פעילות 15 - רשת נוירונים מבוססת Keras לסיווג תוכן בתמונות

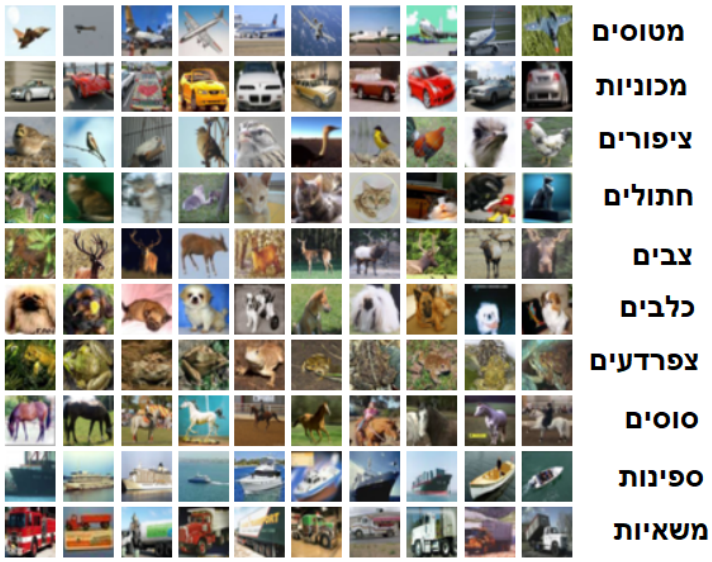
מקורות:

http://www.cs.toronto.edu/~kriz/cifar.html

https://gist.github.com/juliensimon/273bef4c5b4490c687b2f92ee721b546

https://keras.io/datasets/

בפעילות זה נתרגל את היסודות בכתיבת מכונה לומדת מבוססת על רשת נוירונים כדי לבצע סיווג תמונות לפי תוכן מצולם. בפעילות זו נעשה שימוש במאגר התמונות CIFAR-10 הכולל 60000 תמונות ברזולוציה של 32x32 פיקסלים. כל תמונה מסווגת לאחת מ- 10 קטגוריות. באיור הבא ניתן לראות דוגמה של 10 תמונות בכל קטגוריה.



מערך התמונות שב- CIFAR-10 מחולק ל- 50000 תמונות לצורך אימון ועוד 10000 תמונות לצורך בדיקה.

הספרייה Keras כוללת פעולה להורדה ישירה של מאגר התמונות ישירות למחשב. נבחן קוד העושה שימוש בפעולה cifar10.load_data כדי להוריד את המאגר למחשב האישי שלכם:

```
from keras.datasets import cifar10


print('Loading data...')
(train_data, train_lbl), (test_data, test_lbl) = cifar10.load_data()
print(len(train_data), 'train data')
print(len(test_data), 'test data')
```

נקבל את הפלט הבא:

```
Loading data...
50000 train data
10000 test data
```

נבחן את המבנה של מערך התמונות על ידי שימוש בפעולה shape כפי שהכרנו אותה מוקדם יותר כאשר למדנו על NumPy Arrays

```
print("train_data shape:", train_data.shape)
print("train_lbl shape:", train_lbl.shape)
print("test_data shape:", test_data.shape)
print("test_lbl shape:", test_lbl.shape)
```

נקבל את הפלט הבא:

```
train_data shape: (50000, 32, 32, 3)
train_lbl shape: (50000, 1)
test_data shape: (10000, 32, 32, 3)
test_lbl shape: (10000, 1)
```

ניתן לראות שאכן כל תמונה מורכבת מ- 32x32 פיקסלים, כאשר כל פיקסל מורכב מ-3 מספרים red, green, blue. מערך הסיווגים לתמונות (lbl) מכיל מספר בודד בין 0 ל-9.

דרך נוספת לעבוד על מערך התמונות CIFAR10 הוא להוריד את הקבצים מהאינטרנט ולשמור אותם בתיקייה בשם CIFAR10dataset במחשב שלכם.

