## מטלה 2:

**חלק ב:**

הערה: מכיוון שעשיתי הרבה ניסויים היה לי יותר נוח לעקוב אחרי השפעת הפרמטרים השונים על התוצאה בעזרת טבלת אקסל מצרפת קישור לטבלה (המספור תואם למספור בקובץ הנוכחי).
hyperparameters

1.

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

tf.keras.layers.Dense(10),
tf.keras.layers.Activation('relu'),

tf.keras.layers.Dense(82),
tf.keras.layers.Activation('relu'),

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

tf.keras.layers.Dense(13),
tf.keras.layers.Activation('relu'),

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

**Results:**
loss: 0.1642 - sparse_categorical_accuracy: 0.9570 - val_loss: 0.2919 - val_sparse_categorical_accuracy: 0.9444

**The best Results were on epoch-14/50 :**
loss: 0.1424 - sparse_categorical_accuracy: 0.9588 - val_loss: 0.1831 - val_sparse_categorical_accuracy: 0.9533

2.

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

  tf.keras.layers.Dense,(10)
  tf.keras.layers.Activation('sigmoid'),

  tf.keras.layers.Dense,(82)
  tf.keras.layers.Activation('sigmoid'),

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

  tf.keras.layers.Dense,(13)
  tf.keras.layers.Activation('sigmoid'),

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

**Results:**

loss: 0.1805 - sparse_categorical_accuracy: 0.9490 - val_loss: 0.2266 - val_sparse_categorical_accuracy: 0.9381

**The best Results were on epoch-50/50 :**

loss: 0.1805 - sparse_categorical_accuracy: 0.9490 - val_loss: 0.2266 - val_sparse_categorical_accuracy: 0.9381

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

   tf.keras.layers.Dense(10, kernel_regularizer=tf.keras.regularizers.l2(0.0001)),
   tf.keras.layers.Activation('relu'),

   tf.keras.layers.Dense(82, kernel_regularizer=tf.keras.regularizers.l2(0.001)),
   tf.keras.layers.Activation('relu'),

   tf.keras.layers.Dense(46, kernel_regularizer=tf.keras.regularizers.l2(0.01)),
   tf.keras.layers.Activation('relu'),

   tf.keras.layers.Dense(13, kernel_regularizer=tf.keras.regularizers.l2(0.1)),
   tf.keras.layers.Activation('relu'),

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

   **Results:**
   loss: 0.2532 - sparse_categorical_accuracy: 0.9533 - val_loss: 0.2688 - val_sparse_categorical_accuracy: 0.9477

   **The best Results were on epoch-46/50 :**
   loss: 0.2618 - sparse_categorical_accuracy: 0.9519 - val_loss: 0.2692 - val_sparse_categorical_accuracy: 0.9484

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

   tf.keras.layers.Dense(10, kernel_regularizer=tf.keras.regularizers.l2(0.0001)),
   tf.keras.layers.BatchNormalization(),
   tf.keras.layers.Activation('relu'),

   tf.keras.layers.Dense(82, kernel_regularizer=tf.keras.regularizers.l2(0.001)),
   tf.keras.layers.BatchNormalization(),
   tf.keras.layers.Activation('relu'),

   tf.keras.layers.Dense(46, kernel_regularizer=tf.keras.regularizers.l2(0.01)),
   tf.keras.layers.BatchNormalization(),
   tf.keras.layers.Activation('relu'),

   tf.keras.layers.Dense(13, kernel_regularizer=tf.keras.regularizers.l2(0.1)),
   tf.keras.layers.BatchNormalization(),
   tf.keras.layers.Activation('relu'),

   tf.keras.layers.Dense(num_of_classes),
   tf.keras.layers.Softmax()
   ]
   **Results:**
   loss: 0.2350 - sparse_categorical_accuracy: 0.9411 - val_loss: 0.1881 - val_sparse_categorical_accuracy: 0.9567

   **The best Results were on epoch-41/50 :**
   loss: 0.2396 - sparse_categorical_accuracy: 0.9416 - val_loss: 0.1904 - val_sparse_categorical_accuracy: 0.9570

