מטלה 2 למידת מכונה

מגיש: בר שני תז: 206812042

ניסוי 1:

תוצאה:

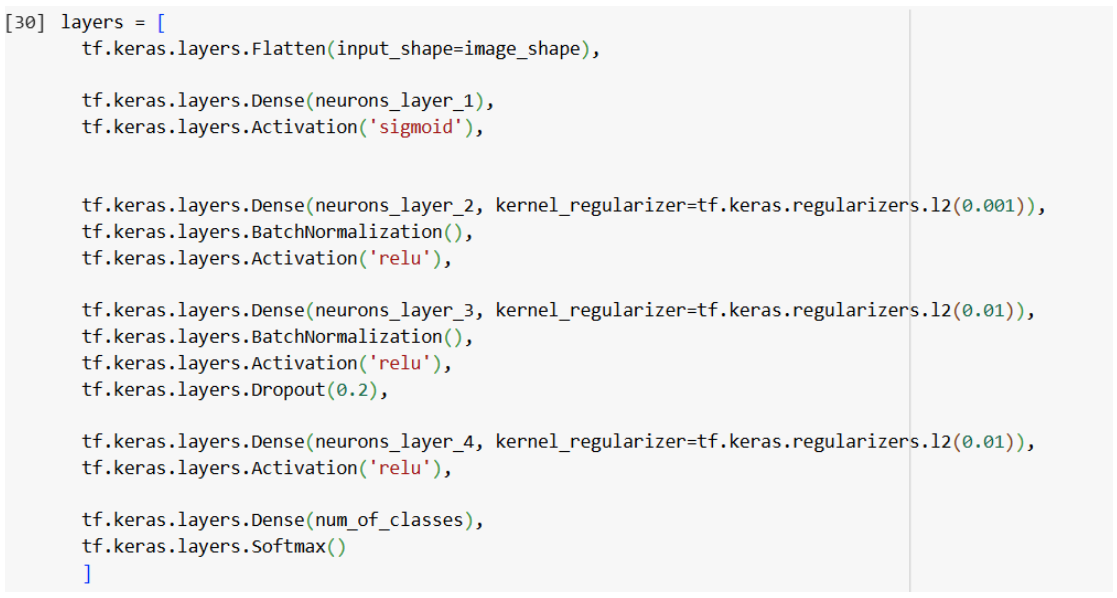
loss: 0.1216

sparse_categorical_accuracy: 0.9726

val_loss: 0.1514

val_sparse_categorical_accuracy: 0.9702

שכבות:



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

         tf.keras.layers.Dense(neurons_layer_1),
         tf.keras.layers.Activation('sigmoid'),


         tf.keras.layers.Dense(neurons_layer_2, kernel_regularizer=tf.keras.regularizers.l2(0.001)),
         tf.keras.layers.BatchNormalization(),
         tf.keras.layers.Activation('relu'),

         tf.keras.layers.Dense(neurons_layer_3, kernel_regularizer=tf.keras.regularizers.l2(0.01)),
         tf.keras.layers.BatchNormalization(),
         tf.keras.layers.Activation('relu'),
         tf.keras.layers.Dropout(0.2),

         tf.keras.layers.Dense(neurons_layer_4, kernel_regularizer=tf.keras.regularizers.l2(0.01)),
         tf.keras.layers.Activation('relu'),

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

ניסוי 2:

תוצאה:

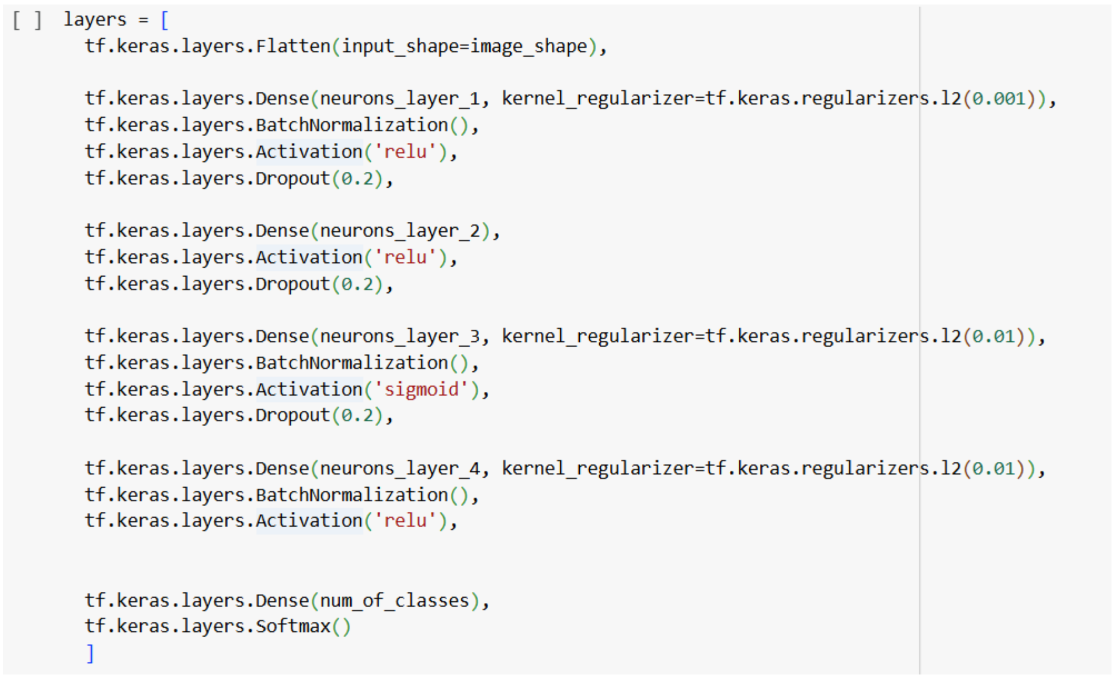
loss: 0.4115

sparse_categorical_accuracy: 0.9012

val_loss: 0.2120

val_sparse_categorical_accuracy: 0.9586

שכבות:



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

       tf.keras.layers.Dense(neurons_layer_1, 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(neurons_layer_2),
       tf.keras.layers.Activation('relu'),
       tf.keras.layers.Dropout(0.2),

       tf.keras.layers.Dense(neurons_layer_3, kernel_regularizer=tf.keras.regularizers.l2(0.01)),
       tf.keras.layers.BatchNormalization(),
       tf.keras.layers.Activation('sigmoid'),
       tf.keras.layers.Dropout(0.2),

       tf.keras.layers.Dense(neurons_layer_4, kernel_regularizer=tf.keras.regularizers.l2(0.01)),
       tf.keras.layers.BatchNormalization(),
       tf.keras.layers.Activation('relu'),


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

ניסוי 3:

תוצאה:

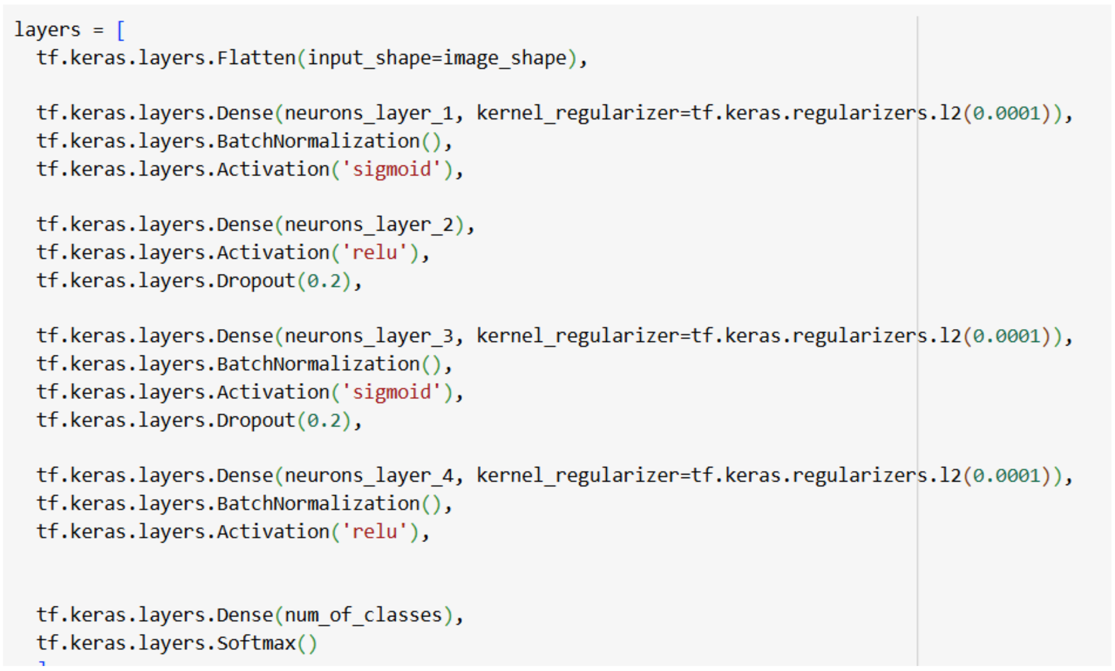
loss: 0.2421

sparse_categorical_accuracy: 0.9379

val_loss: 0.1574

val_sparse_categorical_accuracy: 0.9626

שכבות:



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

  tf.keras.layers.Dense(neurons_layer_1, kernel_regularizer=tf.keras.regularizers.l2(0.0001)),
  tf.keras.layers.BatchNormalization(),
  tf.keras.layers.Activation('sigmoid'),

  tf.keras.layers.Dense(neurons_layer_2),
  tf.keras.layers.Activation('relu'),
  tf.keras.layers.Dropout(0.2),

  tf.keras.layers.Dense(neurons_layer_3, kernel_regularizer=tf.keras.regularizers.l2(0.0001)),
  tf.keras.layers.BatchNormalization(),
  tf.keras.layers.Activation('sigmoid'),
  tf.keras.layers.Dropout(0.2),

  tf.keras.layers.Dense(neurons_layer_4, kernel_regularizer=tf.keras.regularizers.l2(0.0001)),
  tf.keras.layers.BatchNormalization(),
  tf.keras.layers.Activation('relu'),


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

ניסוי 4:

תוצאה:

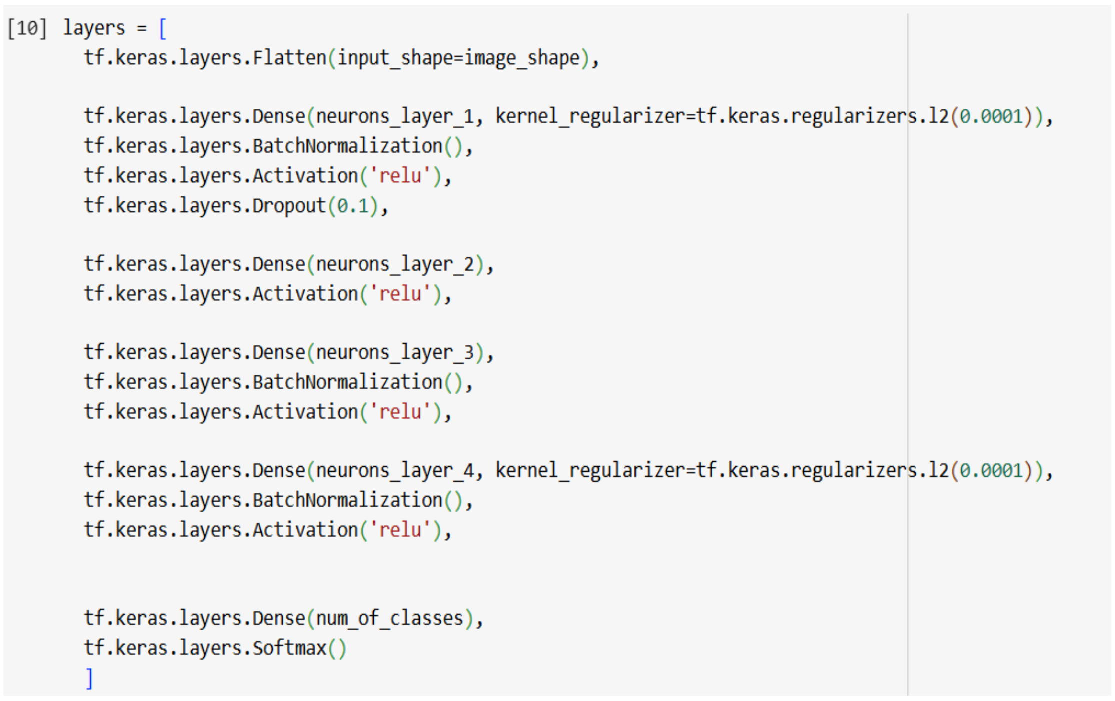
loss: 0.1644

sparse_categorical_accuracy: 0.9576

val_loss: 0.1122

val_sparse_categorical_accuracy: 0.9744

פרמטרים:



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

       tf.keras.layers.Dense(neurons_layer_1, 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(neurons_layer_2),
       tf.keras.layers.Activation('relu'),

       tf.keras.layers.Dense(neurons_layer_3),
       tf.keras.layers.BatchNormalization(),
       tf.keras.layers.Activation('relu'),

       tf.keras.layers.Dense(neurons_layer_4, kernel_regularizer=tf.keras.regularizers.l2(0.0001)),
       tf.keras.layers.BatchNormalization(),
       tf.keras.layers.Activation('relu'),


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

ניסוי 5:

תוצאה:

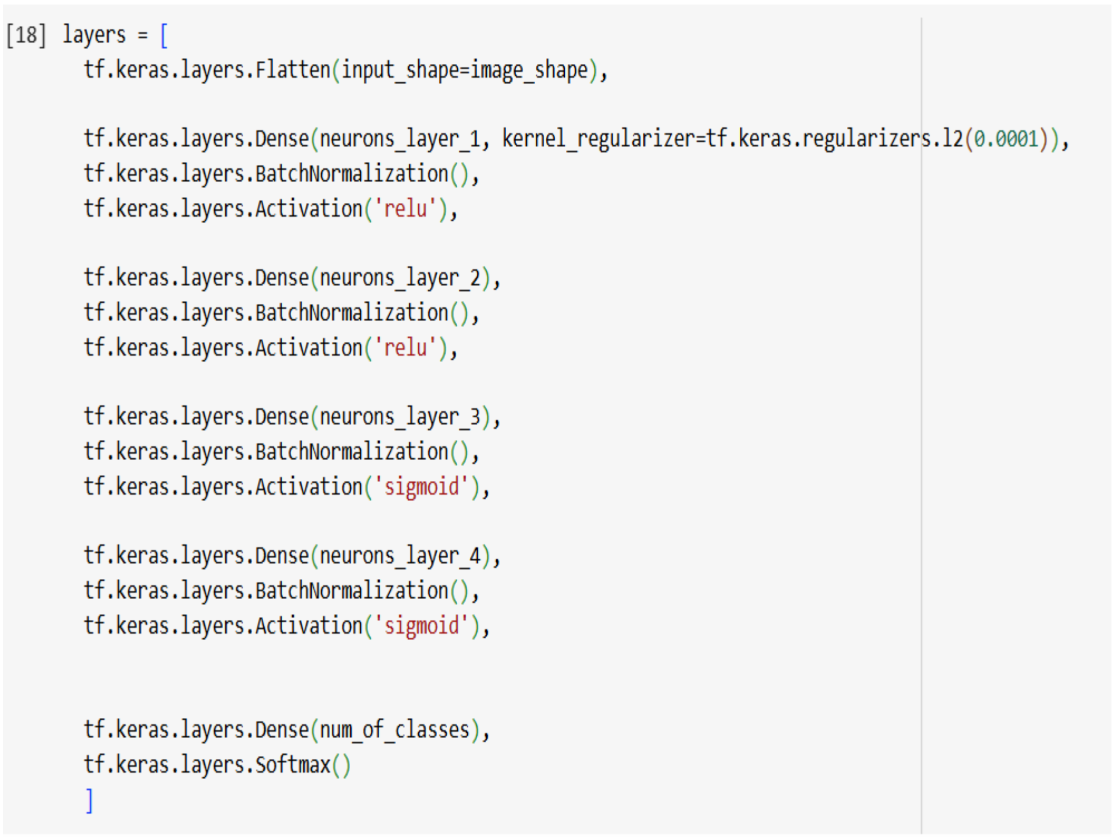
loss: 0.1124

sparse_categorical_accuracy: 0.9724

val_loss: 0.1025

val_sparse_categorical_accuracy: 0.9767

פרמטרים:



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

       tf.keras.layers.Dense(neurons_layer_1, kernel_regularizer=tf.keras.regularizers.l2(0.0001)),
       tf.keras.layers.BatchNormalization(),
       tf.keras.layers.Activation('relu'),

       tf.keras.layers.Dense(neurons_layer_2),
       tf.keras.layers.BatchNormalization(),
       tf.keras.layers.Activation('relu'),

       tf.keras.layers.Dense(neurons_layer_3),
       tf.keras.layers.BatchNormalization(),
       tf.keras.layers.Activation('sigmoid'),

       tf.keras.layers.Dense(neurons_layer_4),
       tf.keras.layers.BatchNormalization(),
       tf.keras.layers.Activation('sigmoid'),


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

ניסוי 6:

תוצאה:

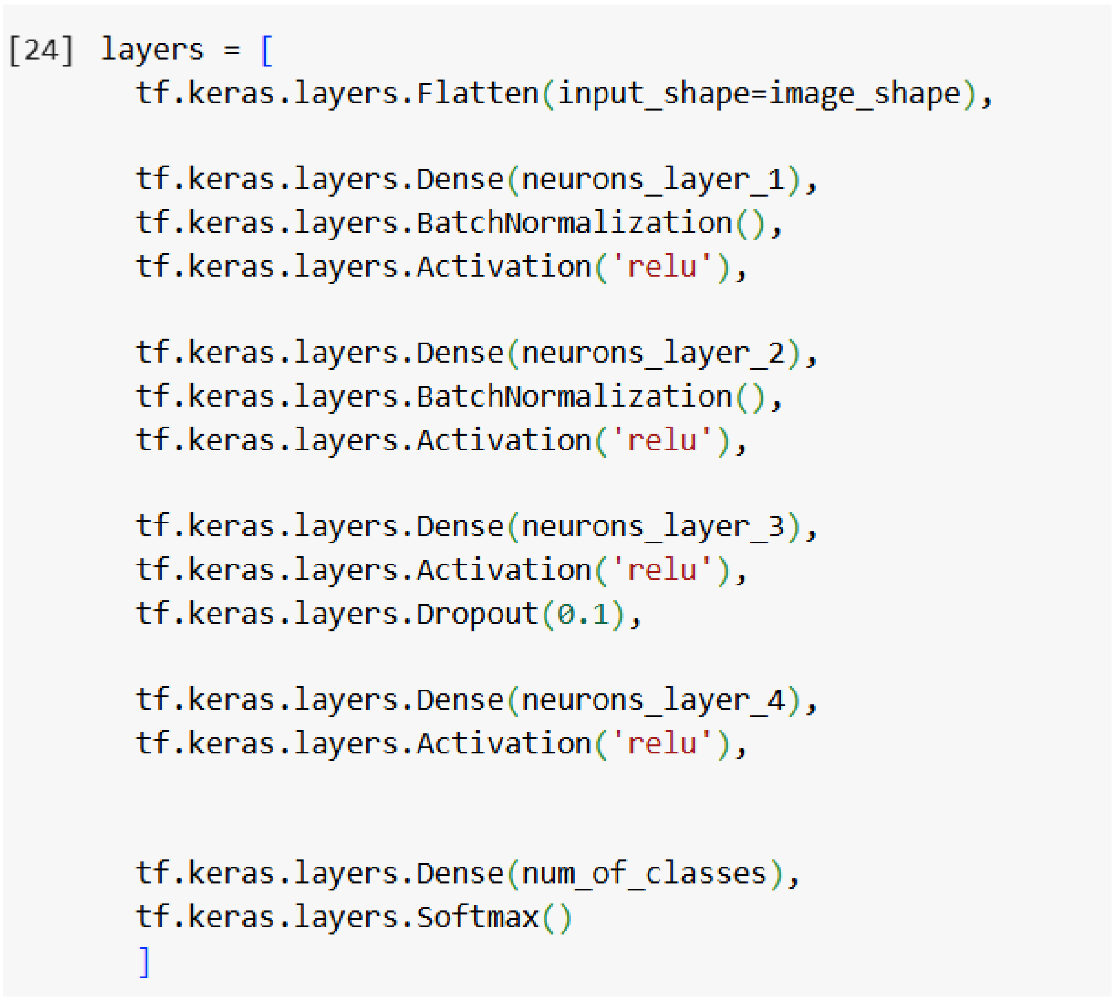
loss: 0.0712

sparse_categorical_accuracy: 0.9786

val_loss: 0.1087

val_sparse_categorical_accuracy: 0.9766

פרמטרים:



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

       tf.keras.layers.Dense(neurons_layer_1),
       tf.keras.layers.BatchNormalization(),
       tf.keras.layers.Activation('relu'),

       tf.keras.layers.Dense(neurons_layer_2),
       tf.keras.layers.BatchNormalization(),
       tf.keras.layers.Activation('relu'),

       tf.keras.layers.Dense(neurons_layer_3),
       tf.keras.layers.Activation('relu'),
       tf.keras.layers.Dropout(0.1),

       tf.keras.layers.Dense(neurons_layer_4),
       tf.keras.layers.Activation('relu'),


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

ניסוי 7:

תוצאה:

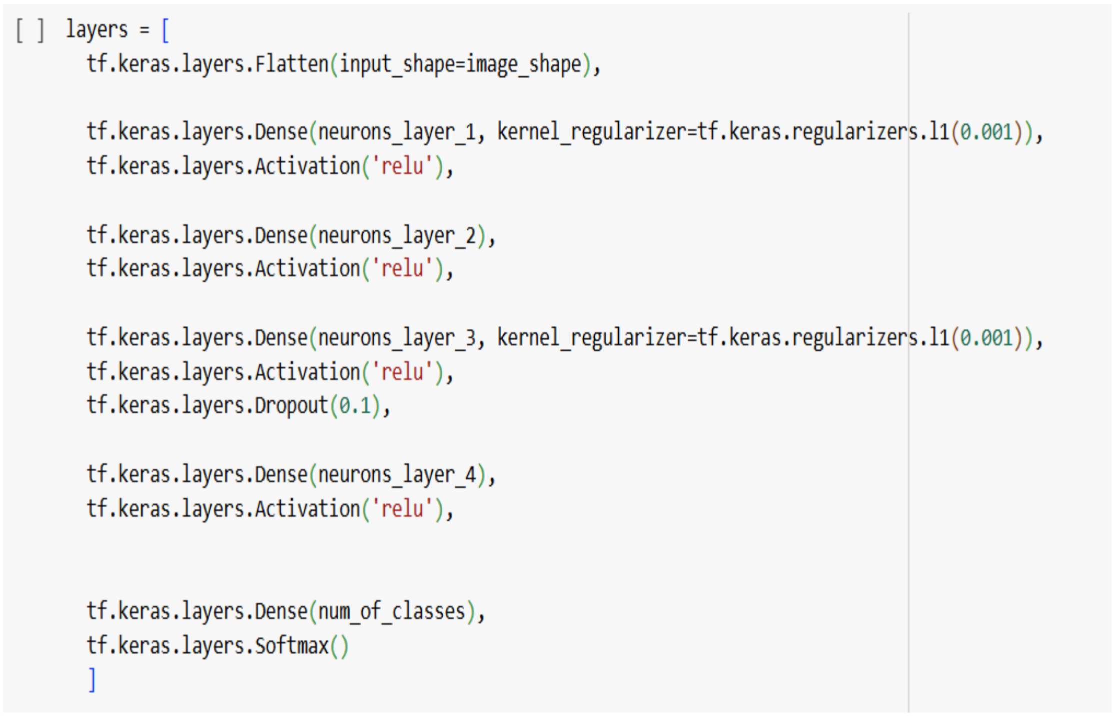
loss: 0.3487

sparse_categorical_accuracy: 0.9504

val_loss: 0.3386

val_sparse_categorical_accuracy: 0.9512

פרמטרים:



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

       tf.keras.layers.Dense(neurons_layer_1, kernel_regularizer=tf.keras.regularizers.l1(0.001)),
       tf.keras.layers.Activation('relu'),

       tf.keras.layers.Dense(neurons_layer_2),
       tf.keras.layers.Activation('relu'),

       tf.keras.layers.Dense(neurons_layer_3, kernel_regularizer=tf.keras.regularizers.l1(0.001)),
       tf.keras.layers.Activation('relu'),
       tf.keras.layers.Dropout(0.1),

       tf.keras.layers.Dense(neurons_layer_4),
       tf.keras.layers.Activation('relu'),


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

ניסוי 8:

תוצאה:

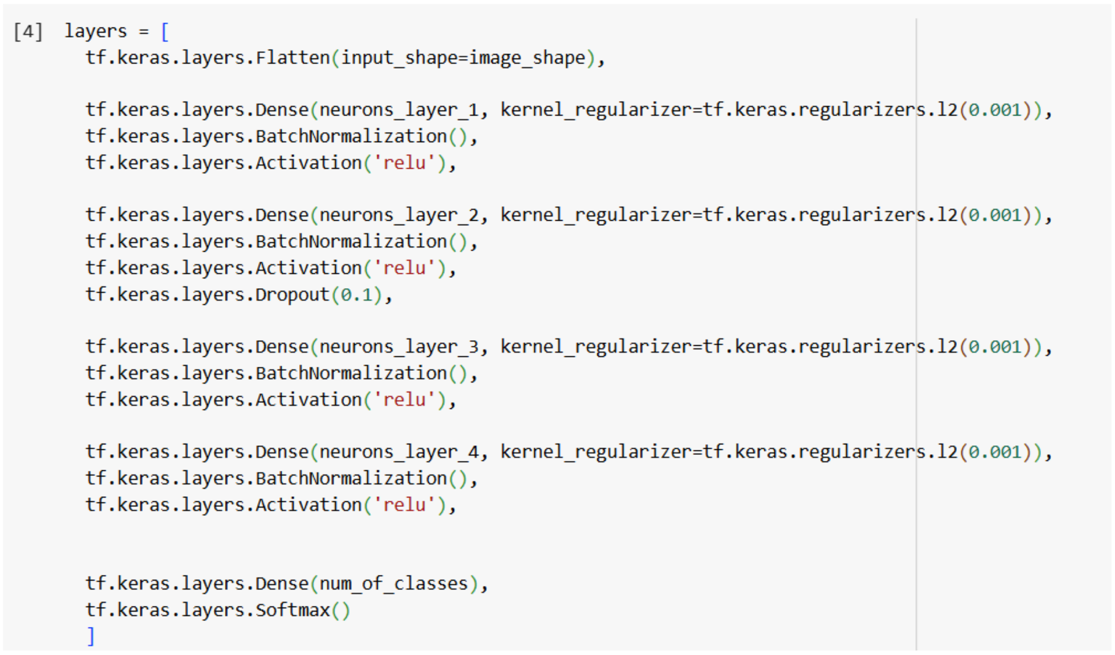
loss: 0.2152

sparse_categorical_accuracy: 0.9557

val_loss: 0.1651

val_sparse_categorical_accuracy: 0.9721

פרמטרים:



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

      tf.keras.layers.Dense(neurons_layer_1, kernel_regularizer=tf.keras.regularizers.l2(0.001)),
      tf.keras.layers.BatchNormalization(),
      tf.keras.layers.Activation('relu'),

      tf.keras.layers.Dense(neurons_layer_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(neurons_layer_3, kernel_regularizer=tf.keras.regularizers.l2(0.001)),
      tf.keras.layers.BatchNormalization(),
      tf.keras.layers.Activation('relu'),

