אוסף הנתונים

אוסף הנתונים של MNIST, המכיל 70 אלף תמונות של הספרות 0-9 בכתב יד, הוא אחד המפורסמים ביותר בתחום למידת המכונה, ומשמש לרוב בתור הדוגמה הראשונה עליה מפעילים אלגוריתם לסיווג: הקלט הוא אחת התמונות והפלט הדרוש הוא זיהוי ספרה המופיעה בתמונה. למרות הפופולריות שלו, בימינו אוסף זה מהווה אתגר קל מדי – אפילו האלגוריתמים הפשוטים ביותר מצליחים לסווג את הספרות נכונה בדיוק של מעל 95%. בהתאם, בהמשך אנו נשתמש באוסף נתונים בעל מאפיינים דומים – 70 אלף תמונות המחולקות ל10 מחלקות שונות, כאשר הפעם מדובר ב**פריטי לבוש** במקום ספרות. זהו אוסף הנתונים Fashion-MNIST. היתרון בשימוש באוסף זה הוא שסיווג פריטי הלבוש היא משימה יותר מאתגרת, אשר תאפשר לנו לבחון את השיפור בביצועי רשת הנוירונים, ככל שנוסיף לה רכיבים מתקדמים בעתיד.

לפני בניית ואימון הרשת, נרצה לייבא את אוסף הנתונים, ולטעון אותו לזיכרון המחשב. ניתן לעשות זאת ישירות בעזרת הספרייה PyTorch, המכילה ממשק לייבוא אוספי נתונים פופולריים. בקטע הקוד הבא אנו מורידים את אוסף הנתונים וטוענים אותו לתוך המשתנה `mnist_train`.

```
import torch
import torchvision
mnist_train = torchvision.datasets.FashionMNIST(
    root="/22961", train=True, download=True)
```

לאחר ריצת הקוד בהצלחה, אוסף הנתונים יישמר באופן מקומי בתיקייה הנקובה בפרמטר `root`, ויתקבל אובייקט דאטא לתוך המשתנה `mnist_train`, אותו נחקור עתה.

```
print(len(mnist_train))
mnist_train.classes
```

פלט:

```
60000
['T-shirt/top',
 'Trouser',
 'Pullover',
 'Dress',
 'Coat',
 'Sandal',
 'Shirt',
 'Sneaker',
 'Bag',
 'Ankle boot']
```

הפקודה הראשונה מדפיסה את אורכו של אוסף הנתונים, הוא 60 אלף פריטים. זהו סט האימון (training set), כאשר עשרת אלפים תמונות נוספות שמורות לבדיקת האלגוריתמים – סט הבדיקה (test set). את תמונות אלו אפשר להוריד גם כן, על ידי שינוי הפרמטר `train` של פונקציית הייבוא ל-`False`. כמו כן, במאפיין `classes` אנו רואים את שמות המחלקות, ואכן ניכר שמדובר בפרטי לבוש.

כעת נטען את אחת התמונות לזכרון:

```
A=mnist_train[0]
print(type(A),len(A),A[0],
      A[1],mnist_train.classes[A[1]],
      sep='\n')
```

פלט:

```
<class 'tuple'>
2
<PIL.Image.Image image mode=L size=28x28 at 0x7EFEE80C8CD0>
9
Ankle boot
```

ניתן לראות שהנתונים נטענים בצורה של זוג סדור, כאשר האיבר הראשון הוא תמונה בפורמט PIL והשני הוא סיווג התמונה. נבדוק זאת, אך קודם כל נמיר את התמונה לטנזור, בכדי שנוכל להמשיך בעבודה עם הכלים המוכרים לנו.

```
convert=torchvision.transforms.PILToTensor()
img=convert(A[0])
print(type(img),img.size(),sep='\n')
```

