אתחול tensor

כדי לעבוד עם טנזורים, עלינו ראשית לאתחל אותם בזיכרון ולשם כך קיימים בנאים (קונסטרקטורים) רבים.

הדרך הבסיסית ביותר לאתחל טנזור היא על ידי העברת מערך פייתוני או `ndarray` של NumPy כפרמטר לבנאי `torch.tensor`, כפי שניתן לראות על ידי הרצת קטע הקוד הבא במחברת ג'ופיטר:

```
import torch
import numpy as np
x = torch.tensor([[1,2],[3,4]])
```

לאחר הריצה, נדפיס את המשתנה `x` ומספר תכונות שלו כך:

```
print(x,x.shape,x.dtype,sep='\n')
```

פלט:

```
tensor([[1, 2],
        [3, 4]])
torch.Size([2, 2])
torch.int64
```

מפלט זה ניתן ללמוד כי `x` הוא אובייקט `tensor` המכיל מטריצת נתונים בגודל 2 על 2, וכל אחד מאיבריה הוא משתנה מאותו טיפוס: `int64`.

אם נשנה במעט את הקלט של הבנאי, למשל על ידי שינוי האיבר הראשון לטיפוס ממשי, נקבל את התוצאה הבאה:

```
x = torch.tensor([1.0,2,3,4])
```

פלט:

```
tensor([1., 2., 3., 4.])
torch.Size([4])
torch.float32
```

ומכך אנו למדים כי כל איברי המערך הומרו לטיפוס ממשי – הטיפוס של כל איבר במטריצה חייב להיות זהה. מבנה הנתונים תומך בטיפוסי הנתונים המספריים הסטנדרטיים של מספרים שלמים, ממשיים, מרוכבים וכן בטיפוס בוליאני.

לרוב יהיה ברצוננו לאתחל טנזורים לפי חוקיות מסוימת, למשל בתור מטריצת היחידה או על ידי דגימת ערכים אקראיים מהתפלגות נתונה, ולמטרות אלו קיימים בנאים אחרים. מספר דוגמאות לכך מופיעות בקטע הקוד הבא:

File #0004878 belongs to Shlomi Domnenko- do not distribute

```
x = torch.arange(9)
y = torch.eye(3,dtype=torch.bool)
z = torch.randn(size=[2, 3, 3])
print(x,y,z,sep='\n')
```

פלט:

```
tensor([0, 1, 2, 3, 4, 5, 6, 7, 8])
tensor([[ True, False, False],
        [False,  True, False],
        [False, False,  True]])
tensor([[[-0.8441,  0.5493,  0.2198],
         [-1.4702, -1.2846,  0.1339],
         [ 1.3861,  1.7858,  0.5074]],

        [[-1.0042, -0.9809,  0.4707],
         [-0.1001, -2.6261,  0.8152],
         [-1.1585,  0.8810, -0.2189]]])
```

עלינו לשים לב כי הטנזור `z` תלת ממדי – הוא מורכב משתי מטריצות 3x3. למעשה, אין מגבלה על מספר הממדים של אובייקט `tensor`, ולעתים קרובות הברירה הנוחה ביותר לשימושינו תהיה של טנזורים רבי ממדים.

ישנו מספר רב של בנאים נוספים, ואין טעם בסקירה ממצה של כולם – כל שיש לדעת הוא שלרוב הצרכים אשר יעלו בעת מימוש רשתות נוירונים קיים הבנאי המתאים, אותו יש למצוא אותו בתיעוד של הספרייה PyTorch.

שאלות לתרגול

1. צרו רשימה סטנדרטית של פייתון המכילה משתנים מסוגים שונים (float, int, char וכו') ונסו להמירה לטנזור של PyTorch. האם הצלחתם? אם לא, האם תוכלו להסביר למה?
2. צרו טנזור 4 מימדי של ביטים אקראיים לחלוטין (0 או 1 בהסתברות שווה). היעזרו בקישור לתיעוד של PyTorch המופיע באתר הקורס.