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

   tf.keras.layers.Dense,(10)
   tf.keras.layers.BatchNormalization,()
   tf.keras.layers.Activation('relu'),

   tf.keras.layers.Dense,(82)
   tf.keras.layers.BatchNormalization,()
   tf.keras.layers.Activation('relu'),

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

   tf.keras.layers.Dense,(13)
   tf.keras.layers.BatchNormalization,()
   tf.keras.layers.Activation('relu'),

   tf.keras.layers.Dense(num_of_classes),
   tf.keras.layers.Softmax()
    ]
   **Results:**
   loss: 0.1277 - sparse_categorical_accuracy: 0.9607 - val_loss: 0.1311 - val_sparse_categorical_accuracy: 0.9637

   **The best Results were on epoch-47/50 :**
   loss: 0.1198 - sparse_categorical_accuracy: 0.9620 - val_loss: 0.1347 - val_sparse_categorical_accuracy: 0.9638

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

   tf.keras.layers.Dense(10),
   tf.keras.layers.BatchNormalization(),
   tf.keras.layers.Activation('relu'),
   tf.keras.layers.Dropout(0.1),

   tf.keras.layers.Dense(82),
   tf.keras.layers.BatchNormalization(),
   tf.keras.layers.Activation('relu'),
   tf.keras.layers.Dropout(0.2),

   tf.keras.layers.Dense(46),
   tf.keras.layers.BatchNormalization(),
   tf.keras.layers.Activation('relu'),
   tf.keras.layers.Dropout(0.3),

   tf.keras.layers.Dense(13),
   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()
   ]

   **Results:**
   loss: 0.7679 - sparse_categorical_accuracy: 0.7763 - val_loss: 0.2828 - val_sparse_categorical_accuracy: 0.9295

   **The best Results were on epoch-34/50 :**
   loss: 0.7822 - sparse_categorical_accuracy: 0.7674 - val_loss: 0.2802 - val_sparse_categorical_accuracy: 0.9323

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

   tf.keras.layers.Dense(10,
   kernel_regularizer=tf.keras.regularizers.l2(0.00001)),
   tf.keras.layers.BatchNormalization(),
   tf.keras.layers.Activation('relu'),
   tf.keras.layers.Dropout(0.1),

   tf.keras.layers.Dense(82,
   kernel_regularizer=tf.keras.regularizers.l2(0.0001)),
   tf.keras.layers.BatchNormalization(),
   tf.keras.layers.Activation('relu'),
   tf.keras.layers.Dropout(0.2),

   tf.keras.layers.Dense(46, kernel_regularizer=tf.keras.regularizers.l2(0.001)),
   tf.keras.layers.BatchNormalization(),
   tf.keras.layers.Activation('relu'),
   tf.keras.layers.Dropout(0.3),

   tf.keras.layers.Dense(13, kernel_regularizer=tf.keras.regularizers.l2(0.01)),
   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()
   ]

   **Results:**
   loss: 0.8121 - sparse_categorical_accuracy: 0.7667 - val_loss: 0.3339 - val_sparse_categorical_accuracy: 0.9188

   **The best Results were on epoch-41/50 :**
   loss: 0.8249 - sparse_categorical_accuracy: 0.7659 - val_loss: 0.3243 - val_sparse_categorical_accuracy: 0.9243

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

   tf.keras.layers.Dense(10,
   kernel_regularizer=tf.keras.regularizers.l2(0.00001)),
   tf.keras.layers.Activation('relu'),
   tf.keras.layers.Dropout(0.1),

   tf.keras.layers.Dense(82,
   kernel_regularizer=tf.keras.regularizers.l2(0.0001)),
   tf.keras.layers.Activation('relu'),
   tf.keras.layers.Dropout(0.3),

   tf.keras.layers.Dense(46, kernel_regularizer=tf.keras.regularizers.l2(0.001)),
   tf.keras.layers.Activation('relu'),
   tf.keras.layers.Dropout(0.2),

   tf.keras.layers.Dense(13, 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()
   ]

   **Results:**
   loss: 0.4546 - sparse_categorical_accuracy: 0.8875 - val_loss: 0.2854 - val_sparse_categorical_accuracy: 0.9289

   **The best Results were on epoch-40/50 :**
   loss: 0.4489 - sparse_categorical_accuracy: 0.8873 - val_loss: 0.2857 - val_sparse_categorical_accuracy: 0.9294

