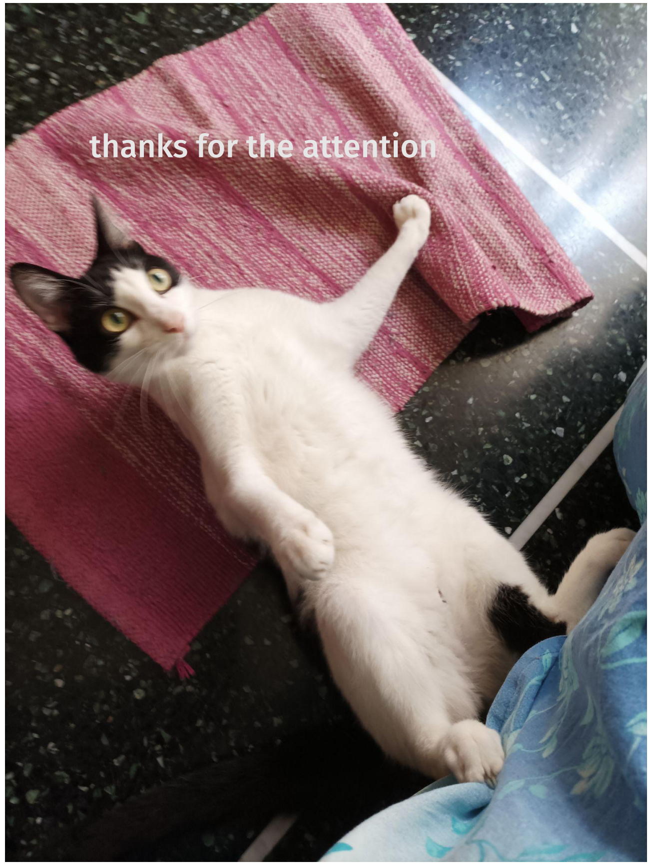
Error on the test

Conclusions

- We had a feeling that the model would classify the images only by **looking at the surroundings** rather than the cats. As it turns out, this is **not really the case**, as on all misclassified images, the model did not recognise the surroundings.

- The model performs **surprisingly well** for the limited size of the dataset, inside the **misclassified images** the cats are **small and harder to identify or even see at all**.

Thanks for the attention



https://github.com/lollomante/ML_Cats - No cats were harmed during the collection of the data.

UniGe

DIBRIS