**CNN model training**

In [7]:

*# Importing req. lib.*

from tensorflow.keras.models import Sequential

from tensorflow.keras.layers import Convolution2D, MaxPooling2D, Flatten, Dense

In [8]:

*# Build a CNN block*

model = Sequential() *# Initializing sequential model*

model.add(Convolution2D(32,(3,3),activation='relu',input\_shape=(64,64,3))) *# convolution layer*

model.add(MaxPooling2D(pool\_size=(2, 2))) *# Max pooling layer*

model.add(Flatten()) *# Flatten layer*

model.add(Dense(300,activation='relu')) *# Hidden layer 1*

model.add(Dense(150,activation='relu')) *# Hidden layer 2*

model.add(Dense(4,activation='softmax')) *# Output layer*

In [10]:

*# Compiling the model*

model.compile(optimizer='adam',loss='categorical\_crossentropy',metrics=['accuracy'])

In [11]:

*# Train model*

model.fit\_generator(xtrain,

steps\_per\_epoch=len(xtrain),

epochs=10,

validation\_data=xtest,

validation\_steps=len(xtest))

/usr/local/lib/python3.7/dist-packages/ipykernel\_launcher.py:7: UserWarning: `Model.fit\_generator` is deprecated and will be removed in a future version. Please use `Model.fit`, which supports generators.

import sys

Epoch 1/10

13/13 [==============================] - 18s 490ms/step - loss: 1.8822 - accuracy: 0.2908 - val\_loss: 1.2656 - val\_accuracy: 0.3558

Epoch 2/10

13/13 [==============================] - 6s 493ms/step - loss: 1.2039 - accuracy: 0.4661 - val\_loss: 1.0490 - val\_accuracy: 0.6319

Epoch 3/10

13/13 [==============================] - 6s 473ms/step - loss: 1.0134 - accuracy: 0.5969 - val\_loss: 0.9052 - val\_accuracy: 0.6135

Epoch 4/10

13/13 [==============================] - 6s 471ms/step - loss: 0.8838 - accuracy: 0.6470 - val\_loss: 0.7871 - val\_accuracy: 0.6442

Epoch 5/10

13/13 [==============================] - 6s 462ms/step - loss: 0.7732 - accuracy: 0.6963 - val\_loss: 0.6946 - val\_accuracy: 0.7423

Epoch 6/10

13/13 [==============================] - 6s 467ms/step - loss: 0.6976 - accuracy: 0.7229 - val\_loss: 0.6000 - val\_accuracy: 0.7669

Epoch 7/10

13/13 [==============================] - 6s 470ms/step - loss: 0.6408 - accuracy: 0.7585 - val\_loss: 0.4942 - val\_accuracy: 0.8405

Epoch 8/10

13/13 [==============================] - 6s 473ms/step - loss: 0.5708 - accuracy: 0.7779 - val\_loss: 0.5511 - val\_accuracy: 0.7945

Epoch 9/10

13/13 [==============================] - 6s 466ms/step - loss: 0.5394 - accuracy: 0.7981 - val\_loss: 0.5997 - val\_accuracy: 0.8006

Epoch 10/10

13/13 [==============================] - 6s 459ms/step - loss: 0.5215 - accuracy: 0.7948 - val\_loss: 0.4270 - val\_accuracy: 0.8620

Out[11]:

<keras.callbacks.History at 0x7f57a224e290>

In [12]:

*# Save model*

model.save('animal.h5')

**Testing model**

In [17]:

from tensorflow.keras.preprocessing import image

import numpy as np

In [24]:

*# Testing 1*

img = image.load\_img('/content/dataset/Testing/elephants/photo\_1552055570\_5c41ef975579.jpeg',target\_size=(64,64)) *# Reading image*

x = image.img\_to\_array(img) *# Converting image into array*

x = np.expand\_dims(x,axis=0) *# expanding Dimensions*

pred = np.argmax(model.predict(x)) *# Predicting the higher probablity index*

op = ['bears','crows','elephants','rats'] *# Creating list*

op[pred] *# List indexing with output*

Out[24]:

'elephants'

In [25]:

*# Testing 2*

img = image.load\_img('/content/dataset/Testing/bears/m10.jpeg',target\_size=(64,64)) *# Reading image*

x = image.img\_to\_array(img) *# Converting image into array*

x = np.expand\_dims(x,axis=0) *# expanding Dimensions*

pred = np.argmax(model.predict(x)) *# Predicting the higher probablity index*

op = ['bears','crows','elephants','rats'] *# Creating list*

op[pred] *# List indexing with output*

Out[25]:

'bears'

In [26]:

*# Testing 3*

img = image.load\_img('/content/dataset/Testing/crows/Z1 (65).jpg',target\_size=(64,64)) *# Reading image*

x = image.img\_to\_array(img) *# Converting image into array*

x = np.expand\_dims(x,axis=0) *# expanding Dimensions*

pred = np.argmax(model.predict(x)) *# Predicting the higher probablity index*

op = ['bears','crows','elephants','rats'] *# Creating list*

op[pred] *# List indexing with output*

Out[26]:

'crows'

In [28]:

*# Testing 4*

img = image.load\_img('/content/dataset/Testing/rats/images (55).jpeg',target\_size=(64,64)) *# Reading image*

x = image.img\_to\_array(img) *# Converting image into array*

x = np.expand\_dims(x,axis=0) *# expanding Dimensions*

pred = np.argmax(model.predict(x)) *# Predicting the higher probablity index*

op = ['bears','crows','elephants','rats'] *# Creating list*

op[pred] *# List indexing with output*

Out[28]:

'bears'

In [22]:

xtrain.class\_indices

Out[22]:

{'bears': 0, 'crows': 1, 'elephants': 2, 'rats': 3}

**Model tuning**

In [29]:

from tensorflow.keras.callbacks import EarlyStopping, ReduceLROnPlateau

In [31]:

early\_stop = EarlyStopping(monitor='val\_accuracy',

patience=5)

lr = ReduceLROnPlateau(monitor='val\_accuaracy',

factor=0.5,

min\_lr=0.00001)

callback = [early\_stop,lr]

In [33]:

*# Train model*

model.fit\_generator(xtrain,

steps\_per\_epoch=len(xtrain),

epochs=100,

callbacks=callback,

validation\_data=xtest,

validation\_steps=len(xtest))

/usr/local/lib/python3.7/dist-packages/ipykernel\_launcher.py:8: UserWarning: `Model.fit\_generator` is deprecated and will be removed in a future version. Please use `Model.fit`, which supports generators.

Epoch 1/100

13/13 [==============================] - ETA: 0s - loss: 0.5002 - accuracy: 0.8174

WARNING:tensorflow:Learning rate reduction is conditioned on metric `val\_accuaracy` which is not available. Available metrics are: loss,accuracy,val\_loss,val\_accuracy,lr

13/13 [==============================] - 6s 503ms/step - loss: 0.5002 - accuracy: 0.8174 - val\_loss: 0.3022 - val\_accuracy: 0.8988 - lr: 0.0010

Epoch 2/100

13/13 [==============================] - ETA: 0s - loss: 0.4280 - accuracy: 0.8409

WARNING:tensorflow:Learning rate reduction is conditioned on metric `val\_accuaracy` which is not available. Available metrics are: loss,accuracy,val\_loss,val\_accuracy,lr

13/13 [==============================] - 6s 458ms/step - loss: 0.4280 - accuracy: 0.8409 - val\_loss: 0.4246 - val\_accuracy: 0.8466 - lr: 0.0010

Epoch 3/100

13/13 [==============================] - ETA: 0s - loss: 0.4018 - accuracy: 0.8522