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

   tf.keras.layers.Dense(10,
   kernel_regularizer=tf.keras.regularizers.l2(0.0001)),
   tf.keras.layers.BatchNormalization(),
   tf.keras.layers.Activation('relu'),
   tf.keras.layers.Dropout(0.3),

   tf.keras.layers.Dense(82, kernel_regularizer=tf.keras.regularizers.l2(0.001)),
   tf.keras.layers.BatchNormalization(),
   tf.keras.layers.Activation('relu'),
   tf.keras.layers.Dropout(0.3),

   tf.keras.layers.Dense(46, 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(13, kernel_regularizer=tf.keras.regularizers.l2(0.1)),
   tf.keras.layers.BatchNormalization(),
   tf.keras.layers.Activation('relu'),
   tf.keras.layers.Dropout(0.3),

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

   **Results:**
   loss: 1.0943 - sparse_categorical_accuracy: 0.6875 - val_loss: 0.6640 - val_sparse_categorical_accuracy: 0.8734

   **The best Results were on epoch-49/50 :**
   loss: 1.0905 - sparse_categorical_accuracy: 0.6876 - val_loss: 0.6417 - val_sparse_categorical_accuracy: 0.8805

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

    tf.keras.layers.Dense(10, kernel_regularizer=tf.keras.regularizers.l1(0.0001)),
    tf.keras.layers.BatchNormalization(),
    tf.keras.layers.Activation('relu'),

    tf.keras.layers.Dense(82, kernel_regularizer=tf.keras.regularizers.l1(0.001)),
    tf.keras.layers.BatchNormalization(),
    tf.keras.layers.Activation('relu'),

    tf.keras.layers.Dense(46, kernel_regularizer=tf.keras.regularizers.l1(0.01)),
    tf.keras.layers.BatchNormalization(),
    tf.keras.layers.Activation('relu'),

    tf.keras.layers.Dense(13, kernel_regularizer=tf.keras.regularizers.l1(0.1)),
    tf.keras.layers.BatchNormalization(),
    tf.keras.layers.Activation('relu'),

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

    **Results:**
    loss: 0.4317 - sparse_categorical_accuracy: 0.9222 - val_loss: 0.4056 - val_sparse_categorical_accuracy: 0.9334

    **The best Results were on epoch-30/50 :**
    loss: 0.4382 - sparse_categorical_accuracy: 0.9200 - val_loss: 0.3784 - val_sparse_categorical_accuracy: 0.9419

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

    tf.keras.layers.Dense(10, kernel_regularizer=tf.keras.regularizers.l1(0.0001)),
    tf.keras.layers.Activation('relu'),

    tf.keras.layers.Dense(82, kernel_regularizer=tf.keras.regularizers.l1(0.001)),
    tf.keras.layers.Activation('relu'),

    tf.keras.layers.Dense(46, kernel_regularizer=tf.keras.regularizers.l1(0.01)),
    tf.keras.layers.Activation('relu'),

    tf.keras.layers.Dense(13, kernel_regularizer=tf.keras.regularizers.l1(0.1)),
    tf.keras.layers.Activation('relu'),

    tf.keras.layers.Dense(num_of_classes),
    tf.keras.layers.Softmax()
     ]
    **Results:**
    loss: 0.5546 - sparse_categorical_accuracy: 0.9058 - val_loss: 0.5825 - val_sparse_categorical_accuracy: 0.8998

    **The best Results were on epoch-42/50 :**
    loss: 0.5611 - sparse_categorical_accuracy: 0.9080 - val_loss: 0.5790 - val_sparse_categorical_accuracy: 0.9057

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

    tf.keras.layers.Dense(10),
    tf.keras.layers.Activation('relu'),
    tf.keras.layers.Dropout(0.1),

    tf.keras.layers.Dense(82),
    tf.keras.layers.Activation('relu'),
    tf.keras.layers.Dropout(0.3),

    tf.keras.layers.Dense(46),
    tf.keras.layers.Activation('relu'),
    tf.keras.layers.Dropout(0.2),

    tf.keras.layers.Dense(13),
    tf.keras.layers.Activation('relu'),
    tf.keras.layers.Dropout(0.1),

