Media Informatic Systems

Image recognition task

Carlos Sánchez Páez

7^{th} December 2018

Index

1 Description of the task

2 Dataset

3 Preprocessing

4 Structure of the net

5 Neural network development

6 Final results

7 Examples

Index

1 Description of the task
2 Dataset
3 Preprocessing
4 Structure of the net
5 Neural network development
6 Final results
7 Examples

Description of the task

1. Convolutional neural network.
2. Keras framework with TensorFlow backend.
3. Python 3.6.
4. NVIDIA 960M GPU (1 505 GFLOPS).

Index

1 Description of the task

2 Dataset

3 Preprocessing

4 Structure of the net

5 Neural network development

6 Final results

7 Examples

CIFAR100

1. 60 000 32x32 colour images (RGB).

CIFAR100

1. 60 000 32x32 colour images (RGB).
2. 100 classes.

CIFAR100

1. 60 000 32x32 colour images (RGB).
2. 100 classes.
3. 600 images per class.

CIFAR100

1. 60 000 32x32 colour images (RGB).
2. 100 classes.
3. 600 images per class.
    1. 500 training.

CIFAR100

1. 60 000 32x32 colour images (RGB).
2. 100 classes.
3. 600 images per class.
    1. 500 training.
    2. 100 testing.

CIFAR100

1. 60 000 32x32 colour images (RGB).
2. 100 classes.
3. 600 images per class.
    1. 500 training.
    2. 100 testing.
4. Best accuracy: 75.72%

Classes in the dataset

aquatic mammals	beaver, dolphin, otter, seal, whale
fish	aquarium fish, flatfish, ray, shark, trout
flowers	orchids, poppies, roses, sunflowers, tulips
food containers	bottles, bowls, cans, cups, plates
fruit and vegetables	apples, mushrooms, oranges, pears, sweet peppers
household electrical devices	clock, computer keyboard, lamp, telephone, television
household furniture	bed, chair, couch, table, wardrobe
insects	bee, beetle, butterfly, caterpillar, cockroach
large carnivores	bear, leopard, lion, tiger, wolf
large man-made outdoor things	bridge, castle, house, road, skyscraper
large natural outdoor scenes	cloud, forest, mountain, plain, sea
large omnivores and herbivores	camel, cattle, chimpanzee, elephant, kangaroo
medium-sized mammals	fox, porcupine, possum, raccoon, skunk
non-insect invertebrates	crab, lobster, snail, spider, worm
people	baby, boy, girl, man, woman
reptiles	crocodile, dinosaur, lizard, snake, turtle
small mammals	hamster, mouse, rabbit, shrew, squirrel
trees	maple, oak, palm, pine, willow
vehicles 1	bicycle, bus, motorcycle, pickup truck, train
vehicles 2	lawn-mower, rocket, streetcar, tank, tractor