      tf.keras.layers.Dense(neurons_layer_4, 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()
      ]
```

ניסוי 9:

תוצאה:

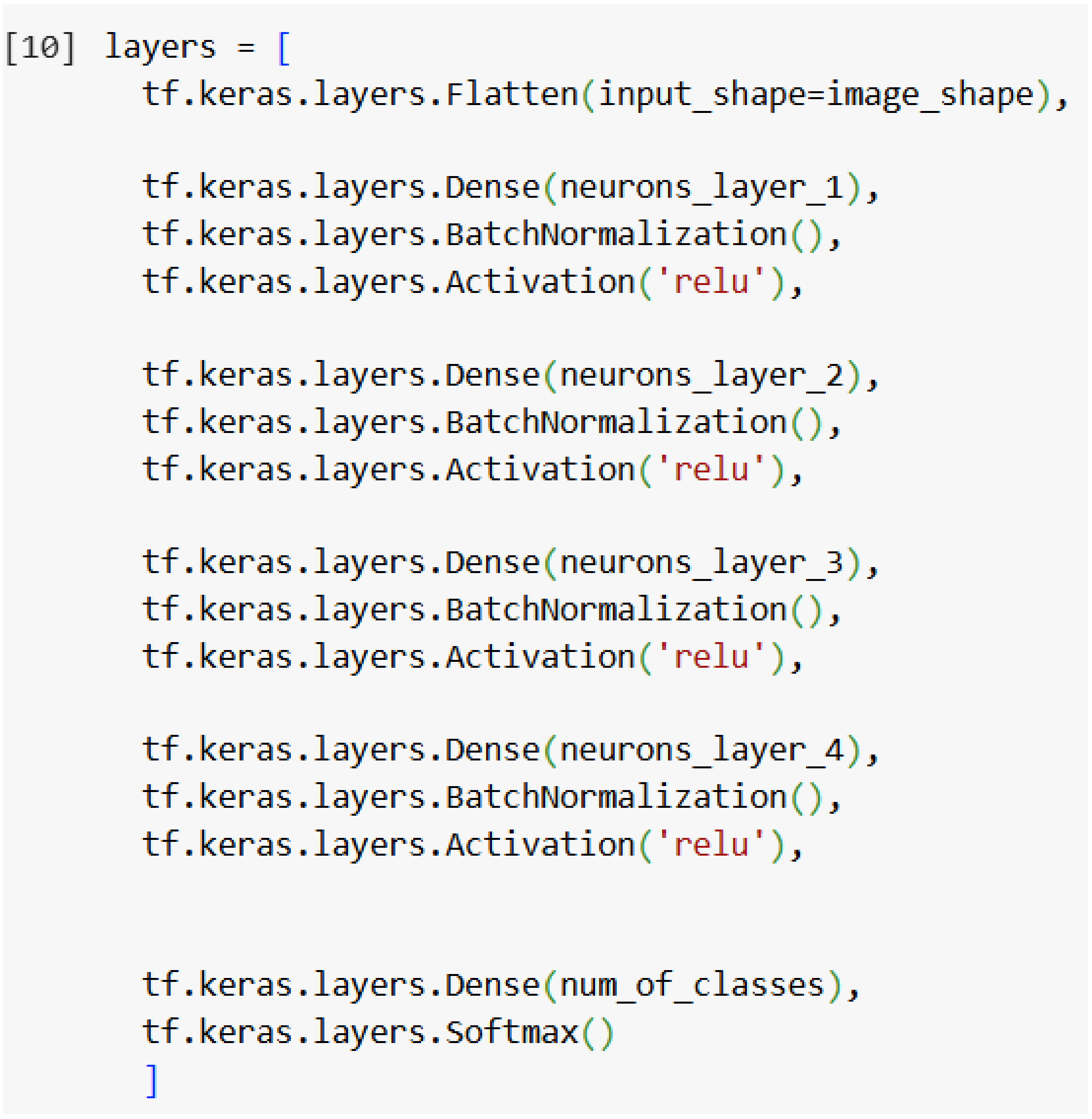
loss: 0.0581

sparse_categorical_accuracy: 0.9818

val_loss: 0.1010

val_sparse_categorical_accuracy: 0.9739

פרמטרים:



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

       tf.keras.layers.Dense(neurons_layer_1),
       tf.keras.layers.BatchNormalization(),
       tf.keras.layers.Activation('relu'),

       tf.keras.layers.Dense(neurons_layer_2),
       tf.keras.layers.BatchNormalization(),
       tf.keras.layers.Activation('relu'),

       tf.keras.layers.Dense(neurons_layer_3),
       tf.keras.layers.BatchNormalization(),
       tf.keras.layers.Activation('relu'),

       tf.keras.layers.Dense(neurons_layer_4),
       tf.keras.layers.BatchNormalization(),
       tf.keras.layers.Activation('relu'),


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

ניסוי 10:

תוצאה:

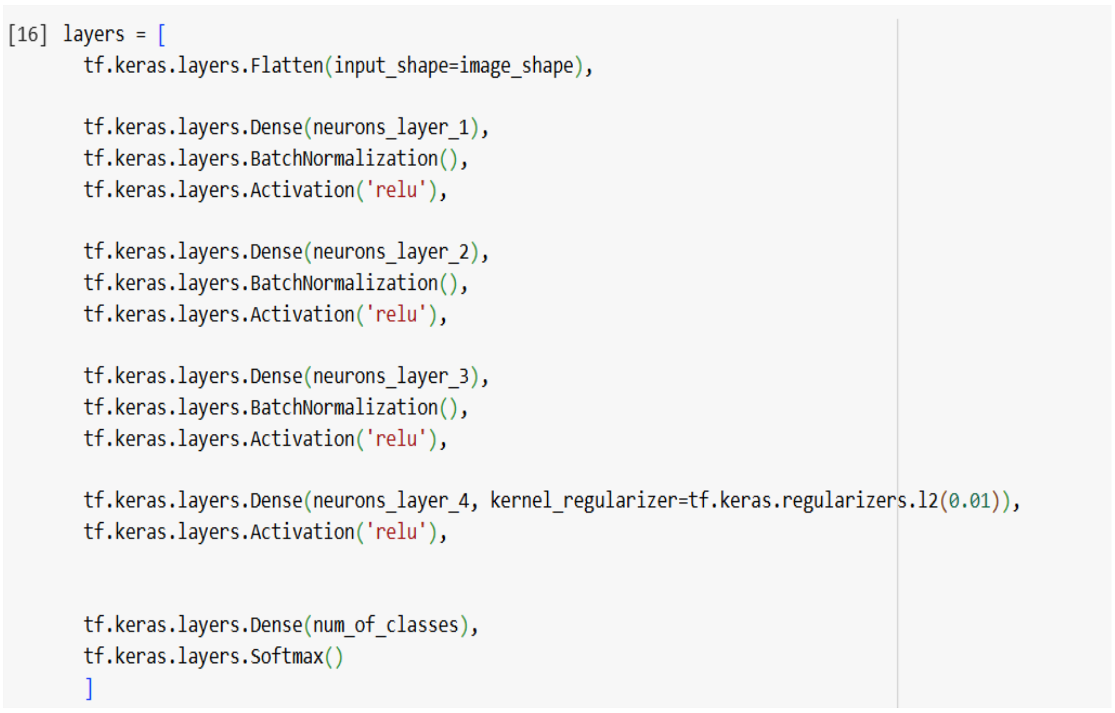
loss: 0.0715

sparse_categorical_accuracy: 0.9799

val_loss: 0.1037

val_sparse_categorical_accuracy: 0.9771

פרמטרים:



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

       tf.keras.layers.Dense(neurons_layer_1),
       tf.keras.layers.BatchNormalization(),
       tf.keras.layers.Activation('relu'),

       tf.keras.layers.Dense(neurons_layer_2),
       tf.keras.layers.BatchNormalization(),
       tf.keras.layers.Activation('relu'),

       tf.keras.layers.Dense(neurons_layer_3),
       tf.keras.layers.BatchNormalization(),
       tf.keras.layers.Activation('relu'),

       tf.keras.layers.Dense(neurons_layer_4, kernel_regularizer=tf.keras.regularizers.l2(0.01)),
       tf.keras.layers.Activation('relu'),


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

ניסוי 11:

תוצאה:

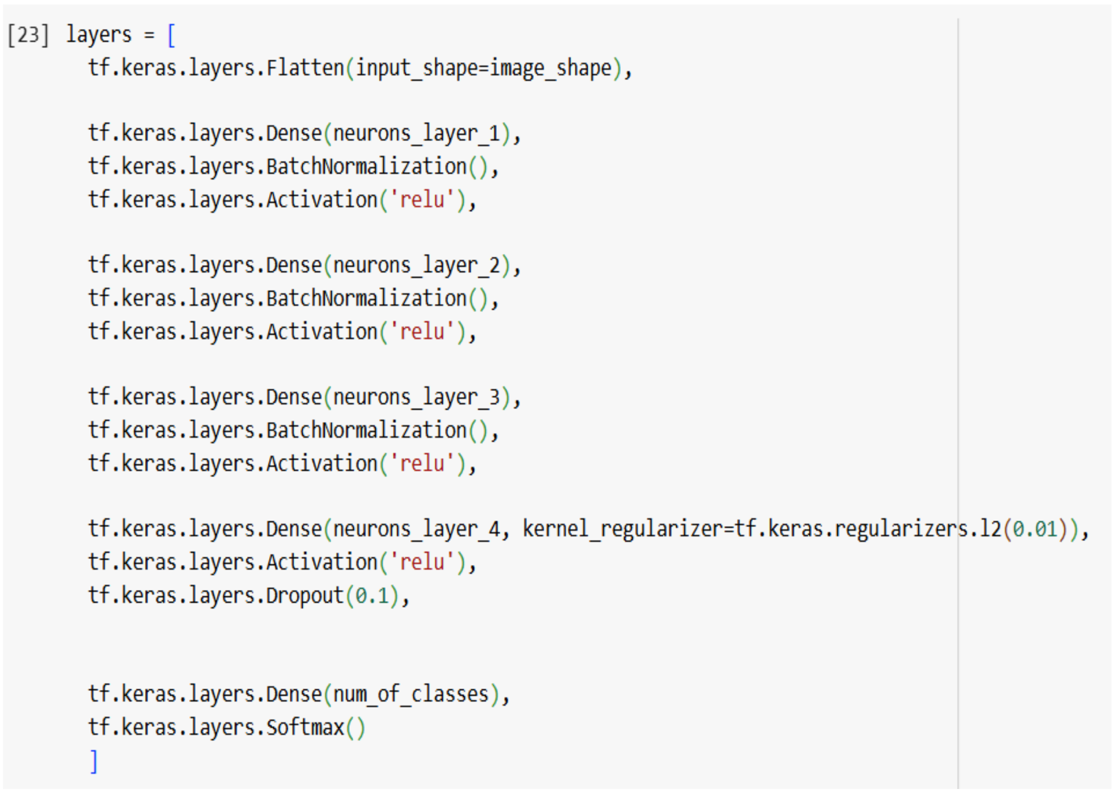
loss: 0.2339

sparse_categorical_accuracy: 0.9432

val_loss: 0.1396

val_sparse_categorical_accuracy: 0.9723

פרמטרים:



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

       tf.keras.layers.Dense(neurons_layer_1),