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

    **Results:**
    loss: 0.5935 - sparse_categorical_accuracy: 0.8450 - val_loss: 0.3898 - val_sparse_categorical_accuracy: 0.8950

    **The best Results were on epoch-46/50 :**
    loss: 0.6313 - sparse_categorical_accuracy: 0.8413 - val_loss: 0.3816 - val_sparse_categorical_accuracy: 0.9075

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

    tf.keras.layers.Dense(10),
    tf.keras.layers.BatchNormalization(),
    tf.keras.layers.Activation('elu'),

    tf.keras.layers.Dense(82),
    tf.keras.layers.BatchNormalization(),
    tf.keras.layers.Activation('elu'),

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

    tf.keras.layers.Dense(13),
    tf.keras.layers.BatchNormalization(),
    tf.keras.layers.Activation('elu'),

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

    **Results:**
    loss: 0.1242 - sparse_categorical_accuracy: 0.9601 - val_loss: 0.1200 - val_sparse_categorical_accuracy: 0.9644

    **The best Results were on epoch-48/50 :**
    loss: 0.1204 - sparse_categorical_accuracy: 0.9619 - val_loss: 0.1189 - val_sparse_categorical_accuracy: 0.9672

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

    tf.keras.layers.Dense(10),
    tf.keras.layers.BatchNormalization(),
    tf.keras.layers.Activation('elu'),
    tf.keras.layers.Dropout(0.3),

    tf.keras.layers.Dense(82),
    tf.keras.layers.BatchNormalization(),
    tf.keras.layers.Activation('elu'),
    tf.keras.layers.Dropout(0.3),

    tf.keras.layers.Dense(46),
    tf.keras.layers.BatchNormalization(),
    tf.keras.layers.Activation('elu'),
    tf.keras.layers.Dropout(0.3),

    tf.keras.layers.Dense(13),
    tf.keras.layers.BatchNormalization(),
    tf.keras.layers.Activation('elu'),
    tf.keras.layers.Dropout(0.3),

    tf.keras.layers.Dense(num_of_classes),
    tf.keras.layers.Softmax()
     ]
    **Results:**
    loss: 0.6501 - sparse_categorical_accuracy: 0.8129 - val_loss: 0.2790 - val_sparse_categorical_accuracy: 0.9166

    **The best Results were on epoch-43/50 :**
    loss: 0.6482 - sparse_categorical_accuracy: 0.8131 - val_loss: 0.2790 - val_sparse_categorical_accuracy: 0.9189

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

    tf.keras.layers.Dense(10),
    tf.keras.layers.BatchNormalization(),
    tf.keras.layers.Activation('elu'),
    tf.keras.layers.Dropout(0.1),

    tf.keras.layers.Dense(82),
    tf.keras.layers.BatchNormalization(),
    tf.keras.layers.Activation('elu'),
    tf.keras.layers.Dropout(0.2),

    tf.keras.layers.Dense(46),
    tf.keras.layers.BatchNormalization(),
    tf.keras.layers.Activation('elu'),
    tf.keras.layers.Dropout(0.3),

    tf.keras.layers.Dense(13),
    tf.keras.layers.BatchNormalization(),
    tf.keras.layers.Activation('elu'),
    tf.keras.layers.Dropout(0.4),

    tf.keras.layers.Dense(num_of_classes),
    tf.keras.layers.Softmax()
     ]
    **Results:**
    loss: 0.4771 - sparse_categorical_accuracy: 0.8694 - val_loss: 0.2054 - val_sparse_categorical_accuracy: 0.9398

    **The best Results were epoch-45/50 :**
    loss: 0.4754 - sparse_categorical_accuracy: 0.8705 - val_loss: 0.2055 - val_sparse_categorical_accuracy: 0.9425

