שם מגיש: יונתן שירמן

ת.ז: 208870246

מטלה מס' 2 – מבוא ללמידת מכונה

TENSORFLOW on MNIST

במטלה זו אבחן את השינוי שנוצר בעקבות שימוש בפרמטרים שונים עם שכבות שונות על דיוק תוצאות המודל. קובץ זה מאגד 15 ניסויים עם תוצאותיהם כאשר לכל ניסוי יהיו פרמטרים שונים היצוינו בכל חלק המוקדש לכל ניסוי. המודל עם הפרמטרים הטובים ביותר יעלה לGitHub. כל ניסוי ביצע 50 EPOCH'S , כלומר תהליך האימון רץ 50 פעמים על בסיס הנתונים שהוקצע לאימון.

ניסוי מס' 1

בניסוי זה השתמשתי ב-4 שכבות כאשר התפלגות מס' הנוירונים (לפי מספר ת.ז מהסוף להתחלה) בכל שכבה היא:

1. שכבה 1: 46
2. שכבה 2: 02 (2)
3. שכבה 3: 87
4. שכבה 4: 08 (8)

סה"כ פרמטרים: 37259 (משקלי W)

על שיכבה בוצעה הפעלת Sigmoid בלבד. להלן הפרמטרים לניסוי:

```
layers = [
  tf.keras.layers.Flatten(input_shape=image_shape),

  tf.keras.layers.Dense(46),
  tf.keras.layers.Activation('sigmoid'),

   tf.keras.layers.Dense(2),
  tf.keras.layers.Activation('sigmoid'),

   tf.keras.layers.Dense(87),
  tf.keras.layers.Activation('sigmoid'),

   tf.keras.layers.Dense(8),
  tf.keras.layers.Activation('sigmoid'),

  tf.keras.layers.Dense(num_of_classes),
  tf.keras.layers.Softmax()
  ]
```

Training accuracy (sparse_categorical_accuracy): 0.9516

Validation accuracy (val_sparse_categorical_accuracy): 0.9338

ניסוי מס' 2

בניסוי זה השתמשתי ב-4 שכבות כאשר התפלגות מס' הנוירונים (לפי מספר ת.ז מהסוף להתחלה) בכל שכבה היא:

1. שכבה 1: 46
2. שכבה 2: 02 (2)
3. שכבה 3: 87
4. שכבה 4: 08 (8)

סה"כ פרמטרים: 37529 (משקלי W)

על שיכבה בוצעה הפעלת relu בלבד. להלן הקוד:

```
layers = [
  tf.keras.layers.Flatten(input_shape=image_shape),

  tf.keras.layers.Dense(46),
  tf.keras.layers.Activation('relu'),

  tf.keras.layers.Dense(2),
  tf.keras.layers.Activation('relu'),

  tf.keras.layers.Dense(87),
  tf.keras.layers.Activation('relu'),

  tf.keras.layers.Dense(8),
  tf.keras.layers.Activation('relu'),

  tf.keras.layers.Dense(num_of_classes),
  tf.keras.layers.Softmax()
  ]
```

Training accuracy (sparse_categorical_accuracy): 0.9605

.Validation accuracy (val_sparse_categorical_accuracy): 0.9366

ניסוי מס' 3

בניסוי זה השתמשתי ב-4 שכבות עם התפלגות נוריונים כפי שהופיע בניסויים הקודמים, הפעלת SIGMOID, ורגולריזציה l2:

```
[13] layers = [
       tf.keras.layers.Flatten(input_shape=image_shape),

       tf.keras.layers.Dense(46, kernel_regularizer=tf.keras.regularizers.l2(0.0001)),
       tf.keras.layers.Activation('sigmoid'),

        tf.keras.layers.Dense(2, kernel_regularizer=tf.keras.regularizers.l2(0.001)),
       tf.keras.layers.Activation('sigmoid'),

        tf.keras.layers.Dense(87, kernel_regularizer=tf.keras.regularizers.l2(0.01)),
       tf.keras.layers.Activation('sigmoid'),

        tf.keras.layers.Dense(8, kernel_regularizer=tf.keras.regularizers.l2(0.1)),
       tf.keras.layers.Activation('sigmoid'),

       tf.keras.layers.Dense(num_of_classes),
       tf.keras.layers.Softmax()
       ]
```