       tf.keras.layers.BatchNormalization(),
       tf.keras.layers.Activation('relu'),

       tf.keras.layers.Dense(neurons_layer_2),
       tf.keras.layers.BatchNormalization(),
       tf.keras.layers.Activation('relu'),

       tf.keras.layers.Dense(neurons_layer_3),
       tf.keras.layers.BatchNormalization(),
       tf.keras.layers.Activation('relu'),

       tf.keras.layers.Dense(neurons_layer_4, kernel_regularizer=tf.keras.regularizers.l2(0.01)),
       tf.keras.layers.Activation('relu'),
       tf.keras.layers.Dropout(0.1),


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

ניסוי 12:

תוצאה:

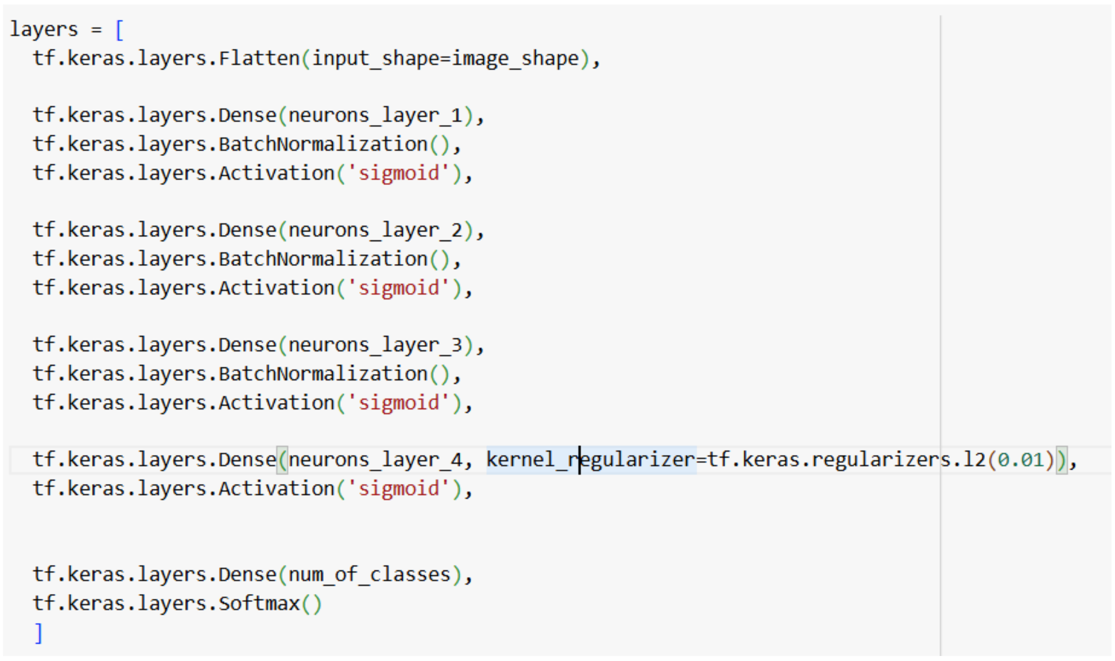
loss: 0.1096

sparse_categorical_accuracy: 0.9743

val_loss: 0.1232

val_sparse_categorical_accuracy: 0.9727

פרמטרים:



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

  tf.keras.layers.Dense(neurons_layer_1),
  tf.keras.layers.BatchNormalization(),
  tf.keras.layers.Activation('sigmoid'),

  tf.keras.layers.Dense(neurons_layer_2),
  tf.keras.layers.BatchNormalization(),
  tf.keras.layers.Activation('sigmoid'),

  tf.keras.layers.Dense(neurons_layer_3),
  tf.keras.layers.BatchNormalization(),
  tf.keras.layers.Activation('sigmoid'),

  tf.keras.layers.Dense(neurons_layer_4, kernel_regularizer=tf.keras.regularizers.l2(0.01)),
  tf.keras.layers.Activation('sigmoid'),


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

ניסוי 13:

תוצאה:

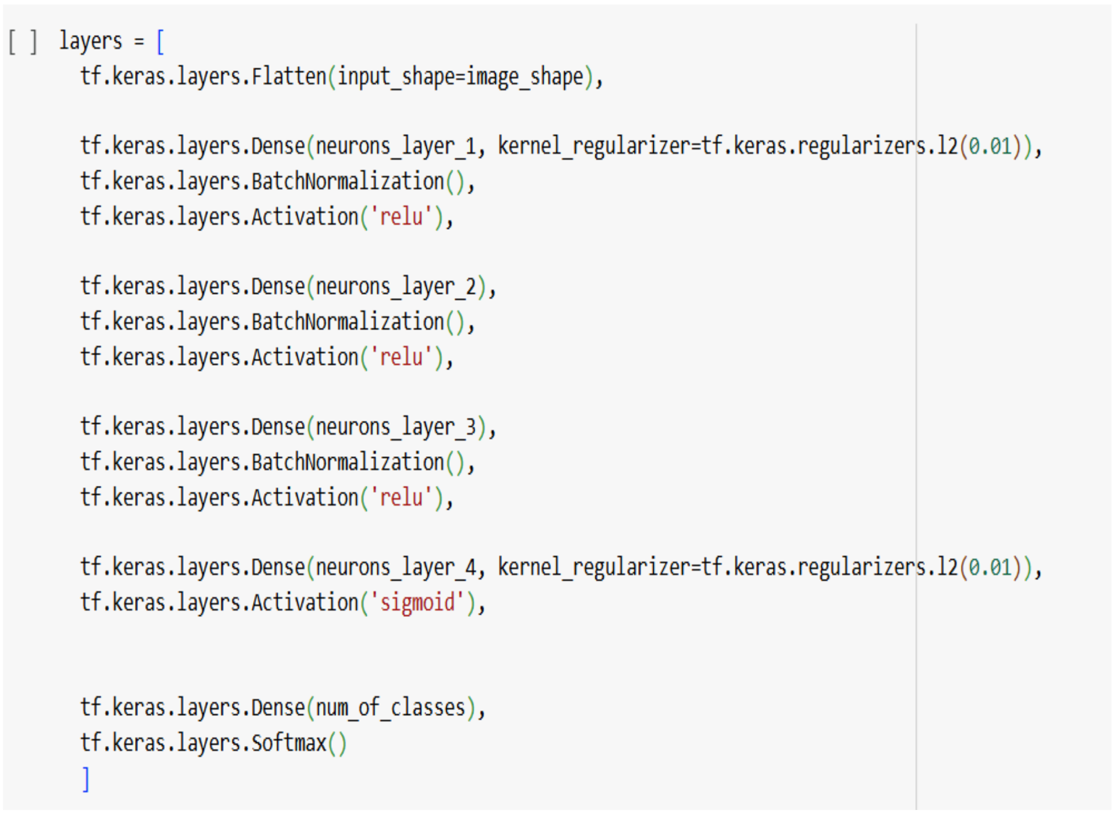
loss: 0.2683

sparse_categorical_accuracy: 0.9473

val_loss: 0.2262

val_sparse_categorical_accuracy: 0.9606

פרמטרים:



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

       tf.keras.layers.Dense(neurons_layer_1, kernel_regularizer=tf.keras.regularizers.l2(0.01)),
       tf.keras.layers.BatchNormalization(),
       tf.keras.layers.Activation('relu'),

       tf.keras.layers.Dense(neurons_layer_2),
       tf.keras.layers.BatchNormalization(),
       tf.keras.layers.Activation('relu'),

       tf.keras.layers.Dense(neurons_layer_3),
       tf.keras.layers.BatchNormalization(),
       tf.keras.layers.Activation('relu'),