WARNING:tensorflow:Learning rate reduction is conditioned on metric `val\_accuaracy` which is not available. Available metrics are: loss,accuracy,val\_loss,val\_accuracy,lr

13/13 [==============================] - 6s 469ms/step - loss: 0.4018 - accuracy: 0.8522 - val\_loss: 0.3677 - val\_accuracy: 0.8773 - lr: 0.0010

Epoch 4/100

13/13 [==============================] - ETA: 0s - loss: 0.3703 - accuracy: 0.8675

WARNING:tensorflow:Learning rate reduction is conditioned on metric `val\_accuaracy` which is not available. Available metrics are: loss,accuracy,val\_loss,val\_accuracy,lr

13/13 [==============================] - 6s 472ms/step - loss: 0.3703 - accuracy: 0.8675 - val\_loss: 0.2848 - val\_accuracy: 0.8681 - lr: 0.0010

Epoch 5/100

13/13 [==============================] - ETA: 0s - loss: 0.3317 - accuracy: 0.8821

WARNING:tensorflow:Learning rate reduction is conditioned on metric `val\_accuaracy` which is not available. Available metrics are: loss,accuracy,val\_loss,val\_accuracy,lr

13/13 [==============================] - 6s 472ms/step - loss: 0.3317 - accuracy: 0.8821 - val\_loss: 0.2316 - val\_accuracy: 0.9110 - lr: 0.0010

Epoch 6/100

13/13 [==============================] - ETA: 0s - loss: 0.2919 - accuracy: 0.8982

WARNING:tensorflow:Learning rate reduction is conditioned on metric `val\_accuaracy` which is not available. Available metrics are: loss,accuracy,val\_loss,val\_accuracy,lr

13/13 [==============================] - 6s 463ms/step - loss: 0.2919 - accuracy: 0.8982 - val\_loss: 0.2037 - val\_accuracy: 0.9141 - lr: 0.0010

Epoch 7/100

13/13 [==============================] - ETA: 0s - loss: 0.2653 - accuracy: 0.9168

WARNING:tensorflow:Learning rate reduction is conditioned on metric `val\_accuaracy` which is not available. Available metrics are: loss,accuracy,val\_loss,val\_accuracy,lr

13/13 [==============================] - 6s 472ms/step - loss: 0.2653 - accuracy: 0.9168 - val\_loss: 0.3426 - val\_accuracy: 0.8681 - lr: 0.0010

Epoch 8/100

13/13 [==============================] - ETA: 0s - loss: 0.2562 - accuracy: 0.9087

WARNING:tensorflow:Learning rate reduction is conditioned on metric `val\_accuaracy` which is not available. Available metrics are: loss,accuracy,val\_loss,val\_accuracy,lr

13/13 [==============================] - 6s 470ms/step - loss: 0.2562 - accuracy: 0.9087 - val\_loss: 0.1796 - val\_accuracy: 0.9264 - lr: 0.0010

Epoch 9/100

13/13 [==============================] - ETA: 0s - loss: 0.2512 - accuracy: 0.9192

WARNING:tensorflow:Learning rate reduction is conditioned on metric `val\_accuaracy` which is not available. Available metrics are: loss,accuracy,val\_loss,val\_accuracy,lr

13/13 [==============================] - 6s 489ms/step - loss: 0.2512 - accuracy: 0.9192 - val\_loss: 0.2118 - val\_accuracy: 0.9080 - lr: 0.0010

Epoch 10/100

13/13 [==============================] - ETA: 0s - loss: 0.2113 - accuracy: 0.9386

WARNING:tensorflow:Learning rate reduction is conditioned on metric `val\_accuaracy` which is not available. Available metrics are: loss,accuracy,val\_loss,val\_accuracy,lr

13/13 [==============================] - 6s 470ms/step - loss: 0.2113 - accuracy: 0.9386 - val\_loss: 0.2592 - val\_accuracy: 0.9141 - lr: 0.0010