כל ההרצות (EPOCH) הציגו שתי תוצאות דיוק בלבד:

Validation accuracy (val_sparse_categorical_accuracy): 0.1010

Validation accuracy (val_sparse_categorical_accuracy): 0.1135

כאשר דיוק הלמידה עמד בטווח:

Training accuracy (sparse_categorical_accuracy): 0.1027-0.1146

לדעתי הפרמטרים בניסוי זה לא מאפשרים למידה לכן הדיוק נשאר נמוך ובאותו ערך. הניסוי המתבקש הבא לשנות לרגולריזציה מסוג l1 ולראות אם יש שוני:

ניסוי מס' 4

כאמור, שונתה הרגולריזציה מ-l2 ל-l1 והתוצאות לא היו שונות.

```
[16] layers = [
       tf.keras.layers.Flatten(input_shape=image_shape),

       tf.keras.layers.Dense(46, kernel_regularizer=tf.keras.regularizers.l1(0.0001)),
       tf.keras.layers.Activation('sigmoid'),

        tf.keras.layers.Dense(2, kernel_regularizer=tf.keras.regularizers.l1(0.001)),
       tf.keras.layers.Activation('sigmoid'),

        tf.keras.layers.Dense(87, kernel_regularizer=tf.keras.regularizers.l1(0.01)),
       tf.keras.layers.Activation('sigmoid'),

        tf.keras.layers.Dense(8, kernel_regularizer=tf.keras.regularizers.l1(0.1)),
       tf.keras.layers.Activation('sigmoid'),

       tf.keras.layers.Dense(num_of_classes),
       tf.keras.layers.Softmax()
       ]
```

למרות השינוי בפורמט הרגוליזרציה מ-l2 ל-l1, הדיוק נשאר על אותם ערכים:

כל ההרצות (EPOCH) הציגו שתי תוצאות דיוק בלבד:

Validation accuracy (val_sparse_categorical_accuracy): 0.1010

Validation accuracy (val_sparse_categorical_accuracy): 0.1135

כאשר דיוק הלמידה עמד בטווח:

Training accuracy (sparse_categorical_accuracy): 0.1027-0.1146

לא בוצעה למידה, הניחושים של המודל היו נוראיים ולא השתפרו – המודל לא למד.

ניסוי מס' 5

בניסוי זה השתמשתי ב-4 שכבות עם התפלגות נוריונים כפי שהופיע בניסויים הקודמים, הפעלת RELU, ורגולריזציה l2:

```
layers = [
  tf.keras.layers.Flatten(input_shape=image_shape),

  tf.keras.layers.Dense(46, kernel_regularizer=tf.keras.regularizers.l2(0.0001)),
  tf.keras.layers.Activation('relu'),

  tf.keras.layers.Dense(2, kernel_regularizer=tf.keras.regularizers.l2(0.001)),
  tf.keras.layers.Activation('relu'),

  tf.keras.layers.Dense(87, kernel_regularizer=tf.keras.regularizers.l2(0.01)),
  tf.keras.layers.Activation('relu'),

  tf.keras.layers.Dense(8, kernel_regularizer=tf.keras.regularizers.l2(0.1)),
  tf.keras.layers.Activation('relu'),

  tf.keras.layers.Dense(num_of_classes),
  tf.keras.layers.Softmax()
  ]
```

התוצאות טובות בהרבה מהפעלת SIGMOID, התוצאות התקבלו:

Training accuracy (sparse_categorical_accuracy): 0.9437

Validation accuracy (val_sparse_categorical_accuracy): 0.9375

ניסוי מס' 6

בדומה מאוד לניסוי מס' 5, ערך הרגלוריזציה בשכבה האחרונה השתנה מ-0.1 ל-0.01:

```
layers = [
  tf.keras.layers.Flatten(input_shape=image_shape),

  tf.keras.layers.Dense(46, kernel_regularizer=tf.keras.regularizers.l2(0.0001)),
  tf.keras.layers.Activation('relu'),

  tf.keras.layers.Dense(2, kernel_regularizer=tf.keras.regularizers.l2(0.001)),
  tf.keras.layers.Activation('relu'),

  tf.keras.layers.Dense(87, kernel_regularizer=tf.keras.regularizers.l2(0.01)),
  tf.keras.layers.Activation('relu'),

  tf.keras.layers.Dense(8, kernel_regularizer=tf.keras.regularizers.l2(0.001)),
  tf.keras.layers.Activation('relu'),

  tf.keras.layers.Dense(num_of_classes),
  tf.keras.layers.Softmax()
  ]
```