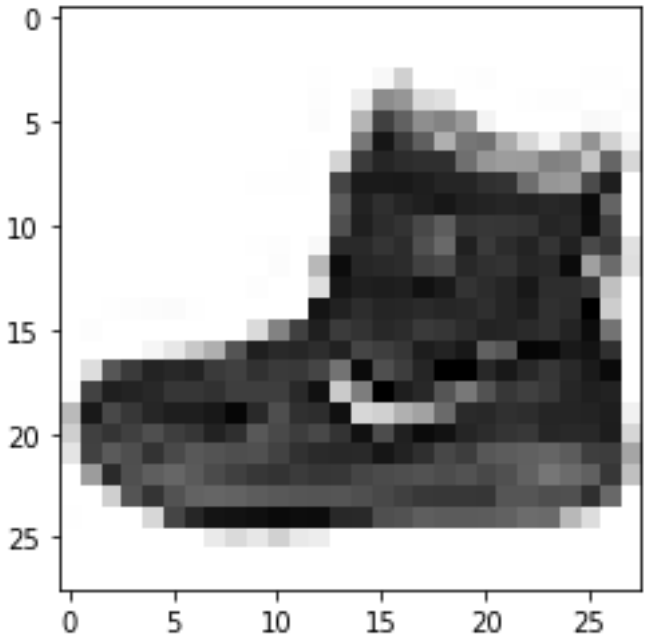
פלט:

```
<class 'torch.Tensor'>
torch.Size([1, 28, 28])
```

נצייר את התמונה בעזרת הפונקציה `imshow` של Matplotlib, אך נשים לב שהקלט הדרוש לפונקציה זו הוא מטריצה (טנזור דו מימדי) ועל כן עלינו להעלים את המימד הראשון המנוון בטנזור שלנו בעזרת הפקודה `squeeze`.

```
import matplotlib.pyplot as plt
plt.imshow(torch.squeeze(img),cmap='Greys');
```

פלט:



אכן, אנו נוכחים כי זהו מגף, המתאים לסיווג הנתון למחלקה מספר 9.

בהמשך, לצורך ייעול תהליך האימון, נרצה לטעון את הנתונים במנות קטנות (minibatches), למשל 64 תמונות בו זמנית, וכן לעשות זאת במקביל על רכיבי חומרה נפרדים, כגון מאיצים גרפיים או TPU (Tensor Processing Unit) – רכיבים ייעודים לעיבוד טנזורי. לצורך זה נשתמש בכלי `DataLoader` של PyTorch, המקבל כקלט אובייקט `Dataset` כגון אוסף הנתונים שאיתו עבדנו עד כה.

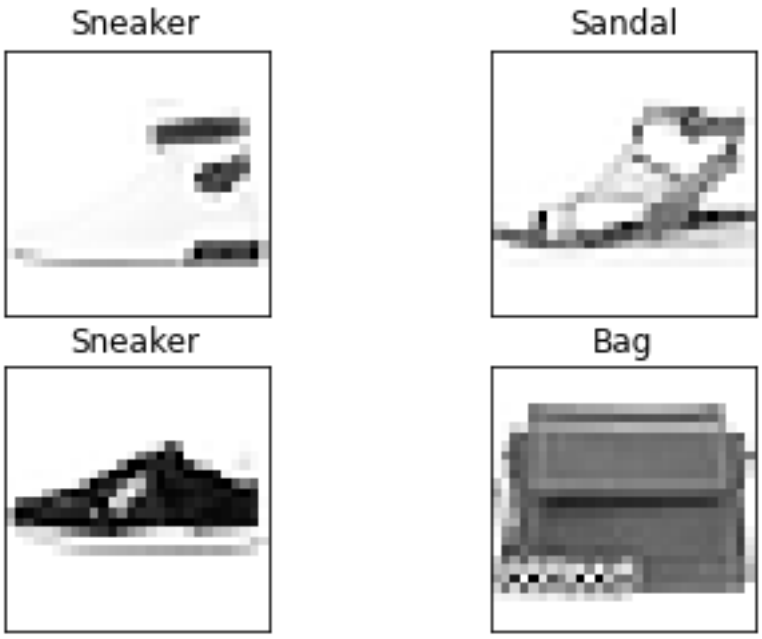
לאחר ריצת הקוד הבא נוכל להשתמש ב-`train_dataloader` לעבור על אוסף הנתונים בצורה יעילה. שימו לב שעבור השימוש בטוען הנתונים יש לשלב את המרת התמונות מPIL לטנזור כבר בשלב יצירת ה-`Dataset`.

```
from torch.utils.data import DataLoader
train_data_transformed = torchvision.datasets.FashionMNIST(
    root="/22961", train=True, download=False,
    transform=torchvision.transforms.PILToTensor())

train_dataloader = DataLoader(
    train_data_transformed, batch_size=4)
```