       tf.keras.layers.Dense(neurons_layer_4, kernel_regularizer=tf.keras.regularizers.l2(0.01)),
       tf.keras.layers.Activation('sigmoid'),


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

ניסוי 14:

תוצאה:

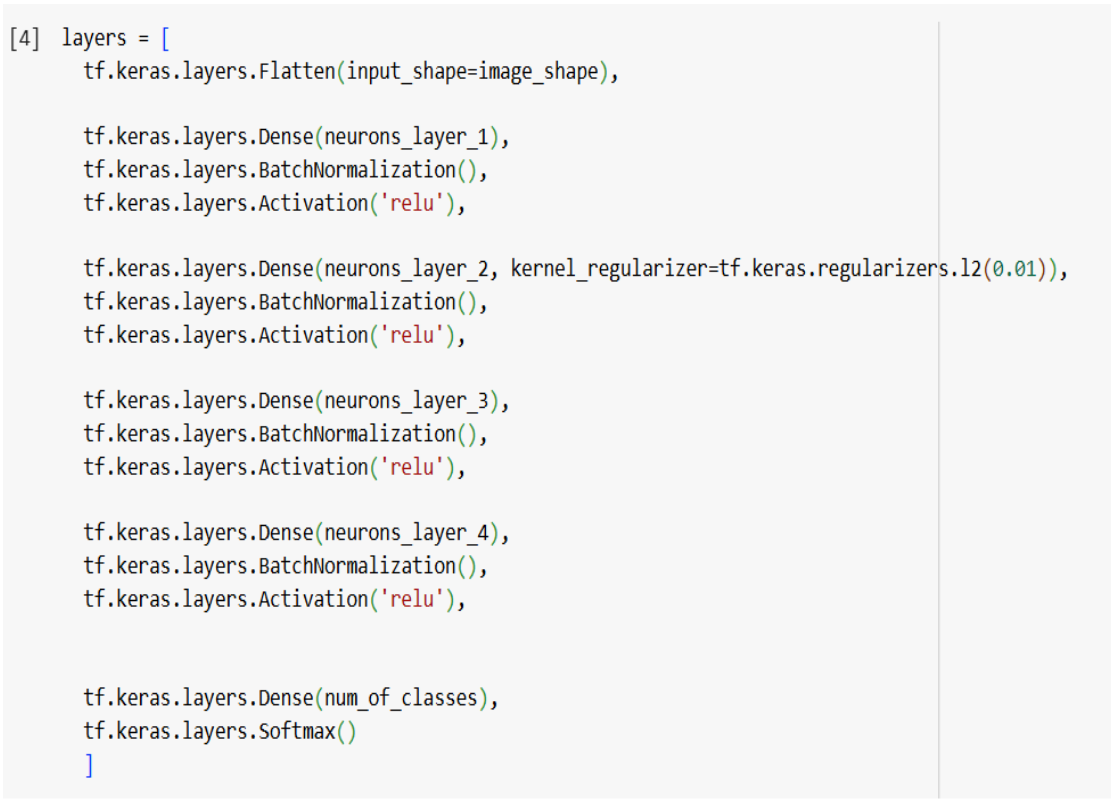
loss: 0.0949

sparse_categorical_accuracy: 0.9760

val_loss: 0.1233

val_sparse_categorical_accuracy: 0.9726

פרמטרים:



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

       tf.keras.layers.Dense(neurons_layer_1),
       tf.keras.layers.BatchNormalization(),
       tf.keras.layers.Activation('relu'),

       tf.keras.layers.Dense(neurons_layer_2, kernel_regularizer=tf.keras.regularizers.l2(0.01)),
       tf.keras.layers.BatchNormalization(),
       tf.keras.layers.Activation('relu'),

       tf.keras.layers.Dense(neurons_layer_3),
       tf.keras.layers.BatchNormalization(),
       tf.keras.layers.Activation('relu'),

       tf.keras.layers.Dense(neurons_layer_4),
       tf.keras.layers.BatchNormalization(),
       tf.keras.layers.Activation('relu'),


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

ניסוי 15:

תוצאה:

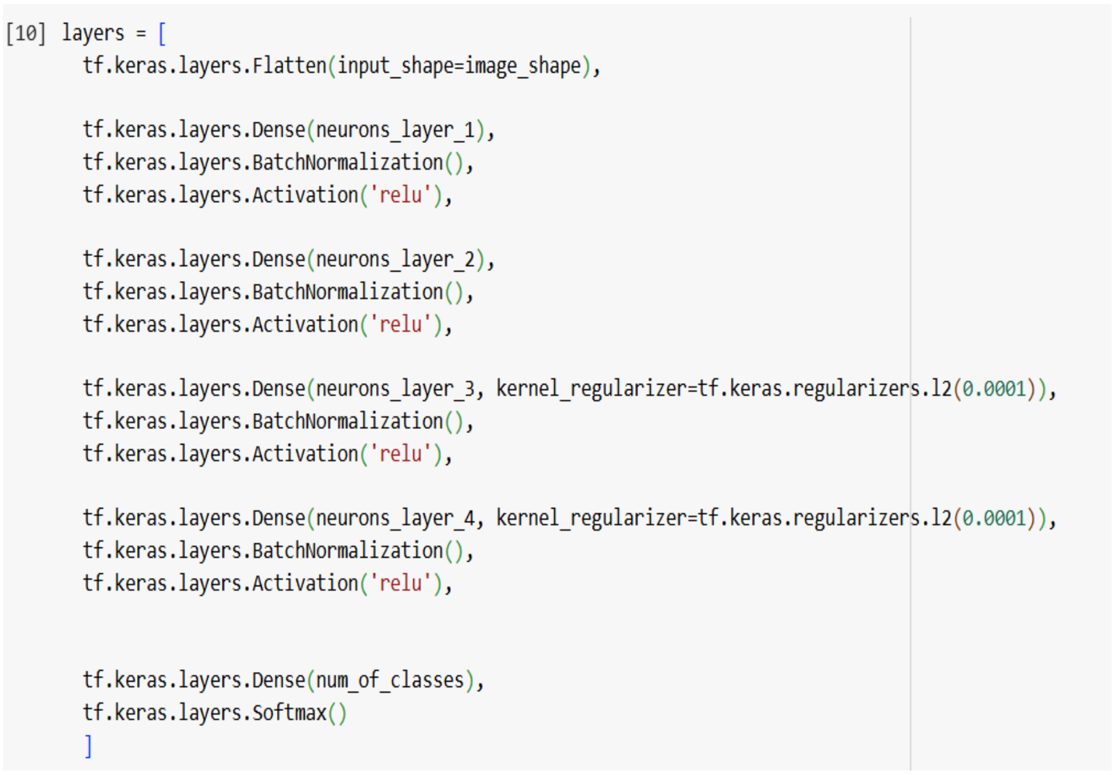
loss: 0.0722

sparse_categorical_accuracy: 0.9804

val_loss: 0.1103

val_sparse_categorical_accuracy: 0.9762

פרמטרים:



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

       tf.keras.layers.Dense(neurons_layer_1),
       tf.keras.layers.BatchNormalization(),
       tf.keras.layers.Activation('relu'),

       tf.keras.layers.Dense(neurons_layer_2),
       tf.keras.layers.BatchNormalization(),
       tf.keras.layers.Activation('relu'),

       tf.keras.layers.Dense(neurons_layer_3, kernel_regularizer=tf.keras.regularizers.l2(0.0001)),
       tf.keras.layers.BatchNormalization(),
       tf.keras.layers.Activation('relu'),

       tf.keras.layers.Dense(neurons_layer_4, kernel_regularizer=tf.keras.regularizers.l2(0.0001)),
       tf.keras.layers.BatchNormalization(),
       tf.keras.layers.Activation('relu'),


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

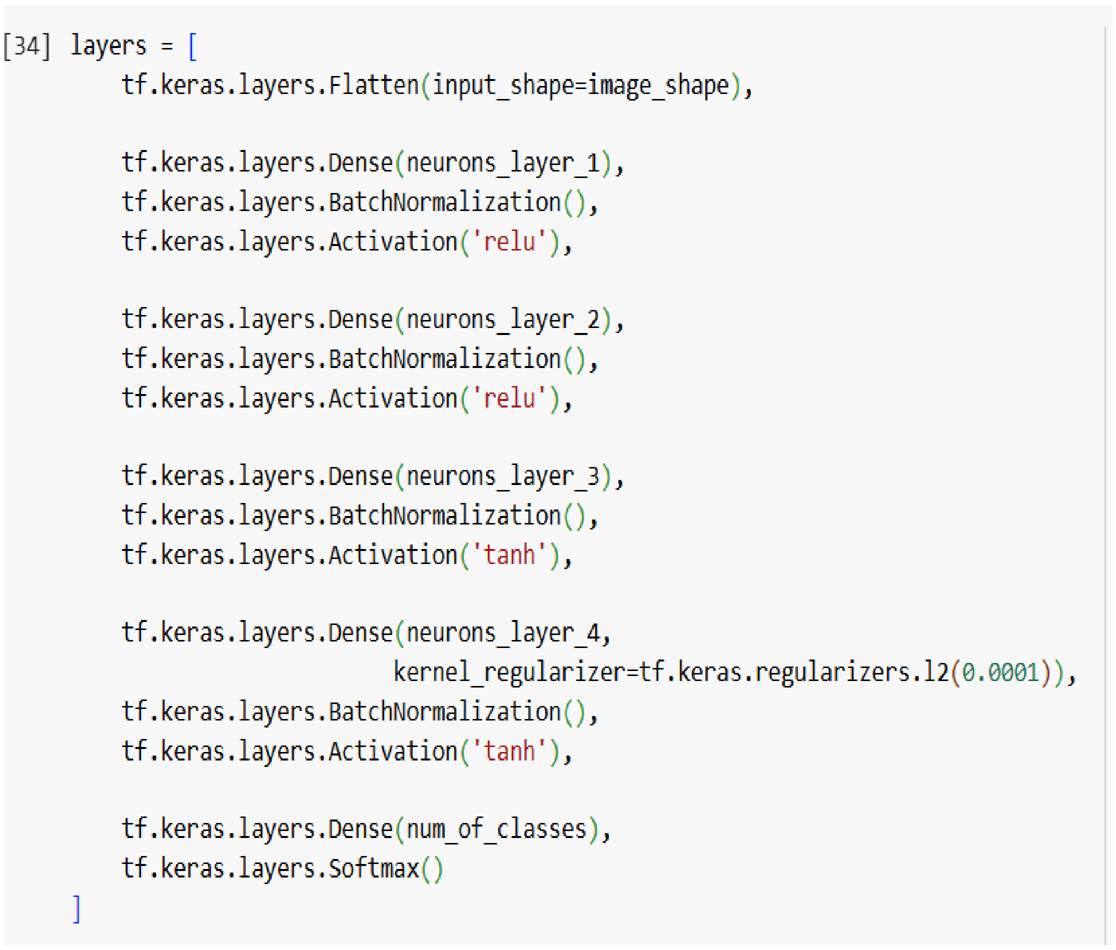