Epoch 11/100

13/13 [==============================] - ETA: 0s - loss: 0.1966 - accuracy: 0.9443

WARNING:tensorflow:Learning rate reduction is conditioned on metric `val\_accuaracy` which is not available. Available metrics are: loss,accuracy,val\_loss,val\_accuracy,lr

13/13 [==============================] - 6s 458ms/step - loss: 0.1966 - accuracy: 0.9443 - val\_loss: 0.1606 - val\_accuracy: 0.9387 - lr: 0.0010

Epoch 12/100

13/13 [==============================] - ETA: 0s - loss: 0.1956 - accuracy: 0.9297

WARNING:tensorflow:Learning rate reduction is conditioned on metric `val\_accuaracy` which is not available. Available metrics are: loss,accuracy,val\_loss,val\_accuracy,lr

13/13 [==============================] - 6s 454ms/step - loss: 0.1956 - accuracy: 0.9297 - val\_loss: 0.1876 - val\_accuracy: 0.9325 - lr: 0.0010

Epoch 13/100

13/13 [==============================] - ETA: 0s - loss: 0.2050 - accuracy: 0.9330

WARNING:tensorflow:Learning rate reduction is conditioned on metric `val\_accuaracy` which is not available. Available metrics are: loss,accuracy,val\_loss,val\_accuracy,lr

13/13 [==============================] - 6s 467ms/step - loss: 0.2050 - accuracy: 0.9330 - val\_loss: 0.1395 - val\_accuracy: 0.9356 - lr: 0.0010

Epoch 14/100

13/13 [==============================] - ETA: 0s - loss: 0.1471 - accuracy: 0.9548

WARNING:tensorflow:Learning rate reduction is conditioned on metric `val\_accuaracy` which is not available. Available metrics are: loss,accuracy,val\_loss,val\_accuracy,lr

13/13 [==============================] - 6s 456ms/step - loss: 0.1471 - accuracy: 0.9548 - val\_loss: 0.1050 - val\_accuracy: 0.9601 - lr: 0.0010

Epoch 15/100

13/13 [==============================] - ETA: 0s - loss: 0.1326 - accuracy: 0.9564

WARNING:tensorflow:Learning rate reduction is conditioned on metric `val\_accuaracy` which is not available. Available metrics are: loss,accuracy,val\_loss,val\_accuracy,lr

13/13 [==============================] - 6s 478ms/step - loss: 0.1326 - accuracy: 0.9564 - val\_loss: 0.0418 - val\_accuracy: 0.9969 - lr: 0.0010

Epoch 16/100

13/13 [==============================] - ETA: 0s - loss: 0.1432 - accuracy: 0.9556

WARNING:tensorflow:Learning rate reduction is conditioned on metric `val\_accuaracy` which is not available. Available metrics are: loss,accuracy,val\_loss,val\_accuracy,lr

13/13 [==============================] - 6s 459ms/step - loss: 0.1432 - accuracy: 0.9556 - val\_loss: 0.1110 - val\_accuracy: 0.9724 - lr: 0.0010

Epoch 17/100

13/13 [==============================] - ETA: 0s - loss: 0.1214 - accuracy: 0.9669

WARNING:tensorflow:Learning rate reduction is conditioned on metric `val\_accuaracy` which is not available. Available metrics are: loss,accuracy,val\_loss,val\_accuracy,lr

13/13 [==============================] - 6s 472ms/step - loss: 0.1214 - accuracy: 0.9669 - val\_loss: 0.0772 - val\_accuracy: 0.9785 - lr: 0.0010

Epoch 18/100

13/13 [==============================] - ETA: 0s - loss: 0.0967 - accuracy: 0.9750

WARNING:tensorflow:Learning rate reduction is conditioned on metric `val\_accuaracy` which is not available. Available metrics are: loss,accuracy,val\_loss,val\_accuracy,lr