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

    tf.keras.layers.Dense(10),
    tf.keras.layers.BatchNormalization(),
    tf.keras.layers.Activation('elu'),
    tf.keras.layers.Dropout(0.1),

    tf.keras.layers.Dense(82),
    tf.keras.layers.BatchNormalization(),
    tf.keras.layers.Activation('elu'),
    tf.keras.layers.Dropout(0.1),

    tf.keras.layers.Dense(46),
    tf.keras.layers.BatchNormalization(),
    tf.keras.layers.Activation('elu'),
    tf.keras.layers.Dropout(0.2),

    tf.keras.layers.Dense(13),
    tf.keras.layers.BatchNormalization(),
    tf.keras.layers.Activation('elu'),
    tf.keras.layers.Dropout(0.3),

    tf.keras.layers.Dense(num_of_classes),
    tf.keras.layers.Softmax()
     ]
    **Results:**
    loss: 0.3506 - sparse_categorical_accuracy: 0.9034 - val_loss: 0.1546 - val_sparse_categorical_accuracy: 0.9542

    **The best Results were epoch-44/50 :**
    loss: 0.3482 - sparse_categorical_accuracy: 0.9035 - val_loss: 0.1563 - val_sparse_categorical_accuracy: 0.9553

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

    tf.keras.layers.Dense(10, kernel_regularizer=tf.keras.regularizers.l2(0.0001)),
    tf.keras.layers.BatchNormalization(),
    tf.keras.layers.Activation('elu'),
    tf.keras.layers.Dropout(0.1),

    tf.keras.layers.Dense(82, kernel_regularizer=tf.keras.regularizers.l2(0.001)),
    tf.keras.layers.BatchNormalization(),
    tf.keras.layers.Activation('elu'),
    tf.keras.layers.Dropout(0.1),

    tf.keras.layers.Dense(46, kernel_regularizer=tf.keras.regularizers.l2(0.01)),
    tf.keras.layers.BatchNormalization(),
    tf.keras.layers.Activation('elu'),
    tf.keras.layers.Dropout(0.2),

    tf.keras.layers.Dense(13, kernel_regularizer=tf.keras.regularizers.l2(0.1)),
    tf.keras.layers.BatchNormalization(),
    tf.keras.layers.Activation('elu'),
    tf.keras.layers.Dropout(0.3),

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

    **Results:**
    loss: 0.5034 - sparse_categorical_accuracy: 0.8790 - val_loss: 0.2750 - val_sparse_categorical_accuracy: 0.9416

    **The best Results were epoch-44/50 :**
    loss: 0.5114 - sparse_categorical_accuracy: 0.8739 - val_loss: 0.2563 - val_sparse_categorical_accuracy: 0.9457

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

    tf.keras.layers.Dense(10, kernel_regularizer=tf.keras.regularizers.l1(0.0001)),
    tf.keras.layers.BatchNormalization(),
    tf.keras.layers.Activation('elu'),
    tf.keras.layers.Dropout(0.1),

    tf.keras.layers.Dense(82, kernel_regularizer=tf.keras.regularizers.l1(0.001)),
    tf.keras.layers.BatchNormalization(),
    tf.keras.layers.Activation('elu'),
    tf.keras.layers.Dropout(0.1),

    tf.keras.layers.Dense(46, kernel_regularizer=tf.keras.regularizers.l1(0.01)),
    tf.keras.layers.BatchNormalization(),
    tf.keras.layers.Activation('elu'),
    tf.keras.layers.Dropout(0.2),

    tf.keras.layers.Dense(13, kernel_regularizer=tf.keras.regularizers.l1(0.1)),
    tf.keras.layers.BatchNormalization(),
    tf.keras.layers.Activation('elu'),
    tf.keras.layers.Dropout(0.3),

    tf.keras.layers.Dense(num_of_classes),
    tf.keras.layers.Softmax()
     ]
    **Results:**
    loss: 0.8333 - sparse_categorical_accuracy: 0.8441 - val_loss: 0.5246 - val_sparse_categorical_accuracy: 0.9299

    **The best Results were epoch-40/50 :**
    loss: 0.8201 - sparse_categorical_accuracy: 0.8466 - val_loss: 0.5167 - val_sparse_categorical_accuracy: 0.9324

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

    tf.keras.layers.Dense(10),
    tf.keras.layers.BatchNormalization(),
    tf.keras.layers.Activation('tanh'),

    tf.keras.layers.Dense(82),
    tf.keras.layers.BatchNormalization(),
    tf.keras.layers.Activation('tanh'),