Index

1 Description of the task

2 Dataset

3 Preprocessing

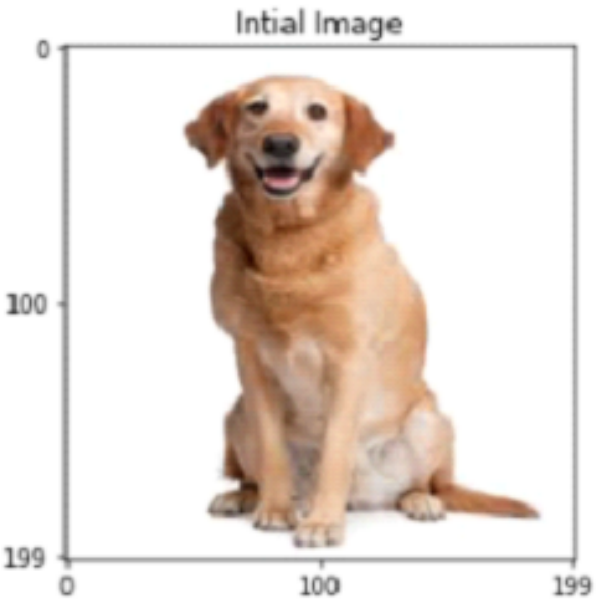
4 Structure of the net

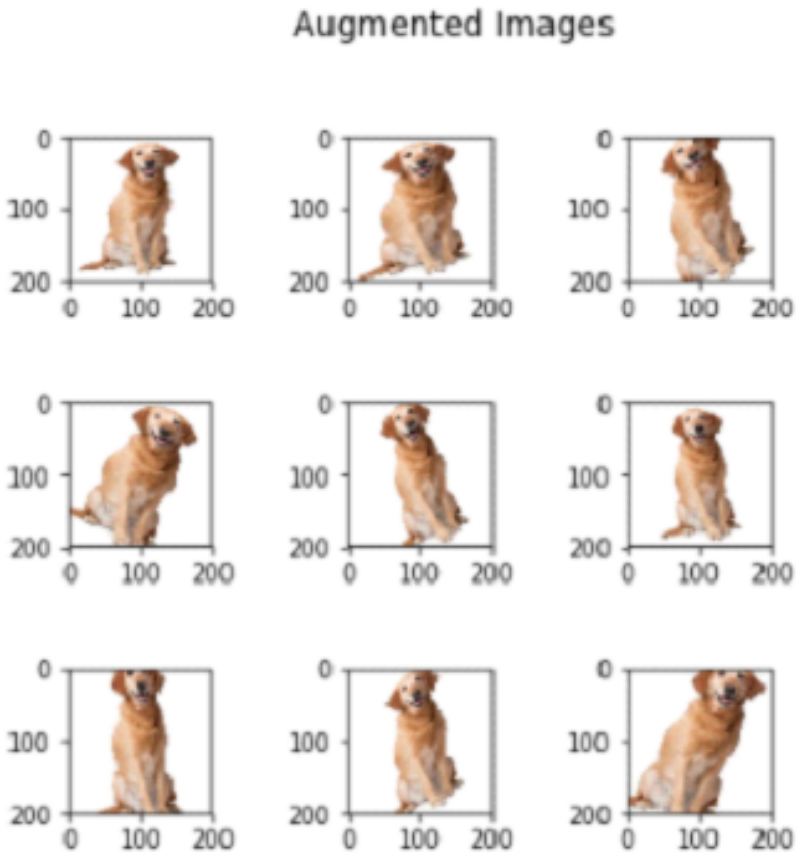
5 Neural network development

6 Final results

7 Examples

Preprocessing (I)





Preprocessing (II)


```
datagen = ImageDataGenerator(
    featurewise_center=False,
    samplewise_center=False,
    featurewise_std_normalization=False,
    samplewise_std_normalization=False,
    zca_whitening=False,
    rotation_range=0,
    width_shift_range=0.1,
    height_shift_range=0.1,
    horizontal_flip=True,
    vertical_flip=True)
```

Index

1 Description of the task

2 Dataset

3 Preprocessing

4 Structure of the net

5 Neural network development

6 Final results

7 Examples

Structure of the net (I)

1. Convolutional layer (128 filters)

Structure of the net (I)

1. Convolutional layer (128 filters)
2. Convolutional layer (128 filters)

Structure of the net (I)

1. Convolutional layer (128 filters)
2. Convolutional layer (128 filters)
3. Max Pooling layer (2x2)

Structure of the net (I)

1. Convolutional layer (128 filters)
2. Convolutional layer (128 filters)
3. Max Pooling layer (2x2)
4. Dropout layer

Structure of the net (I)

1. Convolutional layer (128 filters)
2. Convolutional layer (128 filters)
3. Max Pooling layer (2x2)
4. Dropout layer
5. Convolutional layer (256 filters)

Structure of the net (I)

1. Convolutional layer (128 filters)
2. Convolutional layer (128 filters)
3. Max Pooling layer (2x2)
4. Dropout layer
5. Convolutional layer (256 filters)
6. Convolutional layer (256 filters)

Structure of the net (I)

1. Convolutional layer (128 filters)
2. Convolutional layer (128 filters)
3. Max Pooling layer (2x2)
4. Dropout layer
5. Convolutional layer (256 filters)
6. Convolutional layer (256 filters)
7. Max Pooling layer (2x2)

Structure of the net (I)

1. Convolutional layer (128 filters)
2. Convolutional layer (128 filters)
3. Max Pooling layer (2x2)
4. Dropout layer
5. Convolutional layer (256 filters)
6. Convolutional layer (256 filters)
7. Max Pooling layer (2x2)
8. Dropout layer

Structure of the net (I)

1. Convolutional layer (128 filters)
2. Convolutional layer (128 filters)
3. Max Pooling layer (2x2)
4. Dropout layer
5. Convolutional layer (256 filters)
6. Convolutional layer (256 filters)
7. Max Pooling layer (2x2)
8. Dropout layer
9. Convolutional layer (512 filters)

Structure of the net (I)

1. Convolutional layer (128 filters)
2. Convolutional layer (128 filters)
3. Max Pooling layer (2x2)
4. Dropout layer
5. Convolutional layer (256 filters)
6. Convolutional layer (256 filters)
7. Max Pooling layer (2x2)
8. Dropout layer
9. Convolutional layer (512 filters)
10. Convolutional layer (512 filters)

Structure of the net (I)

1. Convolutional layer (128 filters)
2. Convolutional layer (128 filters)
3. Max Pooling layer (2x2)
4. Dropout layer
5. Convolutional layer (256 filters)
6. Convolutional layer (256 filters)
7. Max Pooling layer (2x2)
8. Dropout layer
9. Convolutional layer (512 filters)
10. Convolutional layer (512 filters)
11. Max Pooling layer (2x2)