13/13 [==============================] - 6s 455ms/step - loss: 0.0967 - accuracy: 0.9750 - val\_loss: 0.0904 - val\_accuracy: 0.9693 - lr: 0.0010

Epoch 19/100

13/13 [==============================] - ETA: 0s - loss: 0.1162 - accuracy: 0.9637

WARNING:tensorflow:Learning rate reduction is conditioned on metric `val\_accuaracy` which is not available. Available metrics are: loss,accuracy,val\_loss,val\_accuracy,lr

13/13 [==============================] - 6s 511ms/step - loss: 0.1162 - accuracy: 0.9637 - val\_loss: 0.0409 - val\_accuracy: 0.9908 - lr: 0.0010

Epoch 20/100

13/13 [==============================] - ETA: 0s - loss: 0.0937 - accuracy: 0.9717

WARNING:tensorflow:Learning rate reduction is conditioned on metric `val\_accuaracy` which is not available. Available metrics are: loss,accuracy,val\_loss,val\_accuracy,lr

13/13 [==============================] - 6s 468ms/step - loss: 0.0937 - accuracy: 0.9717 - val\_loss: 0.0448 - val\_accuracy: 0.9877 - lr: 0.0010

Out[33]:

<keras.callbacks.History at 0x7f57902a8490>

In [34]:

*# Testing 4*

img = image.load\_img('/content/dataset/Testing/rats/images (55).jpeg',target\_size=(64,64)) *# Reading image*

x = image.img\_to\_array(img) *# Converting image into array*

x = np.expand\_dims(x,axis=0) *# expanding Dimensions*

pred = np.argmax(model.predict(x)) *# Predicting the higher probablity index*

op = ['bears','crows','elephants','rats'] *# Creating list*

op[pred] *# List indexing with output*

Out[34]:

'rats'

In [35]:

*# Testing google image*

img = image.load\_img('/content/42ny6cwj8t\_Polar\_bear\_on\_ice\_in\_Svalbard\_Norway\_WW294883.jpg',target\_size=(64,64)) *# Reading image*

x = image.img\_to\_array(img) *# Converting image into array*

x = np.expand\_dims(x,axis=0) *# expanding Dimensions*

pred = np.argmax(model.predict(x)) *# Predicting the higher probablity index*

op = ['bears','crows','elephants','rats'] *# Creating list*

op[pred] *# List indexing with output*

Out[35]:

'rats'

In [36]:

*# Testing google image*

img = image.load\_img('/content/images.jpg',target\_size=(64,64)) *# Reading image*

x = image.img\_to\_array(img) *# Converting image into array*

x = np.expand\_dims(x,axis=0) *# expanding Dimensions*

pred = np.argmax(model.predict(x)) *# Predicting the higher probablity index*

op = ['bears','crows','elephants','rats'] *# Creating list*

op[pred] *# List indexing with output*

Out[36]:

'elephants'

In [37]:

*# Testing google image*

img = image.load\_img('/content/01-rat-friends-nationalgeographic\_1162144\_16x9.jpg',target\_size=(64,64)) *# Reading image*

x = image.img\_to\_array(img) *# Converting image into array*

x = np.expand\_dims(x,axis=0) *# expanding Dimensions*

pred = np.argmax(model.predict(x)) *# Predicting the higher probablity index*

op = ['bears','crows','elephants','rats'] *# Creating list*

op[pred] *# List indexing with output*

Out[37]:

'bears'

In [38]:

*# Testing google image*

img = image.load\_img('/content/photo-1599921778557-082147629542.jpg',target\_size=(64,64)) *# Reading image*

x = image.img\_to\_array(img) *# Converting image into array*

x = np.expand\_dims(x,axis=0) *# expanding Dimensions*

pred = np.argmax(model.predict(x)) *# Predicting the higher probablity index*

op = ['bears','crows','elephants','rats'] *# Creating list*

op[pred] *# List indexing with output*

Out[38]:

'elephants'