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


    tf.keras.layers.Dense(13),
    tf.keras.layers.BatchNormalization(),
    tf.keras.layers.Activation('tanh'),


    tf.keras.layers.Dense(num_of_classes),
    tf.keras.layers.Softmax()
     ]
    **Results:**
    loss: 0.1179 - sparse_categorical_accuracy: 0.9628 - val_loss: 0.1296 - val_sparse_categorical_accuracy: 0.9634

    **The best Results were epoch-45/50 :**
    loss: 0.1188 - sparse_categorical_accuracy: 0.9622 - val_loss: 0.1241 - val_sparse_categorical_accuracy: 0.9648

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

    tf.keras.layers.Dense(10, kernel_regularizer=tf.keras.regularizers.l2(0.0001)),
    tf.keras.layers.BatchNormalization(),
    tf.keras.layers.Activation('tanh'),

    tf.keras.layers.Dense(82, kernel_regularizer=tf.keras.regularizers.l2(0.001)),
    tf.keras.layers.BatchNormalization(),
    tf.keras.layers.Activation('tanh'),


    tf.keras.layers.Dense(46, kernel_regularizer=tf.keras.regularizers.l2(0.01)),
    tf.keras.layers.BatchNormalization(),
    tf.keras.layers.Activation('tanh'),


    tf.keras.layers.Dense(13, kernel_regularizer=tf.keras.regularizers.l2(0.1)),
    tf.keras.layers.BatchNormalization(),
    tf.keras.layers.Activation('tanh'),


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

    **Results:**
    loss: 0.2271 - sparse_categorical_accuracy: 0.9446 - val_loss: 0.1992 - val_sparse_categorical_accuracy: 0.9530

    **The best Results were epoch-38/50 :**
    loss: 0.2322 - sparse_categorical_accuracy: 0.9426 - val_loss: 0.1922 - val_sparse_categorical_accuracy: 0.9566

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

    tf.keras.layers.Dense(10, kernel_regularizer=tf.keras.regularizers.l2(0.0001)),
    tf.keras.layers.BatchNormalization(),
    tf.keras.layers.Activation('elu'),

    tf.keras.layers.Dense(82, kernel_regularizer=tf.keras.regularizers.l2(0.001)),
    tf.keras.layers.BatchNormalization(),
    tf.keras.layers.Activation('elu'),


    tf.keras.layers.Dense(46, kernel_regularizer=tf.keras.regularizers.l2(0.01)),
    tf.keras.layers.BatchNormalization(),
    tf.keras.layers.Activation('elu'),


    tf.keras.layers.Dense(13, kernel_regularizer=tf.keras.regularizers.l2(0.1)),
    tf.keras.layers.BatchNormalization(),
    tf.keras.layers.Activation('elu'),


    tf.keras.layers.Dense(num_of_classes),
    tf.keras.layers.Softmax()
     ]
    **Results:**
    loss: 0.2245 - sparse_categorical_accuracy: 0.9430 - val_loss: 0.2091 - val_sparse_categorical_accuracy: 0.9500

    **The best Results were epoch-34/50 :**
    loss: 0.2364 - sparse_categorical_accuracy: 0.9405 - val_loss: 0.1856 - val_sparse_categorical_accuracy: 0.9570

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

    tf.keras.layers.Dense(10),
    tf.keras.layers.Activation('elu'),
    tf.keras.layers.Dropout(0.1),

    tf.keras.layers.Dense(82),
    tf.keras.layers.Activation('elu'),
    tf.keras.layers.Dropout(0.1),

    tf.keras.layers.Dense(46),
    tf.keras.layers.Activation('elu'),
    tf.keras.layers.Dropout(0.2),

    tf.keras.layers.Dense(13),
    tf.keras.layers.Activation('elu'),
    tf.keras.layers.Dropout(0.3),

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

    **Results:**
    loss: 0.3399 - sparse_categorical_accuracy: 0.9125 - val_loss: 0.2132 - val_sparse_categorical_accuracy: 0.9415

    **The best Results were epoch-47/50 :**
    loss: 0.3416 - sparse_categorical_accuracy: 0.9118 - val_loss: 0.2139 - val_sparse_categorical_accuracy: 0.9453