תוצאת הדיוק כמעט זהה לניסוי הקודם עם תוצאות:

Training accuracy (sparse_categorical_accuracy): 0.9465

Validation accuracy (val_sparse_categorical_accuracy): 0.9344

ניסוי מס' 7

דומה מאוד לניסויים 5 ו-6, 4 שכבות, אקטיבציה RULE רק שבניסוי זה הרגולריזציה תתבסס על סוג l1:

```
layers = [
  tf.keras.layers.Flatten(input_shape=image_shape),

  tf.keras.layers.Dense(46, kernel_regularizer=tf.keras.regularizers.l1(0.0001)),
  tf.keras.layers.Activation('relu'),

  tf.keras.layers.Dense(2, kernel_regularizer=tf.keras.regularizers.l1(0.001)),
  tf.keras.layers.Activation('relu'),

  tf.keras.layers.Dense(87, kernel_regularizer=tf.keras.regularizers.l1(0.01)),
  tf.keras.layers.Activation('relu'),

  tf.keras.layers.Dense(8, kernel_regularizer=tf.keras.regularizers.l1(0.001)),
  tf.keras.layers.Activation('relu'),

  tf.keras.layers.Dense(num_of_classes),
  tf.keras.layers.Softmax()
  ]
```

הדיוק מעט גרוע יותר ועומד על:

Training accuracy (sparse_categorical_accuracy): 0.9385

Validation accuracy (val_sparse_categorical_accuracy): 0.9329

בניסויים הבאים אראה כיצד כמות מידע BATCH משפיעה על התוצאה

ניסוי מס' 8

בניסוי זה אראה כיצד כמות דחיפת הנתונים באימון משפיעה על התוצאה (בהתבסס על הפרמטרים שבוצעו בניסוי מס'7) :

היה: `ds_train = ds_train.batch(32)` (דיפולטיבי לכל הניסויים)

לניסוי זה: `ds_train = ds_train.batch(64)`

הגדלת כמות הנתונים באימון השפיעה אנושות על התוצאות:

Training accuracy (sparse_categorical_accuracy): 0.570

Validation accuracy (val_sparse_categorical_accuracy): 0.561

בניסוי הבא אנסה להוריד את כמות הנתונים בכל שלב.

<u>ניסוי מס' 9</u>

ניסוי זה מתבסס על הפרמטרים מניסוי מס' 7 רק אוריד את כמות הנתונים:

היה: `ds_train = ds_train.batch(64)`

לניסוי זה: `ds_train = ds_train.batch(16)`

Training accuracy (sparse_categorical_accuracy): 0.9278

Validation accuracy (val_sparse_categorical_accuracy): 0.9225

<u>ניתן לראות שקצב הנתונים באימון משפיע דרסטית. כאשר הBATCH עמד על 32 התוצאה הייתה המדויקת ביותר לכן אצמד אליה.</u>

ניסוי מס' 10

שימוש ב-4 שכבות, רגולריזציה מסוג l2 ונורמליזציה עם הפעלה מסוג RELU:

```
layers = [
  tf.keras.layers.Flatten(input_shape=image_shape),

  tf.keras.layers.Dense(46, kernel_regularizer=tf.keras.regularizers.l2(0.0001)),
  tf.keras.layers.BatchNormalization(),
  tf.keras.layers.Activation('relu'),


  tf.keras.layers.Dense(2, kernel_regularizer=tf.keras.regularizers.l2(0.001)),
  tf.keras.layers.BatchNormalization(),
  tf.keras.layers.Activation('relu'),

  tf.keras.layers.Dense(87, kernel_regularizer=tf.keras.regularizers.l2(0.01)),
  tf.keras.layers.BatchNormalization(),
  tf.keras.layers.Activation('relu'),

  tf.keras.layers.Dense(8, kernel_regularizer=tf.keras.regularizers.l2(0.001)),
  tf.keras.layers.BatchNormalization(),
  tf.keras.layers.Activation('relu'),

  tf.keras.layers.Dense(num_of_classes),
  tf.keras.layers.Softmax()
  ]
```