את הקבצים יש להוריד דרך הקישור הבא:

https://www.cs.toronto.edu/~kriz/cifar-10-python.tar.gz

נקבל קובץ אחד דחוס הכולל את הקבצים הבאים:

Name	Size	Packed	Type	Modified
..			File folder	
test_batch	31,035,526	?	File	31/03/2009 7:32
readme.html	88	?	Chrome HTML Do...	04/06/2009 23:47
data_batch_5	31,035,623	?	File	31/03/2009 7:32
data_batch_4	31,035,696	?	File	31/03/2009 7:32
data_batch_3	31,035,999	?	File	31/03/2009 7:32
data_batch_2	31,035,320	?	File	31/03/2009 7:32
data_batch_1	31,035,704	?	File	31/03/2009 7:32
batches.meta	158	?	META File	31/03/2009 7:45

יש להעתיק את הקבצים data_batch ו- text_batch לתוך התיקייה CIFAR10dataset . כל קובץ מכיל 10000 תמונות. כמו כן הקובץ batches.meta מכיל את שמות הקטגוריות שעל פיהם מתויגות התמונות.
נבחן את תוכן הקובץ batches.meta על ידי הדוגמה באה:

```
import _pickle as pickle

f = open("CIFAR10dataset/batches.meta", 'rb')
meta = pickle.load(f, encoding='bytes')
print("category:", meta[b'label_names'])
```

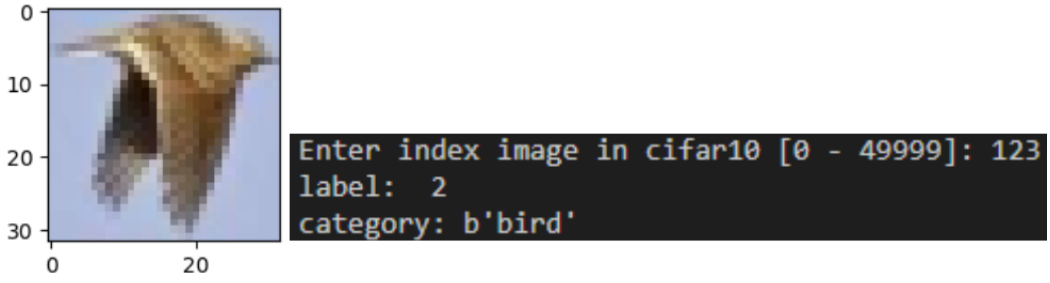
נקבל את הפלט הבא:

```
category: [b'airplane', b'automobile', b'bird', b'cat', b'deer', b'dog', b'frog', b'horse', b'ship', b'truck']
```

נעבור לחקור את 5 הקבצים המכילים את תמונות האימון. ניתן לבחון כל אחד 50000 התמונות על ידי הקוד הבא:

```
import _pickle as pickle
import numpy as np
import matplotlib.pyplot as plt

imageid = int(input("Enter index image in cifar10 [0 - 49999]: "))
batch = (imageid // 10000) + 1
idx = imageid - (batch-1)*10000

f1 = open("CIFAR10dataset/data_batch_" + str(batch), 'rb')
data = pickle.load(f1, encoding='bytes')
f2 = open("CIFAR10dataset/batches.meta", 'rb')
meta = pickle.load(f2, encoding='bytes')

im = data[b'data'][idx, :]
im_r = im[0:1024].reshape(32, 32)
im_g = im[1024:2048].reshape(32, 32)
im_b = im[2048:].reshape(32, 32)
img = np.dstack((im_r, im_g, im_b))

print("label: ", data[b'labels'][idx])
print("category:", meta[b'label_names'][data[b'labels'][idx]])
plt.imshow(img)
plt.show()
```

נקבל את הפלט הבא:



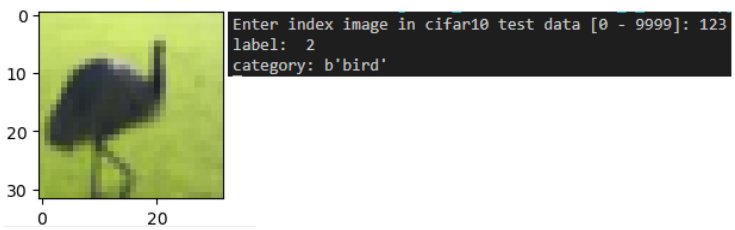
תרגיל

כתבו קוד המציג תמונה אחת לבחירה מתוך קובץ הבדיקה test_batch. קובץ זה מכיל 10000 תמונות.