על ידי איטרציה על האובייקט שהתקבל, נוכל לקבל ארבע תמונות חדשות, ולציירן יחד עם המחלקה המתאימה שלהן.

```
imgs, labels = next(iter(train_dataloader))
class_names= train_data_transformed.classes

fig = plt.figure()
for i in range(4):
  ax = fig.add_subplot(2, 2, i+1)
  plt.imshow(torch.squeeze(imgs[i]),cmap='Greys')
  ax.set_title(class_names[labels[i]])
  ax.axes.get_xaxis().set_visible(False)
  ax.axes.get_yaxis().set_visible(False)
```

פלט:



בדוגמה זו אנו מכירים לראשונה לעומק את דרך הפעולה של החבילה Matplotlib ועל כן נסביר בפירוט את הפקודות והלוגיקה:

1. הפקודה `plt.figure()` יוצרת אובייקט תמונה אשר יכיל את כל שנצייר בהמשך.
2. הפקודה `add_subplot(2, 2, x)` מחלקת את התמונה לשתי שורות ושתי עמודות, ונותנת לנו גישה לצייר באחד הרביעים שנוצרו. למשל כאשר `x=1` נצייר לתוך הרביע השמאלי העליון וכאשר `x=3` נצייר לתוך הרביע השמאלי התחתון.
3. הפקודות `ax.set_title`, `ax.axes.get_xaxis` משנות את המאפיינים של מערכת הצירים הנוכחית (אחד הרביעים הנ"ל).

קיים מגוון גדול של מאפייני מערכת צירים בהם ניתן לשלוט בצורה דומה, המשנים במעט את הנראות של התמונה. אין צורך לזכור אותם בעל פה, אלא כרגיל לחפש את הרצוי בתיעוד של Matplotlib.

שאלות לתרגול

1. בשאלה זו יהיה עליכם לבנות בעצמכם אובייקט `Dataset` אשר יכיל את אוסף הנתונים Fashion-MNIST ולטעון תמונות ממנו בעזרת `DataLoader`.
   א. הורידו את אוסף הנתונים Fashion-MNIST מהקישור ל-github המופיע באתר הקורס.
   ב. כתבו מחלקה בשם `FashionMNISTDataset` היורשת מהמחלקה `Dataset` של Pytorch. עליכם לממש 3 מתודות:

- **`__init__(self, labels_file, dir, transform=None, target_transform=None)`**

  מתודה זו מאתחלת את האובייקט, והקלט שלה הוא מיקום קובץ הסיווגים של כל תמונה, ומיקום התיקייה בה מופיעות התמונות.
- **`__len__(self)`**

  מתודה זו מחזירה את מספר הדגימות באוסף הנתונים.
- **`__getitem__(self, idx)`**

  מתודה זו מחזירה את הדגימה ה-**idx** באוסף הנתונים. הפלט שלה הוא מהצורה: **return image, label**.

   ג. אתחלו אובייקט מהמחלקה הזו המצביע את התיקייה בה נמצא אוסף הנתונים.
   ד. צרו אובייקט `DataLoader` העוטף את האובייקט הקודם.
   ה. טענו נתונים בעזרת ה-`DataLoader` וציירו אותם בעזרת הפקודה `plt.imshow()`.

הערה: מומלץ להיעזר בתיעוד של PyTorch לפתרון שאלה זו. קישור מופיע באתר הקורס.