Training accuracy (sparse_categorical_accuracy): 0.9033

Validation accuracy (val_sparse_categorical_accuracy): 0.9253

ניסוי מס' 11

שימוש באותם פרמטרים כמו בניסוי 10 רק הפעלה בתצורת SIGMOID ולא RELU:

```
[121] layers = [
        tf.keras.layers.Flatten(input_shape=image_shape),

        tf.keras.layers.Dense(46, kernel_regularizer=tf.keras.regularizers.l2(0.0001)),
        tf.keras.layers.BatchNormalization(),
        tf.keras.layers.Activation('sigmoid'),


        tf.keras.layers.Dense(2, kernel_regularizer=tf.keras.regularizers.l2(0.001)),
        tf.keras.layers.BatchNormalization(),
        tf.keras.layers.Activation('sigmoid'),

        tf.keras.layers.Dense(87, kernel_regularizer=tf.keras.regularizers.l2(0.01)),
        tf.keras.layers.BatchNormalization(),
        tf.keras.layers.Activation('sigmoid'),

        tf.keras.layers.Dense(8, kernel_regularizer=tf.keras.regularizers.l2(0.001)),
        tf.keras.layers.BatchNormalization(),
        tf.keras.layers.Activation('sigmoid'),

        tf.keras.layers.Dense(num_of_classes),
        tf.keras.layers.Softmax()
        ]
```

תוצאות:

Training accuracy (sparse_categorical_accuracy): 0.8987

Validation accuracy (val_sparse_categorical_accuracy): 0.9113

שאר התוצאות היו משמעותית גרועות יותר:

```
val_sparse_categorical_accuracy: 0.5970
val_sparse_categorical_accuracy: 0.7084
val_sparse_categorical_accuracy: 0.7089
val_sparse_categorical_accuracy: 0.8628
val_sparse_categorical_accuracy: 0.9113
val_sparse_categorical_accuracy: 0.5865
val_sparse_categorical_accuracy: 0.8640
val_sparse_categorical_accuracy: 0.7278
val_sparse_categorical_accuracy: 0.6401
val_sparse_categorical_accuracy: 0.7298
```

ניתן להסיק הפעלת SIGMOID אינה משפרת את התוצאה לכן אשתמש ב-RELU.

ניסוי מס' 12

פרמטרים זהים לניסוי מס' 10 פרט לערכים ברגולריזציה שהספרה 1 שונתה לספרה 5:

```
layers = [
  tf.keras.layers.Flatten(input_shape=image_shape),

  tf.keras.layers.Dense(46, kernel_regularizer=tf.keras.regularizers.l2(0.0005)),
  tf.keras.layers.BatchNormalization(),
  tf.keras.layers.Activation('relu'),


  tf.keras.layers.Dense(2, kernel_regularizer=tf.keras.regularizers.l2(0.005)),
  tf.keras.layers.BatchNormalization(),
  tf.keras.layers.Activation('relu'),

  tf.keras.layers.Dense(87, kernel_regularizer=tf.keras.regularizers.l2(0.05)),
  tf.keras.layers.BatchNormalization(),
  tf.keras.layers.Activation('relu'),

  tf.keras.layers.Dense(8, kernel_regularizer=tf.keras.regularizers.l2(0.005)),
  tf.keras.layers.BatchNormalization(),
  tf.keras.layers.Activation('relu'),

  tf.keras.layers.Dense(num_of_classes),
  tf.keras.layers.Softmax()
  ]
```

Training accuracy (sparse_categorical_accuracy): 0.9029

Validation accuracy (val_sparse_categorical_accuracy): 0.9232

הדיוקים היו לא טובים, הרוב המוחלט סביב ה80%. אחזור לערכים המקוריים.

ניסוי מס' 13:

בהמשך לפרמטרים בניסוי 11, הוספת DROPOUT של 10% לכל שכבה:

```
tf.keras.layers.Flatten(input_shape=image_shape),

tf.keras.layers.Dense(46, kernel_regularizer=tf.keras.regularizers.l2(0.0001)),
tf.keras.layers.BatchNormalization(),
tf.keras.layers.Activation('relu'),
tf.keras.layers.Dropout(0.1),


tf.keras.layers.Dense(2, kernel_regularizer=tf.keras.regularizers.l2(0.001)),
tf.keras.layers.BatchNormalization(),
tf.keras.layers.Activation('relu'),
tf.keras.layers.Dropout(0.1),

tf.keras.layers.Dense(87, kernel_regularizer=tf.keras.regularizers.l2(0.01)),
tf.keras.layers.BatchNormalization(),
tf.keras.layers.Activation('relu'),
tf.keras.layers.Dropout(0.1),

tf.keras.layers.Dense(8, kernel_regularizer=tf.keras.regularizers.l2(0.001)),
tf.keras.layers.BatchNormalization(),
tf.keras.layers.Activation('relu'),
tf.keras.layers.Dropout(0.1),

tf.keras.layers.Dense(num_of_classes),
tf.keras.layers.Softmax()
```