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

    tf.keras.layers.Dense(10),
    tf.keras.layers.Activation('elu'),
    tf.keras.layers.Dropout(0.1),

    tf.keras.layers.Dense(82),
    tf.keras.layers.Activation('elu'),
    tf.keras.layers.Dropout(0.1),

    tf.keras.layers.Dense(46),
    tf.keras.layers.Activation('elu'),
    tf.keras.layers.Dropout(0.2),

    tf.keras.layers.Dense(13),
    tf.keras.layers.Activation('elu'),
    tf.keras.layers.Dropout(0.2),

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

    **Results:**

    loss: 0.2869 - sparse_categorical_accuracy: 0.9223 - val_loss: 0.2150 - val_sparse_categorical_accuracy: 0.9446

    **The best Results were epoch-41/50 :**

    loss: 0.2928 - sparse_categorical_accuracy: 0.9222 - val_loss: 0.1964 - val_sparse_categorical_accuracy: 0.9472

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

    tf.keras.layers.Dense(10, kernel_regularizer=tf.keras.regularizers.l2(0.0001)),
    tf.keras.layers.Activation('elu'),
    tf.keras.layers.Dropout(0.1),

    tf.keras.layers.Dense(82, kernel_regularizer=tf.keras.regularizers.l2(0.001)),
    tf.keras.layers.Activation('elu'),
    tf.keras.layers.Dropout(0.1),

    tf.keras.layers.Dense(46, kernel_regularizer=tf.keras.regularizers.l2(0.01)),
    tf.keras.layers.Activation('elu'),
    tf.keras.layers.Dropout(0.2),

    tf.keras.layers.Dense(13, kernel_regularizer=tf.keras.regularizers.l2(0.1)),
    tf.keras.layers.Activation('elu'),
    tf.keras.layers.Dropout(0.2),

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

    **Results:**
    loss: 0.4596 - sparse_categorical_accuracy: 0.9022 - val_loss: 0.3413 - val_sparse_categorical_accuracy: 0.9338

    **The best Results were epoch-46/50 :**
    loss: 0.4591 - sparse_categorical_accuracy: 0.9015 - val_loss: 0.3271 - val_sparse_categorical_accuracy: 0.9374

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

    tf.keras.layers.Dense(10),
    tf.keras.layers.Activation('sigmoid'),
    tf.keras.layers.Dropout(0.1),

    tf.keras.layers.Dense(82),
    tf.keras.layers.Activation('sigmoid'),
    tf.keras.layers.Dropout(0.1),

    tf.keras.layers.Dense(46),
    tf.keras.layers.Activation('sigmoid'),
    tf.keras.layers.Dropout(0.2),

    tf.keras.layers.Dense(13),
    tf.keras.layers.Activation('sigmoid'),
    tf.keras.layers.Dropout(0.2),

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

    **Results:**
    loss: 0.5642 - sparse_categorical_accuracy: 0.8526 - val_loss: 0.3406 - val_sparse_categorical_accuracy: 0.9130

    **The best Results were epoch-49/50 :**
    loss: 0.5489 - sparse_categorical_accuracy: 0.8557 - val_loss: 0.3397 - val_sparse_categorical_accuracy: 0.9160

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

    tf.keras.layers.Dense(10),
    tf.keras.layers.BatchNormalization(),
    tf.keras.layers.Activation('sigmoid'),
    tf.keras.layers.Dropout(0.1),

    tf.keras.layers.Dense(82),
    tf.keras.layers.BatchNormalization(),
    tf.keras.layers.Activation('sigmoid'),
    tf.keras.layers.Dropout(0.1),

    tf.keras.layers.Dense(46),
    tf.keras.layers.BatchNormalization(),
    tf.keras.layers.Activation('sigmoid'),
    tf.keras.layers.Dropout(0.2),

    tf.keras.layers.Dense(13),
    tf.keras.layers.BatchNormalization(),
    tf.keras.layers.Activation('sigmoid'),
    tf.keras.layers.Dropout(0.2),

    tf.keras.layers.Dense(num_of_classes),
    tf.keras.layers.Softmax()
     ]
    **Results:**
    loss: 0.5606 - sparse_categorical_accuracy: 0.8402 - val_loss: 0.2326 - val_sparse_categorical_accuracy: 0.9352

    **The best Results were epoch-50/50 :**
    loss: 0.5606 - sparse_categorical_accuracy: 0.8402 - val_loss: 0.2326 - val_sparse_categorical_accuracy: 0.9352