Structure of the net (I)

1. Convolutional layer (128 filters)
2. Convolutional layer (128 filters)
3. Max Pooling layer (2x2)
4. Dropout layer
5. Convolutional layer (256 filters)
6. Convolutional layer (256 filters)
7. Max Pooling layer (2x2)
8. Dropout layer
9. Convolutional layer (512 filters)
10. Convolutional layer (512 filters)
11. Max Pooling layer (2x2)
12. Dropout layer

Structure of the net (I)

1. Convolutional layer (128 filters)
2. Convolutional layer (128 filters)
3. Max Pooling layer (2x2)
4. Dropout layer
5. Convolutional layer (256 filters)
6. Convolutional layer (256 filters)
7. Max Pooling layer (2x2)
8. Dropout layer
9. Convolutional layer (512 filters)
10. Convolutional layer (512 filters)
11. Max Pooling layer (2x2)
12. Dropout layer
13. Flatten layer

Structure of the net (I)

1. Convolutional layer (128 filters)
2. Convolutional layer (128 filters)
3. Max Pooling layer (2x2)
4. Dropout layer
5. Convolutional layer (256 filters)
6. Convolutional layer (256 filters)
7. Max Pooling layer (2x2)
8. Dropout layer
9. Convolutional layer (512 filters)
10. Convolutional layer (512 filters)
11. Max Pooling layer (2x2)
12. Dropout layer
13. Flatten layer
14. Fully connected layer (1024 neurons)

Structure of the net (I)

1. Convolutional layer (128 filters)
2. Convolutional layer (128 filters)
3. Max Pooling layer (2x2)
4. Dropout layer
5. Convolutional layer (256 filters)
6. Convolutional layer (256 filters)
7. Max Pooling layer (2x2)
8. Dropout layer
9. Convolutional layer (512 filters)
10. Convolutional layer (512 filters)
11. Max Pooling layer (2x2)
12. Dropout layer
13. Flatten layer
14. Fully connected layer (1024 neurons)
15. Dropout layer

Structure of the net (I)

1. Convolutional layer (128 filters)
2. Convolutional layer (128 filters)
3. Max Pooling layer (2x2)
4. Dropout layer
5. Convolutional layer (256 filters)
6. Convolutional layer (256 filters)
7. Max Pooling layer (2x2)
8. Dropout layer
9. Convolutional layer (512 filters)
10. Convolutional layer (512 filters)
11. Max Pooling layer (2x2)
12. Dropout layer
13. Flatten layer
14. Fully connected layer (1024 neurons)
15. Dropout layer
16. Output layer (10 neurons)

Structure of the net (II)


```
model = Sequential()

model.add(Conv2D(128, (3, 3), padding='same',
                 input_shape=x_train.shape[1:], activation='elu'))
model.add(Conv2D(128, (3, 3), activation='elu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(0.25))

model.add(Conv2D(256, (3, 3), padding='same', activation='elu'))
model.add(Conv2D(256, (3, 3), activation='elu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(0.25))

model.add(Conv2D(512, (3, 3), padding='same', activation='elu'))
model.add(Conv2D(512, (3, 3), activation='elu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(0.25))

model.add(Flatten())
model.add(Dense(1024, activation='elu'))
model.add(Dropout(0.5))
model.add(Dense(parameters.NUM_CLASSES, activation='softmax'))
```