הדיוקים לא טובים בלשון המעטה:

Training accuracy (sparse_categorical_accuracy): 0.6013

Validation accuracy (val_sparse_categorical_accuracy): 0.7456

<u>ניתן לראות ש-DROPOUT משפיע רבות על הדיוק בעיקר באימון כי רק בתהליך האימון הוא משפיע, בתהליך הפרדיקציה לא קיים DROPOUT.</u>

ניסוי מס' 14

בהמשך לניסוי מס' 13, שיניתי את ערכי ה-DROPOUT לראות אם יש שיפור

```
tf.keras.layers.Dense(46, kernel_regularizer=tf.keras.regularizers.l2(0.0001)),
tf.keras.layers.BatchNormalization(),
tf.keras.layers.Activation('relu'),
tf.keras.layers.Dropout(0.1),


tf.keras.layers.Dense(2, kernel_regularizer=tf.keras.regularizers.l2(0.001)),
tf.keras.layers.BatchNormalization(),
tf.keras.layers.Activation('relu'),
tf.keras.layers.Dropout(0.2),

tf.keras.layers.Dense(87, kernel_regularizer=tf.keras.regularizers.l2(0.01)),
tf.keras.layers.BatchNormalization(),
tf.keras.layers.Activation('relu'),
tf.keras.layers.Dropout(0.3),

tf.keras.layers.Dense(8, kernel_regularizer=tf.keras.regularizers.l2(0.001)),
tf.keras.layers.BatchNormalization(),
tf.keras.layers.Activation('relu'),
tf.keras.layers.Dropout(0.4),

tf.keras.layers.Dense(num_of_classes),
tf.keras.layers.Softmax()
```

Training accuracy (sparse_categorical_accuracy): 0.4519

Validation accuracy (val_sparse_categorical_accuracy): 0.6960

ניתן לראות כי ה-DROPOUT בקונסטלציה הנוכחית עם רגורלריזציה, נורמליזציה, אקטיבציה עם RELU ו-DROPOUT לא עובד יחדיו.

ניסוי מס' 15

המשך ישיר לניסוי 14 רק עם הפעלת SIGMOID ולא RELU:

```
tf.keras.layers.Flatten(input_shape=image_shape),

tf.keras.layers.Dense(46, kernel_regularizer=tf.keras.regularizers.l2(0.0001)),
tf.keras.layers.BatchNormalization(),
tf.keras.layers.Activation('sigmoid'),
tf.keras.layers.Dropout(0.1),


tf.keras.layers.Dense(2, kernel_regularizer=tf.keras.regularizers.l2(0.001)),
tf.keras.layers.BatchNormalization(),
tf.keras.layers.Activation('sigmoid'),
tf.keras.layers.Dropout(0.2),

tf.keras.layers.Dense(87, kernel_regularizer=tf.keras.regularizers.l2(0.01)),
tf.keras.layers.BatchNormalization(),
tf.keras.layers.Activation('sigmoid'),
tf.keras.layers.Dropout(0.3),

tf.keras.layers.Dense(8, kernel_regularizer=tf.keras.regularizers.l2(0.001)),
tf.keras.layers.BatchNormalization(),
tf.keras.layers.Activation('sigmoid'),
tf.keras.layers.Dropout(0.4),

tf.keras.layers.Dense(num_of_classes),
tf.keras.layers.Softmax()
```