פתרון

```
mport _pickle as pickle
import numpy as np
import matplotlib.pyplot as plt

imageid = int(input("Enter index image in cifar10 test data [0 - 9999]: "))

f1 = open("CIFAR10dataset/test_batch", 'rb')
data = pickle.load(f1, encoding='bytes')
f2 = open("CIFAR10dataset/batches.meta", 'rb')
meta = pickle.load(f2, encoding='bytes')

im = data[b'data'][imageid, :]

im_r = im[0:1024].reshape(32, 32)
im_g = im[1024:2048].reshape(32, 32)
im_b = im[2048:].reshape(32, 32)
img = np.dstack((im_r, im_g, im_b))

print("label: ", data[b'labels'][imageid])
print("category:", meta[b'label_names'][data[b'labels'][imageid]])

plt.imshow(img)
plt.show()
```

נקבל את הפלט הבא:



להלן קוד תוכנה הכולל את השלבים הבאים:

1. יבוא הספריות שבהם נשתמש לאמן את המכונה.
2. הורדת מערך התמונות למחשב והתאמתם למכונה.
3. בניית המודל שעליו תעבוד המכונה.
4. הידר המודל.
5. אימון ובדיקת המכונה.
6. פלט גרפים המכילים מידע על האימון.

```
from keras.datasets import cifar10
from keras.utils import np_utils
from keras.models import Sequential
from keras.layers.core import Dense, Dropout, Flatten
from keras.layers.convolutional import Conv2D, MaxPooling2D
from keras.optimizers import Adam
import matplotlib.pyplot as plt

# Download the photos and Tags to my computer and customize them
(train_data, train_lbl), (test_data, test_lbl) = cifar10.load_data()
print("train_data shape:", train_data.shape)
print(train_data.shape[0], "train samples")
print(test_data.shape[0], "test samples")
Y_train = np_utils.to_categorical(train_lbl, 10)
Y_test = np_utils.to_categorical(test_lbl, 10)
train_data = train_data.astype("float32")
test_data = test_data.astype("float32")
train_data /= 255
test_data /= 255

# Building the neuron network model
model = Sequential()
model.add(Conv2D(32, (3, 3), activation="relu", padding="same", input_shape=(32,32,3)))
model.add(Conv2D(32, (3, 3), activation="relu"))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(0.25))
model.add(Conv2D(64, (3, 3), activation="relu", padding="same"))
model.add(Conv2D(64, (3, 3), activation="relu"))
model.add(MaxPooling2D(pool_size=(2, 2)))
```

```
model.add(Dropout(0.25))
model.add(Flatten())
model.add(Dense(512, activation="relu"))
model.add(Dropout(0.5))
model.add(Dense(10, activation="softmax"))
model.summary()

# Compile the neuron network model
model.compile(      loss="categorical_crossentropy",
                    optimizer=Adam(lr=1.0e-4),
                    metrics=["accuracy"])

# Neuron network training
results = model.fit(train_data, Y_train,
                    batch_size=128,
                    epochs=1,
                    validation_split=0.2,
                    validation_data= (test_data, Y_test),
                    verbose=1)

# Saving the Neuron network training into 2 files
model_json = model.to_json()
open("C:/saved_models/CIFAR10model.json", "w").write(model_json)
model.save_weights("C:/saved_models/CIFAR10weights.h5", overwrite=True)

# Creating the system learning graph
plt.plot(results.history["val_accuracy"])
plt.ylabel("accuracy")
plt.xlabel("epoch")
plt.legend(["train", "test"], loc="upper right")
plt.show()
```

נקבל את הפלט הבא:

```
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_1 (Conv2D)            (None, 32, 32, 32)        896
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 30, 30, 32)        9248
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 15, 15, 32)        0
_________________________________________________________________
dropout_1 (Dropout)          (None, 15, 15, 32)        0
_________________________________________________________________
conv2d_3 (Conv2D)            (None, 15, 15, 64)        18496
_________________________________________________________________
conv2d_4 (Conv2D)            (None, 13, 13, 64)        36928
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (None, 6, 6, 64)          0
_________________________________________________________________
dropout_2 (Dropout)          (None, 6, 6, 64)          0
_________________________________________________________________
flatten_1 (Flatten)          (None, 2304)              0
_________________________________________________________________
dense_1 (Dense)              (None, 512)               1180160
_________________________________________________________________
dropout_3 (Dropout)          (None, 512)               0
_________________________________________________________________
dense_2 (Dense)              (None, 10)                5130
=================================================================
Total params: 1,250,858
```

מבנה הרשת הכולל מבוא הקולת תמונות ברזולציה של 28x28 פיקבלים. בהמשך מספר המרות מטריציוניות ובהמשך רשת הכוללת מעל מליון קשרים. סה"כ הרשת כולל 1250858 פרמטרים.

