גישה לנתונים השמורים בטנזור

לנתונים השמורים בטנזור ניתן לגשת כמו לכל מערך פייתוני רגיל, ובדומה ניתן כך גם לכתוב נתונים לתוך הטנזור. ראו לדוגמה:

```
x = torch.arange(10)
print(x)
```

פלט:

```
tensor([0, 1, 2, 3, 4, 5, 6, 7, 8, 9])
```

זכרו שהאינדקס של האיבר הראשון במערך פייתוני הוא 0 בעוד שניתן לגשת אל האיבר האחרון במערך עם האינדקס השלילי 1-.

```
x[-1] = 100
print(x[2],x[-1],sep='\n')
```

פלט:

```
tensor(2)
tensor(100)
```

בדומה למבנה הנתונים של הספרייה NumPy, ישנן דרכים מתקדמות לגשת לאיברים ספציפיים בטנזור הכוללות אינדקסים רב ממדיים. לצורך דוגמה נשנה את צורת הטנזור הקודם:

```
x = torch.arange(10).reshape(2,5)
print(x)
```

פלט:

```
tensor([[0, 1, 2, 3, 4],
        [5, 6, 7, 8, 9]])
```

כעת x הוא מטריצה, ולאיבריה ניתן לגשת בעזרת אינדקס שורה ועמודה:

```
print(x[1,2],x[0,-1],sep='\n')
```

פלט:

```
tensor(7)
tensor(4)
```

שיטה זו עובדת עבור טנזור מכל גודל, ועבור כל אחד מממדיו, ראו כאן לדוגמה:

```
y = torch.zeros(size=(2,3,4))
y[1,0,0] = -2
y[-1,-1,-1] =  1
print(y)
```

פלט:

```
tensor([[[ 0.,  0.,  0.,  0.],
         [ 0.,  0.,  0.,  0.],
         [ 0.,  0.,  0.,  0.]],

        [[-2.,  0.,  0.,  0.],
         [ 0.,  0.,  0.,  0.],
         [ 0.,  0.,  0.,  1.]]])
```

בדומה ניתן לקבל **חתך (slice)** מהמטריצה, על ידי מתן טווח ערכים באחד האינדקסים. עבור y הנ"ל, בדוגמה הבאה נציב 100 בכל האיברים אשר אינדקס הממד הראשון שלהם הוא 1, אינדקס המימד השני הוא 1 או 2 ואינדקס המימד השלישי הוא כל המספרים בין אפס (כולל) לבין 4 (לא כולל), בקפיצות של 2:

```
y[1,1:2,0:4:2]=100
print(y)
```

פלט:

```
tensor([[[  0.,   0.,   0.,   0.],
         [  0.,   0.,   0.,   0.],
         [  0.,   0.,   0.,   0.]],

        [[ -2.,   0.,   0.,   0.],
         [100.,   0., 100.,   0.],
         [  0.,   0.,   0.,   1.]]])
```

כמובן שבדרך זו אפשר גם לאחזר גם את המידע השמור בy ולא רק לכתוב לתוכו.

דרך נוספת ושימושית ביותר לגשת לאיברי הטנזור היא בעזרת **אינדקסים בוליאניים** (Masks), בעזרתם ניתן לחלץ מתוך הטנזור איברים המקיימים תנאי מסוים. לדוגמה:

```
print(y[y>0])
```

פלט:

```
tensor([100., 100.,   1.])
```

למעשה, הפעולה האלגנטית הנ"ל מתבצעת בשני שלבים: ראשית מחושב טנזור בוליאני בעל אותו ממד כשל y. אחרי כן, הטנזור הבוליאני מועבר כאינדקס לy ומוחזרים רק הערכים אשר באותו מקום יש ערך אמת בטנזור הבוליאני. ניתן להמחיש זאת על ידי פיצול החישוב לשני שלבים:

```
z = y>0
print(z,y[z],sep='\n')
```

פלט:

```
tensor([[[False, False, False, False],
         [False, False, False, False],
         [False, False, False, False]],

        [[False, False, False, False],
         [ True, False,  True, False],
         [False, False, False,  True]]])
tensor([100., 100.,   1.])
```

יש לשים לב בשיטה זו הפלט המתקבל הינו תמיד חד ממדי.

טנזור האינדקס הבוליאני יכול להיות גם ממד נמוך יותר. במקרה זה הפלט יהיה רב ממדי:

```
print(y[[False,True]])
```

פלט:

```
tensor([[[ -2.,   0.,   0.,   0.],
         [100.,   0., 100.,   0.],
         [  0.,   0.,   0.,   1.]]])
```

בדוגמה זו ניתן לראות שהאינדקס הבוליאני התייחס לממד הראשון של `y` והחזיר למעשה את כל המידע בעל אינדקס ממד ראשון 1: `y[1,:,:]`. זהו הכלל במקרה שהאינדקס ממימד נמוך יותר: הוא מתייחס לממדים הראשונים, ואינו משפיע על שאר הממדים.

שאלות לתרגול

1. דגמו טנזור X בגודל 10X10 של משתנים מקריים נורמליים אחידים וצרו טנזור Y בעל אותם מימדים המכיל רק ערכי Inf. השתמשו בשניהם להחזיר טנזור מאותו גודל אשר מכיל את האיברים החיוביים של X, ובמקום האיברים השליליים – Inf. **רמז:** כדאי להשתמש בפונקציה `torch.where()`.
2. שנו את הטנזור X מהשאלה הקודמת לטנזור תלת מימדי בגודל 10X5X2, וצרו חתך שלו המכיל רק את האיברים אשר שלושת האינדקסים שלהם אי זוגיים.