Training accuracy (sparse_categorical_accuracy): 0.4895

Validation accuracy (val_sparse_categorical_accuracy): 0.8216

ניסוי מס' 16

הורדת BATCHNORMALIZATION:

```
tf.keras.layers.Flatten(input_shape=image_shape),

tf.keras.layers.Dense(46, kernel_regularizer=tf.keras.regularizers.l2(0.0001)),
tf.keras.layers.Activation('sigmoid'),
tf.keras.layers.Dropout(0.1),


tf.keras.layers.Dense(2, kernel_regularizer=tf.keras.regularizers.l2(0.001)),
tf.keras.layers.Activation('sigmoid'),
tf.keras.layers.Dropout(0.2),

tf.keras.layers.Dense(87, kernel_regularizer=tf.keras.regularizers.l2(0.01)),
tf.keras.layers.Activation('sigmoid'),
tf.keras.layers.Dropout(0.3),

tf.keras.layers.Dense(8, kernel_regularizer=tf.keras.regularizers.l2(0.001)),
tf.keras.layers.Activation('sigmoid'),
tf.keras.layers.Dropout(0.4),

tf.keras.layers.Dense(num_of_classes),
tf.keras.layers.Softmax()
]
```

התקבלו תוצאות כמעט זהות לניסוי מס' 4:

Validation accuracy (val_sparse_categorical_accuracy): 0.1028

Validation accuracy (val_sparse_categorical_accuracy): 0.1135

כאשר דיוק הלמידה עמד בטווח:

Training accuracy (sparse_categorical_accuracy): 0.1050-0.1144

לא הייתה למידה, כנראה המשקלים שנבחרו בהתחלה לא שונו והתקבלה כל פעם אותה תוצאה.

ניסוי מס' 17

המשך לניסוי 16 רק החלפת SIGMOID ב-RELU

```
tf.keras.layers.Flatten(input_shape=image_shape),

tf.keras.layers.Dense(46, kernel_regularizer=tf.keras.regularizers.l2(0.0001)),
tf.keras.layers.Activation('relu'),
tf.keras.layers.Dropout(0.1),


tf.keras.layers.Dense(2, kernel_regularizer=tf.keras.regularizers.l2(0.001)),
tf.keras.layers.Activation('relu'),
tf.keras.layers.Dropout(0.2),

tf.keras.layers.Dense(87, kernel_regularizer=tf.keras.regularizers.l2(0.01)),
tf.keras.layers.Activation('relu'),
tf.keras.layers.Dropout(0.3),

tf.keras.layers.Dense(8, kernel_regularizer=tf.keras.regularizers.l2(0.001)),
tf.keras.layers.Activation('relu'),
tf.keras.layers.Dropout(0.4),

tf.keras.layers.Dense(num_of_classes),
tf.keras.layers.Softmax()
]
```

Validation accuracy (val_sparse_categorical_accuracy): 0.7906

Training accuracy (sparse_categorical_accuracy): 0.5699

דיוק הולידציה גבוהה יותר מהאימון, מצביע על UNDERFITTING כנראה.

ניסוי מס' 18

שינוי ערכי DROPOUT לזהה בכולם ושווה ל0.1:

```
tf.keras.layers.Flatten(input_shape=image_shape),

tf.keras.layers.Dense(46, kernel_regularizer=tf.keras.regularizers.l2(0.0001)),
tf.keras.layers.Activation('relu'),
tf.keras.layers.Dropout(0.1),


tf.keras.layers.Dense(2, kernel_regularizer=tf.keras.regularizers.l2(0.001)),
tf.keras.layers.Activation('relu'),
tf.keras.layers.Dropout(0.1),

tf.keras.layers.Dense(87, kernel_regularizer=tf.keras.regularizers.l2(0.01)),
tf.keras.layers.Activation('relu'),
tf.keras.layers.Dropout(0.1),

tf.keras.layers.Dense(8, kernel_regularizer=tf.keras.regularizers.l2(0.001)),
tf.keras.layers.Activation('relu'),
tf.keras.layers.Dropout(0.1),

tf.keras.layers.Dense(num_of_classes),
tf.keras.layers.Softmax()
]
```

Validation accuracy (val_sparse_categorical_accuracy): 0.8252

Training accuracy (sparse_categorical_accuracy): 0.7215

גם כאן ניתן להבחין ב UNDERFITTING.