הערה:

שלב הלמידה של מכונה מסוג זה לוקח בין דקה למספר דקות כל אימון. כדי להגיע לתוצאות מספקות קבענו לבצע 100 אימונים. משמעות הדבר שזמן האימון הכללי של הרשת נמשך בין שעה למספר שעות בהתאם למאפייני המחשב ובמיוחד האם ואיזה סוג של כרטיס גרפי GPU קיים במחשב.

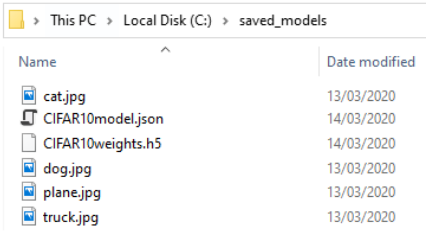
המחשב שלי לא כולל מאיץ גרפי GPU על כן זמן הלמידה לקח שלוש וחצי שעות!!!

כדי לא לגרור את התוכנה שלנו לאימון של מעל שעה בכל פעם שנבצע בה שינויים. הוספתי בסוף התוכנית קוד השומר את מודל רשת הנוירונים בקובץ חיצוני בשם CIFAR10model.json ואת כלל מערך המשקלים בקובץ נוסף בשם CIFAR10weights.h5. את שני הקבצים נשמור בספרייה בשם saved_models בכונן C.

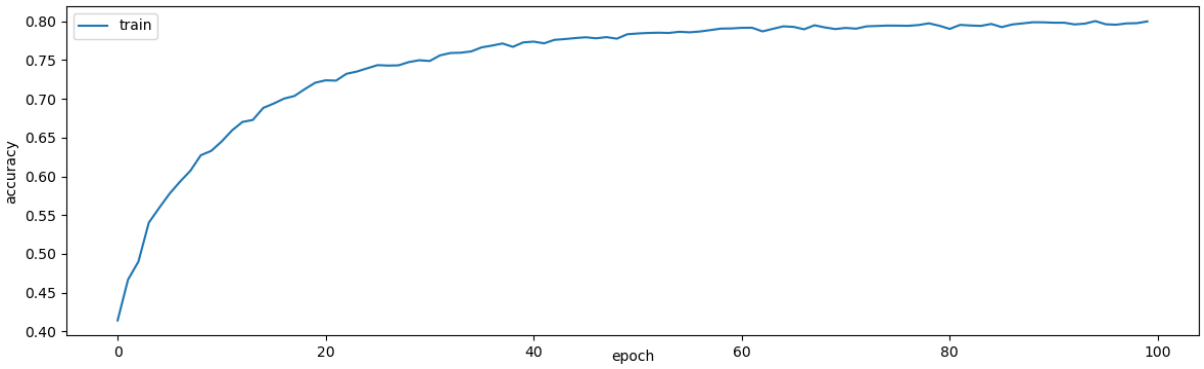
להלן הקוד:

```
model_json = model.to_json()
open("C:/saved_models/CIFAR10model.json", "w").write(model_json)
model.save_weights("C:/saved_models/CIFAR10weights.h5", overwrite=True)
```

בשלב הבא נכתוב קוד הקורא את 2 הקבצים בונה את רשת הנוירונים לפי תוכן קובץ CIFAR10model.json בהמשך מכוון את משקלי הרשת על פי תוכן קובץ CIFAR10weights.h5 ובכך אנו מדלגים על הצורך באימונים חוזרים ונשנים בכל פעם שמריצים את התוכנית. מידע מעמיק יותר בנושא חשוב זה נלמד בהמשך המדריך בפעילות 18 בנושא גיבוי ושחזור מערך המשקלים של רשת נוירונים. להלן דוגמה לתוכן הספרייה saved_models:



פלט תוצאות האימון בגרף:



ניתן לראות שלאחר 100 אימונים כאשר כל אימון מבצע עיבוד של 50000 תמונות אימון ועוד 10000 תמונות בדיקה הגענו במערכת המסוגלת לסווג תוכן בתמונות ברמה של מעל 80 אחוז הצלחה. כמו כן ניתן לראות תוצאות דומות התקבלו כבר לאחר 60 אימונים.