ניסוי מספר 16:
תוצאה:

loss: 0.0621

sparse_categorical_accuracy: 0.9811

val_loss: 0.0919

val_sparse_categorical_accuracy: 0.9769



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

         tf.keras.layers.Dense(neurons_layer_1),
         tf.keras.layers.BatchNormalization(),
         tf.keras.layers.Activation('relu'),

         tf.keras.layers.Dense(neurons_layer_2),
         tf.keras.layers.BatchNormalization(),
         tf.keras.layers.Activation('relu'),

         tf.keras.layers.Dense(neurons_layer_3),
         tf.keras.layers.BatchNormalization(),
         tf.keras.layers.Activation('tanh'),

         tf.keras.layers.Dense(neurons_layer_4,
                               kernel_regularizer=tf.keras.regularizers.l2(0.0001)),
         tf.keras.layers.BatchNormalization(),
         tf.keras.layers.Activation('tanh'),

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

ניסוי 16:

תוצאה:

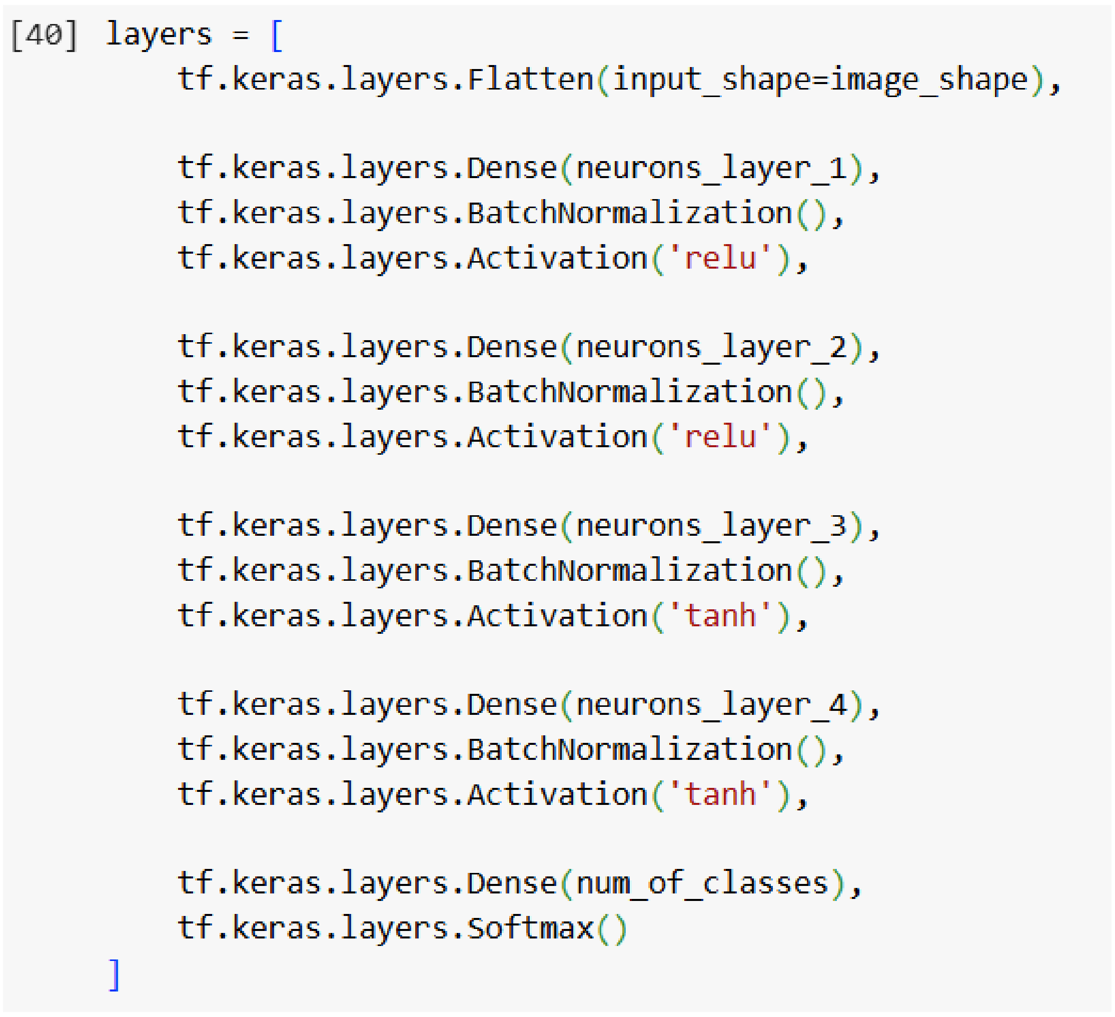
loss: 0.0679

sparse_categorical_accuracy: 0.9794

val_loss: 0.0837

val_sparse_categorical_accuracy: 0.9780

פרמטרים:



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

         tf.keras.layers.Dense(neurons_layer_1),
         tf.keras.layers.BatchNormalization(),
         tf.keras.layers.Activation('relu'),

         tf.keras.layers.Dense(neurons_layer_2),
         tf.keras.layers.BatchNormalization(),
         tf.keras.layers.Activation('relu'),

         tf.keras.layers.Dense(neurons_layer_3),
         tf.keras.layers.BatchNormalization(),
         tf.keras.layers.Activation('tanh'),

         tf.keras.layers.Dense(neurons_layer_4),
         tf.keras.layers.BatchNormalization(),
         tf.keras.layers.Activation('tanh'),

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

הניסוי הטוב ביותר הוא ניסוי מספר 16:

* הפער בין דיוק האימון (97.94%) לבין דיוק האימות (97.80%) הוא קטן, מה שאומר שהמודל אינו סובל מ-overfitting

* ערך האובדן באימות של 0.0837 הוא מהנמוכים ביותר מבין כל הניסויים, מה שמעיד על ביטחון גבוה בתחזיות

* המודל משיג דיוק גבוה תוך שמירה על ערכי אובדן נמוכים הן באימון והן באימות