Index

Description of the task
Dataset
Preprocessing
Structure of the net

5 Neural network development

Final results
Examples

Neural network development

1. Calibration of parameters.

Neural network development

1. Calibration of parameters.
    1. Learning rate.

Neural network development

1. Calibration of parameters.
    1. Learning rate.
    2. Number of epochs.

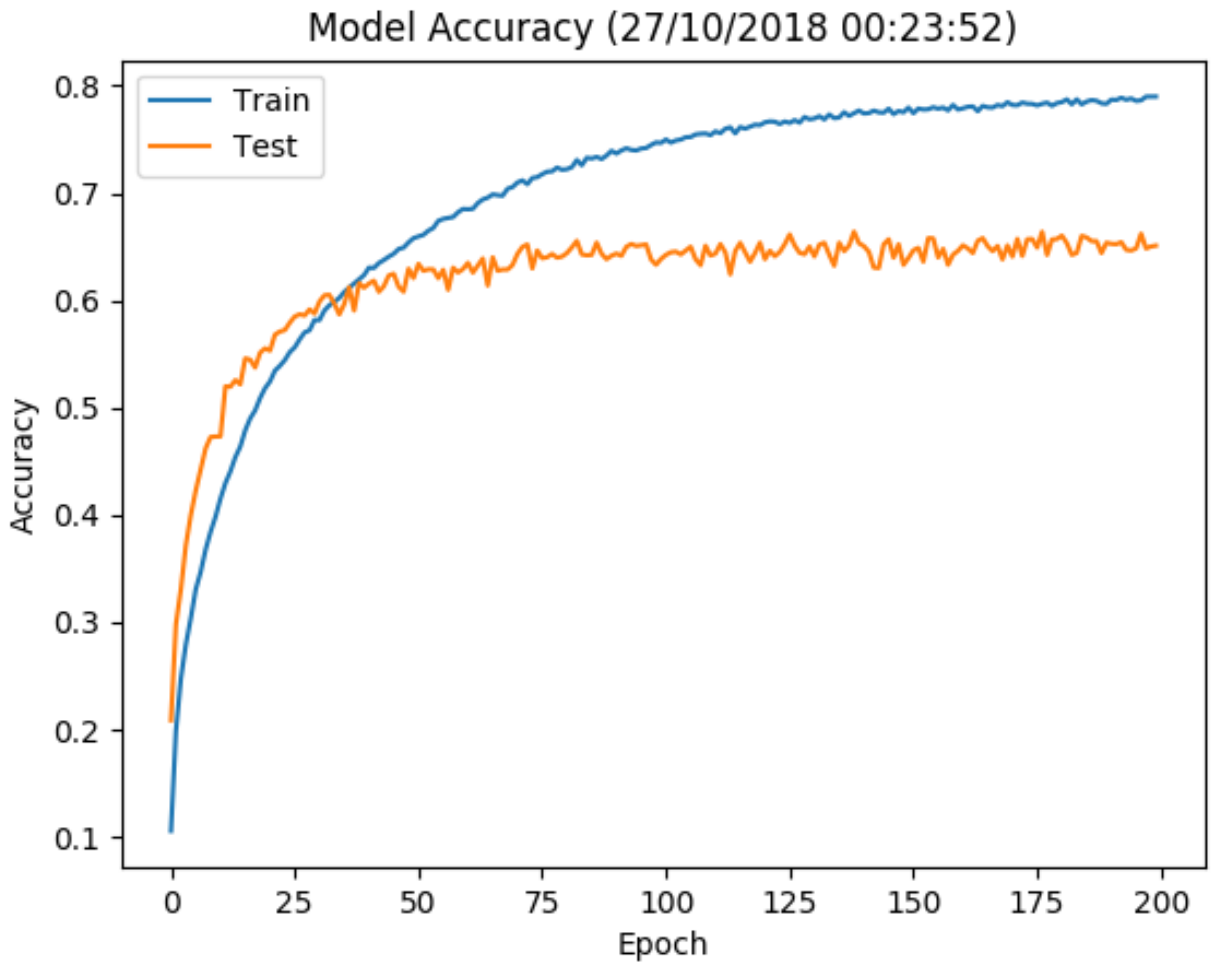
Neural network development

1. Calibration of parameters.
    1. Learning rate.
    2. Number of epochs.
    3. Batch size.

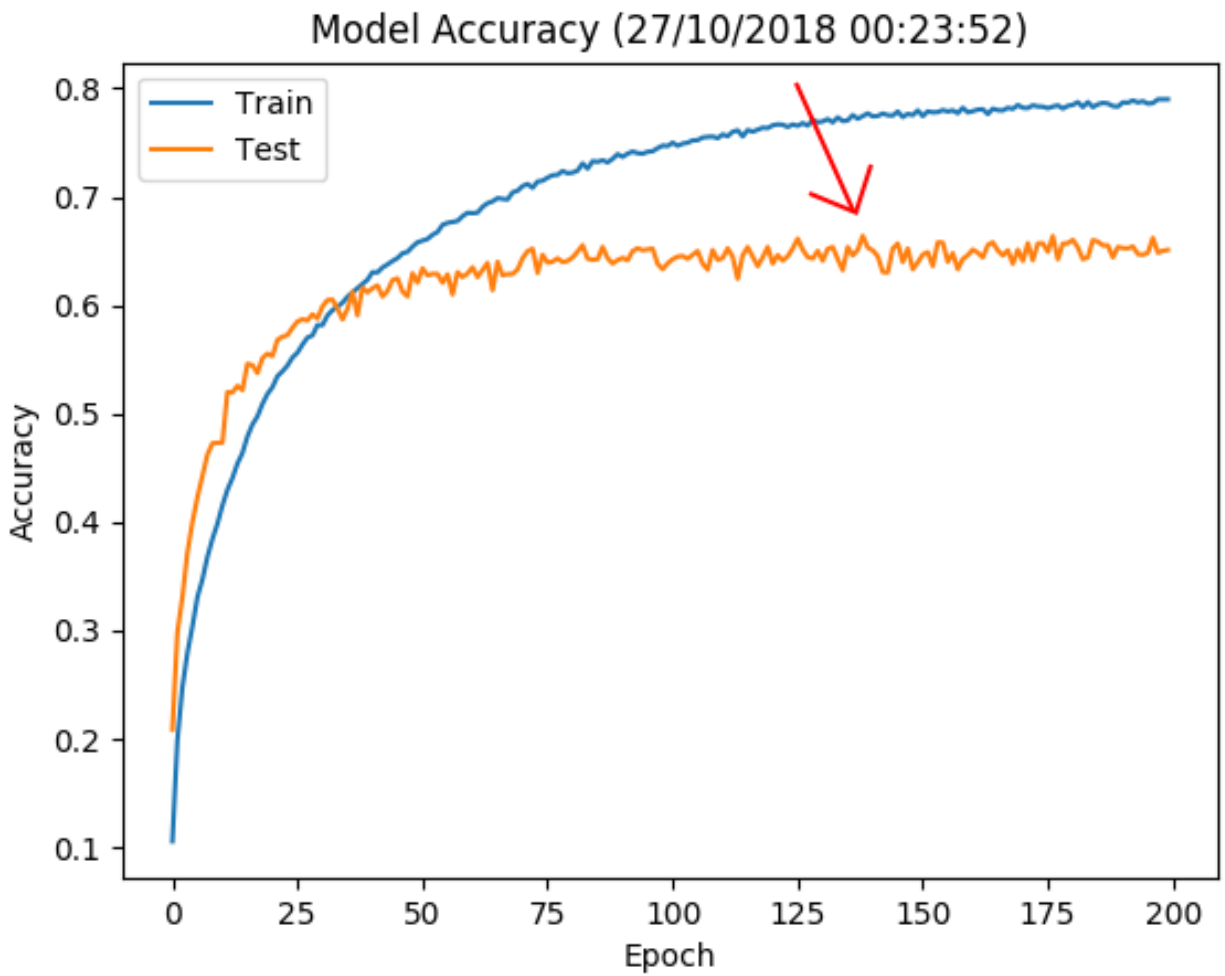
Neural network development

1. Calibration of parameters.
   1. Learning rate.
   2. Number of epochs.
   3. Batch size.
2. Find the maximum global accuracy.

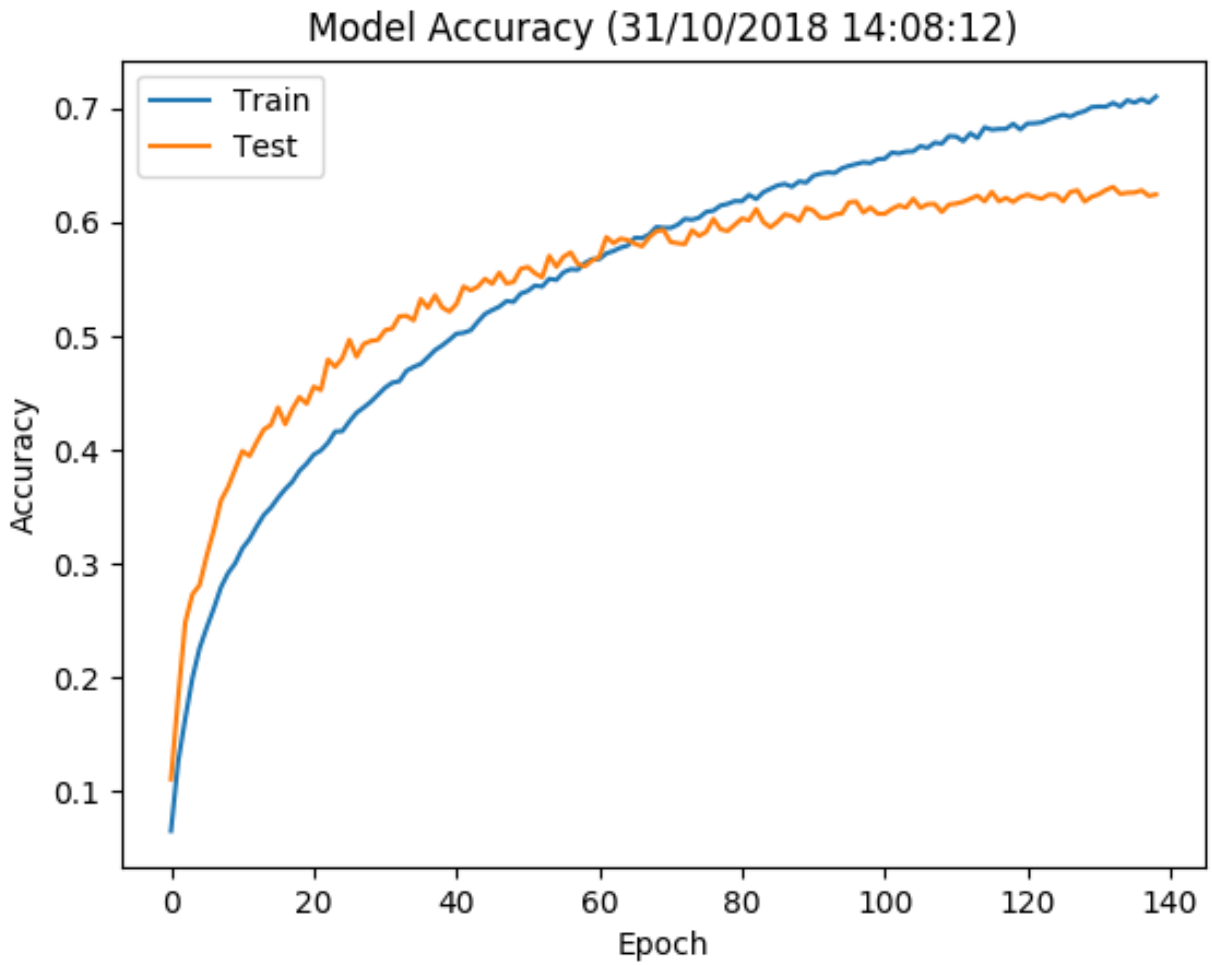
200 epochs training



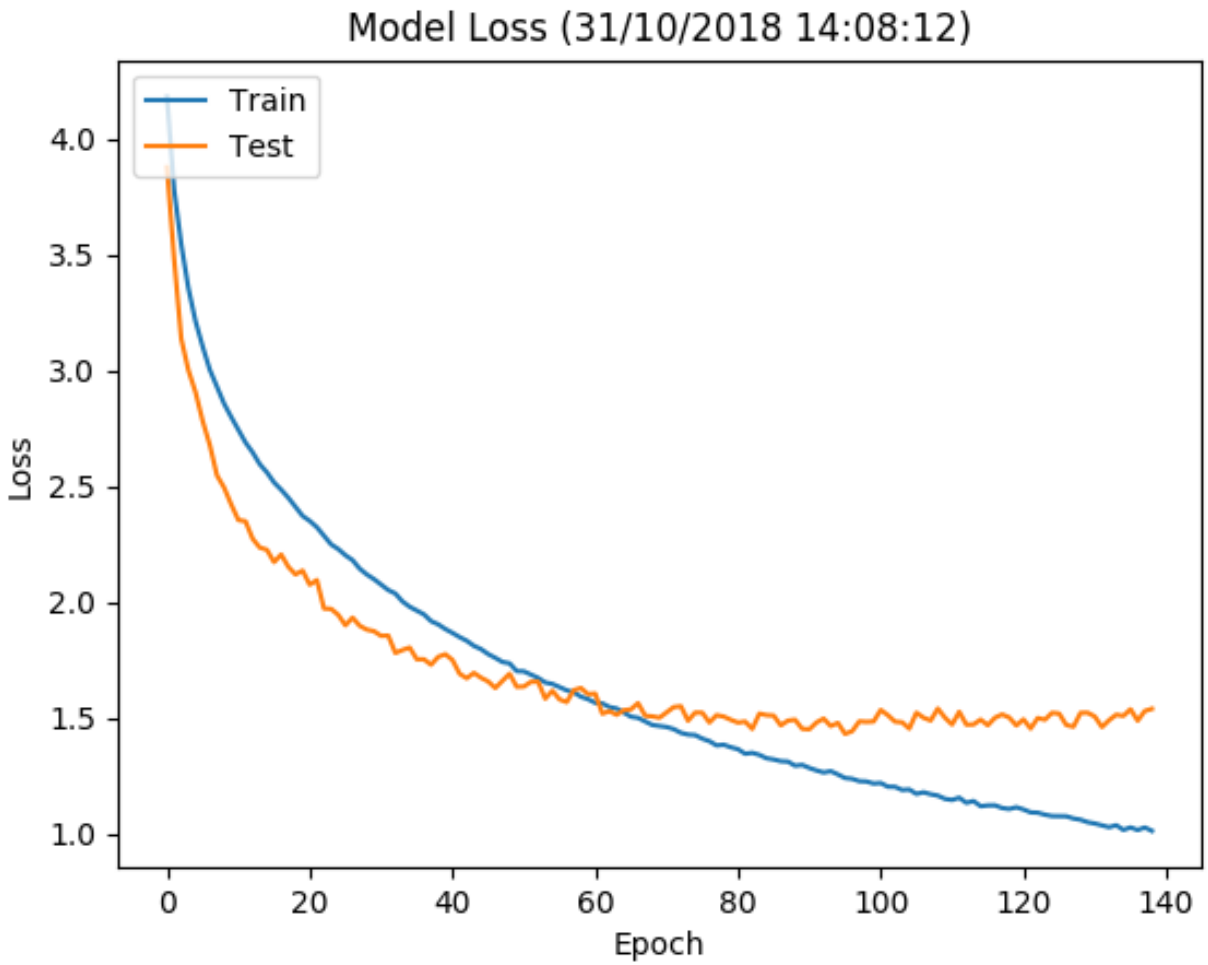
200 epochs training



139 epochs training



139 epochs training



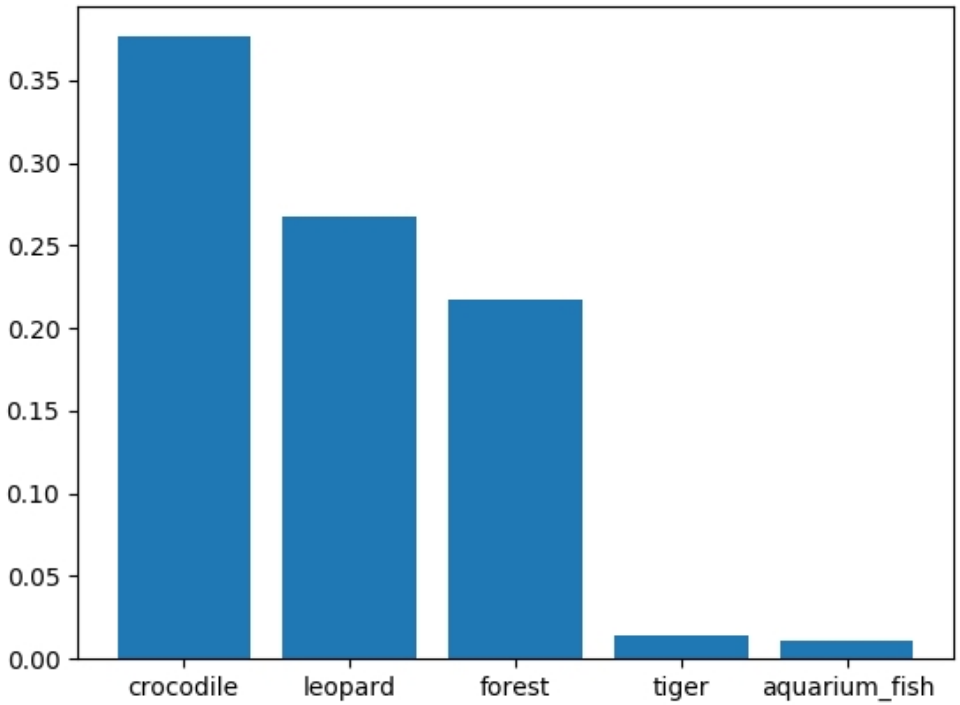
Index

139 epochs training

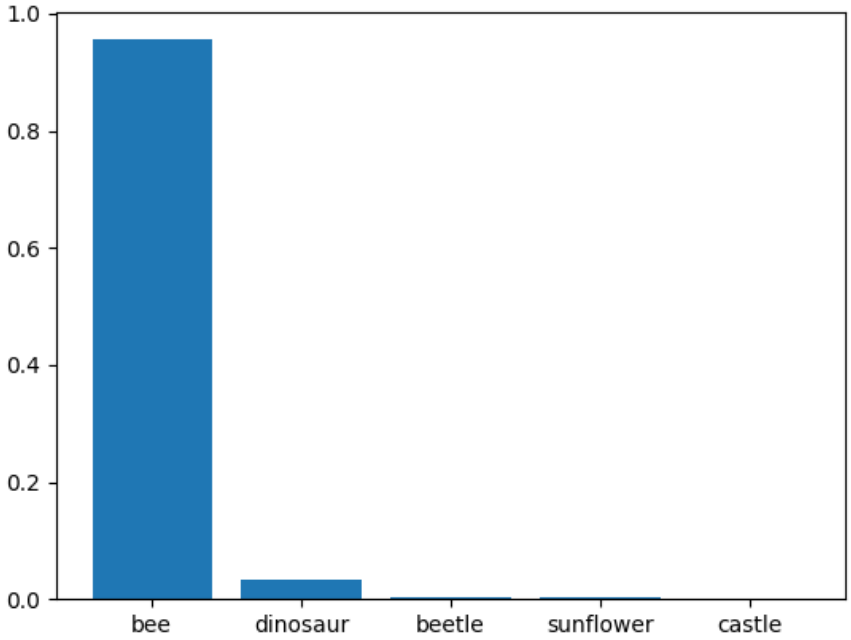
1. Validation accuracy: 0.6187 %
2. Validation loss: 1.5597 %