ניסוי מס' 19

הורדת רגולריזציה ו-DROPOUT והוספה BATCHNORMALIZATION

```
tf.keras.layers.Flatten(input_shape=image_shape),

tf.keras.layers.Dense(46),
tf.keras.layers.BatchNormalization(),
tf.keras.layers.Activation('relu'),


tf.keras.layers.Dense(2),
tf.keras.layers.BatchNormalization(),
tf.keras.layers.Activation('relu'),


tf.keras.layers.Dense(87),
tf.keras.layers.BatchNormalization(),
tf.keras.layers.Activation('relu'),


tf.keras.layers.Dense(8),
tf.keras.layers.BatchNormalization(),
tf.keras.layers.Activation('relu'),


tf.keras.layers.Dense(num_of_classes),
tf.keras.layers.Softmax()
]
```

Validation accuracy (val_sparse_categorical_accuracy): 0.9349

Training accuracy (sparse_categorical_accuracy): 0.9283

ניסוי מס' 20

המשך לניסוי 19, החלפה בין RULE ל-SIGMOID:

```
tf.keras.layers.Flatten(input_shape=image_shape),

tf.keras.layers.Dense(46),
tf.keras.layers.BatchNormalization(),
tf.keras.layers.Activation('sigmoid'),


tf.keras.layers.Dense(2),
tf.keras.layers.BatchNormalization(),
tf.keras.layers.Activation('sigmoid'),


tf.keras.layers.Dense(87),
tf.keras.layers.BatchNormalization(),
tf.keras.layers.Activation('sigmoid'),


tf.keras.layers.Dense(8),
tf.keras.layers.BatchNormalization(),
tf.keras.layers.Activation('sigmoid'),


tf.keras.layers.Dense(num_of_classes),
tf.keras.layers.Softmax()
]
```

Validation accuracy (val_sparse_categorical_accuracy): 0.9334

Training accuracy (sparse_categorical_accuracy): 0.9296

מסקנה

התוצאות הטובות ביותר התקבלו עם הפרמטרים "הפשוטים" ביותר.

הרצת בונוס

ללא קשר למטלה שבה לא ניתן לשנות בין סדר השכבות, רציתי לראות כיצד הסדר בין השכבות משפיע על דיוק המודל

```
tf.keras.layers.Flatten(input_shape=image_shape),

tf.keras.layers.Dense(46),
tf.keras.layers.BatchNormalization(),
tf.keras.layers.Activation('sigmoid'),


 tf.keras.layers.Dense(87),
tf.keras.layers.BatchNormalization(),
tf.keras.layers.Activation('sigmoid'),

tf.keras.layers.Dense(2),
tf.keras.layers.BatchNormalization(),
tf.keras.layers.Activation('sigmoid'),


#tf.keras.layers.Dense(87),
#tf.keras.layers.BatchNormalization(),
#tf.keras.layers.Activation('sigmoid'),


tf.keras.layers.Dense(8),
tf.keras.layers.BatchNormalization(),
tf.keras.layers.Activation('sigmoid'),


tf.keras.layers.Dense(num_of_classes),
tf.keras.layers.Softmax()
]
```

סידרתי בין השכבות כך שהשכבות עם מספר עשרוני של נוירונים יהיה בחלק העליון אחד אחרי השני והשכבות עם מספר נוירונים חד ספרתי אחריהם.

Validation accuracy (val_sparse_categorical_accuracy): 0.9504

Training accuracy (sparse_categorical_accuracy): 0.9302

ניתן לראות בבירור ששינוי סדר השכבות תורם לדיוק המודל. הרצה זו ללמידה אישית בלבד ולא כלולה כחלק מהמטלה.

ניסוי מס'5 הציג את התוצאות הטובות ביותר, בהתבסס על דיוק ולידציה ולא דיוק למידה:

```
layers = [
  tf.keras.layers.Flatten(input_shape=image_shape),

  tf.keras.layers.Dense(46, kernel_regularizer=tf.keras.regularizers.l2(0.0001)),
  tf.keras.layers.Activation('relu'),

  tf.keras.layers.Dense(2, kernel_regularizer=tf.keras.regularizers.l2(0.001)),
  tf.keras.layers.Activation('relu'),

  tf.keras.layers.Dense(87, kernel_regularizer=tf.keras.regularizers.l2(0.01)),
  tf.keras.layers.Activation('relu'),

  tf.keras.layers.Dense(8, kernel_regularizer=tf.keras.regularizers.l2(0.1)),
  tf.keras.layers.Activation('relu'),

  tf.keras.layers.Dense(num_of_classes),
  tf.keras.layers.Softmax()
  ]
```

חלק זה יישאר בקוד הסופי.