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

    tf.keras.layers.Dense(10),
    tf.keras.layers.BatchNormalization(),
    tf.keras.layers.Activation('sigmoid'),

    tf.keras.layers.Dense(82),
    tf.keras.layers.BatchNormalization(),
    tf.keras.layers.Activation('sigmoid'),

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

    tf.keras.layers.Dense(13),
    tf.keras.layers.BatchNormalization(),
    tf.keras.layers.Activation('sigmoid'),

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

    **Results:**
    loss: 0.1224 - sparse_categorical_accuracy: 0.9599 - val_loss: 0.1221 - val_sparse_categorical_accuracy: 0.9629

    **The best Results were epoch-49/50 :**
    loss: 0.1211 - sparse_categorical_accuracy: 0.9605 - val_loss: 0.1243 - val_sparse_categorical_accuracy: 0.9646

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

    tf.keras.layers.Dense(10, kernel_regularizer=tf.keras.regularizers.l2(0.0001)),
    tf.keras.layers.BatchNormalization(),
    tf.keras.layers.Activation('sigmoid'),

    tf.keras.layers.Dense(82, kernel_regularizer=tf.keras.regularizers.l2(0.001)),
    tf.keras.layers.BatchNormalization(),
    tf.keras.layers.Activation('sigmoid'),

    tf.keras.layers.Dense(46, kernel_regularizer=tf.keras.regularizers.l2(0.01)),
    tf.keras.layers.BatchNormalization(),
    tf.keras.layers.Activation('sigmoid'),

    tf.keras.layers.Dense(13, kernel_regularizer=tf.keras.regularizers.l2(0.1)),
    tf.keras.layers.BatchNormalization(),
    tf.keras.layers.Activation('sigmoid'),

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

    **Results:**
    loss: 0.2298 - sparse_categorical_accuracy: 0.9412 - val_loss: 0.2414 - val_sparse_categorical_accuracy: 0.9392

    **The best Results were epoch-47/50 :**
    loss: 0.2289 - sparse_categorical_accuracy: 0.9425 - val_loss: 0.2003 - val_sparse_categorical_accuracy: 0.9543

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

    tf.keras.layers.Dense(10, kernel_regularizer=tf.keras.regularizers.l1(0.0001)),
    tf.keras.layers.BatchNormalization(),
    tf.keras.layers.Activation('sigmoid'),

    tf.keras.layers.Dense(82, kernel_regularizer=tf.keras.regularizers.l1(0.001)),
    tf.keras.layers.BatchNormalization(),
    tf.keras.layers.Activation('sigmoid'),

    tf.keras.layers.Dense(46, kernel_regularizer=tf.keras.regularizers.l1(0.01)),
    tf.keras.layers.BatchNormalization(),
    tf.keras.layers.Activation('sigmoid'),

    tf.keras.layers.Dense(13, kernel_regularizer=tf.keras.regularizers.l1(0.1)),
    tf.keras.layers.BatchNormalization(),
    tf.keras.layers.Activation('sigmoid'),

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

    **Results:**
    loss: 0.4354 - sparse_categorical_accuracy: 0.9236 - val_loss: 0.4082 - val_sparse_categorical_accuracy: 0.9323

    **The best Results were epoch-27/50 :**
    loss: 0.4522 - sparse_categorical_accuracy: 0.9196 - val_loss: 0.3978 - val_sparse_categorical_accuracy: 0.9343

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

    tf.keras.layers.Dense(10),
    tf.keras.layers.Activation('elu'),

    tf.keras.layers.Dense(82),
    tf.keras.layers.Activation('elu'),

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

    tf.keras.layers.Dense(13),
    tf.keras.layers.Activation('elu'),

    tf.keras.layers.Dense(num_of_classes),
    tf.keras.layers.Softmax()
     ]
    **Results:**
    loss: 0.0864 - sparse_categorical_accuracy: 0.9753 - val_loss: 0.2184 - val_sparse_categorical_accuracy: 0.9550

    **The best Results were epoch-18/50 :**
    loss: 0.1097 - sparse_categorical_accuracy: 0.9665 - val_loss: 0.1518 - val_sparse_categorical_accuracy: 0.9574