Index

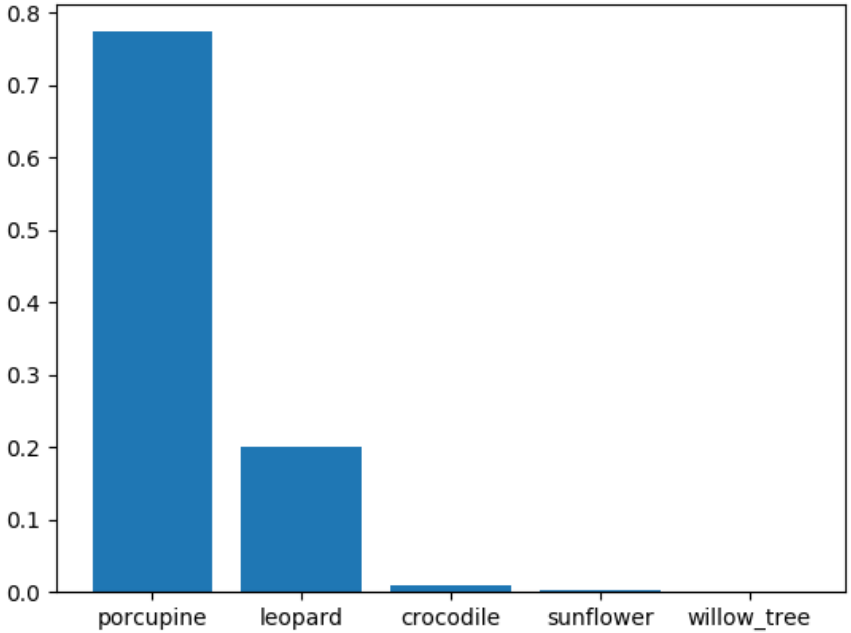
Crocodile prediction



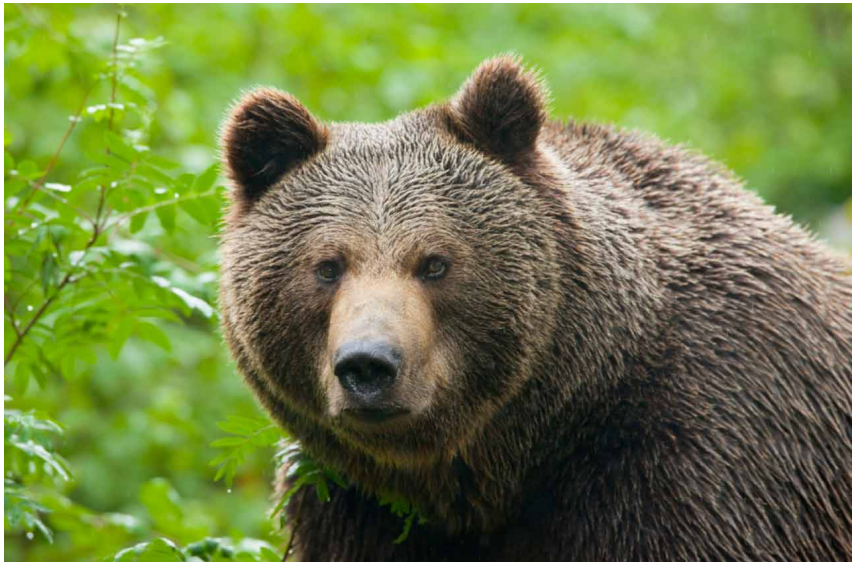
Bee prediction

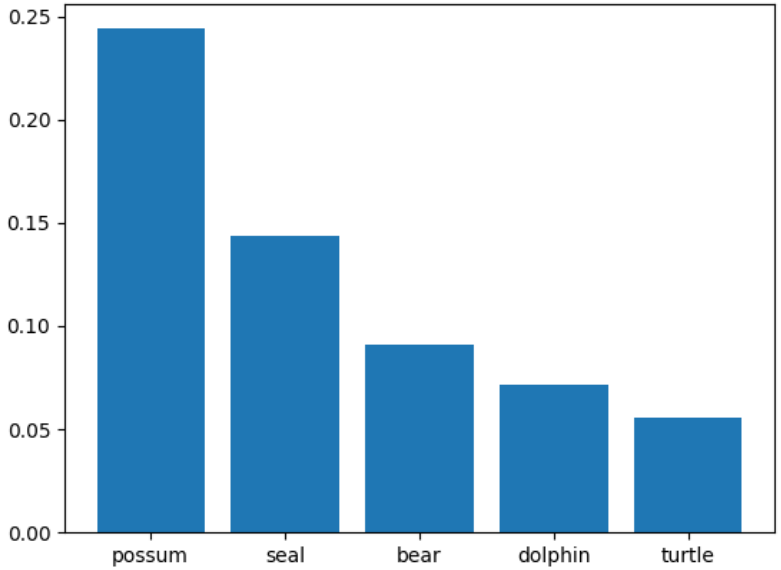


Porcupine prediction



Bear prediction





Thanks for your attention!

Source code available at http://www.github.com/csp98