בדיקת המכונה על ידי תמונות חדשות שלא מתוך מאגר התמונות

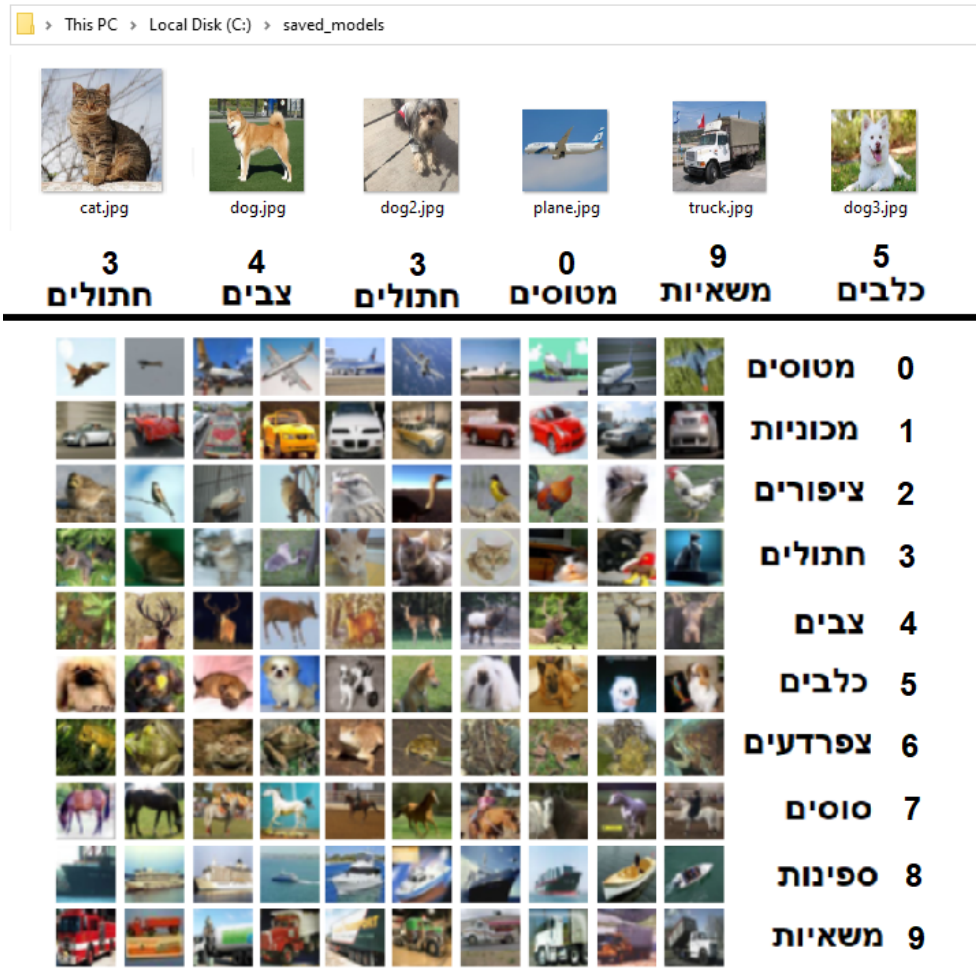
נכתוב קוד הקורא את 2 הקבצים CIFAR10model.json ו- CIFAR10weights.h5 כדי לבנות מחדש את מודל המכונה. אז נבדוק את יכולות המכונה על תמונות חדשות מהאינטרנט. להלן קוד התוכנית:

```
from keras.models import model_from_json
from keras.optimizers import Adam
from keras.preprocessing import image
# Load the Neuron network training from 2 files
model_architecture = "C:/saved_models/CIFAR10model.json"
model_weights = "C:/saved_models/CIFAR10weights.h5"
model = model_from_json(open(model_architecture).read())
model.load_weights(model_weights)
# Lode the image from my computer and do customize
MyPIC = image.load_img('C:/saved_models/dog.jpg', target_size=(32,32))
MyPIC = image.img_to_array(MyPIC)
MyPIC = MyPIC.reshape((1,) + MyPIC.shape)
MyPIC = MyPIC/255.
# Compile the neuron network model
model.compile(     loss="categorical_crossentropy",
                   optimizer=Adam(lr=1.0e-4),
                   metrics=["accuracy"])
# Predicting the image
predictions = model.predict_classes(MyPIC)
```

```
print(predictions)
```

נדגום מהאינטרנט מספר תמונות לבדיקה ונבחן את התוצאות.

להלן תוצאות הבדיקה:



ניתן ללמוד שהמכונה מזהה את תמונת המטוס החתול והמשאית אך מתקשה לזהות את הכלב. עושה הרושם שכלב קטן במימדים של חתול מזוהה כחתול וכלב עם רגליים ארוכות מזוהה כצבי. רק כלב שהראש שלו מצולם במרכז התמונה מזוהה נכון.

תנאי השימוש

תנאי השימוש במסמך זה הם לפי הסטנדרט הבא: