1. Basic network
2. model = Sequential()
3. model.add(Flatten(input\_shape=(28, 28)))  # Flatten the 28x28 images to a 1D array
4. model.add(Dense(128, activation='relu'))
5. model.add(Dense(64, activation='relu'))
6. model.add(Dense(num\_classes, activation='softmax'))
7. # Compile the model
8. model.compile(optimizer=Adam(learning\_rate=0.001),loss='categorical\_crossentropy',metrics=['accuracy'])

11. # Train the model on the dataset
12. history = model.fit(train\_images, train\_labels,
13. epochs=10, batch\_size=128,
14. validation\_data=(test\_images, test\_labels))

Normal 97.75%

Shifted 93.8%

Rotated 96.75%

Both 90.67%

2. Convolution Network

model = Sequential()

    model.add(Conv2D(32, kernel\_size=(3, 3), activation='relu', input\_shape=(28, 28, 1)))

    model.add(MaxPooling2D(pool\_size=(2, 2)))

    model.add(Conv2D(64, kernel\_size=(3, 3), activation='relu'))

    model.add(MaxPooling2D(pool\_size=(2, 2)))

    model.add(Flatten())

    model.add(Dense(128, activation='relu'))

    model.add(Dropout(0.5))  # Adding dropout for regularization

    model.add(Dense(num\_classes, activation='softmax'))

    # Compile the model

    model.compile(optimizer=Adam(learning\_rate=0.001),

                loss='categorical\_crossentropy',

                metrics=['accuracy'])

    # Train the model on the rotated and shifted images

    history = model.fit(train\_images, train\_labels,

                        epochs=10, batch\_size=128,

                        validation\_data=(test\_images, test\_labels))

normal 99.22%

shifted 98.22

rotated 98.81

both 96.71

Convolution with zeropadding

 model = Sequential()

    model.add(ZeroPadding2D((1,1),input\_shape=(28,28,1)))

    model.add(Conv2D(32, kernel\_size=(3, 3), activation='relu'))

    model.add(MaxPooling2D(pool\_size=(2, 2)))

    model.add(Conv2D(64, kernel\_size=(3, 3), activation='relu'))

    model.add(MaxPooling2D(pool\_size=(2, 2)))

    model.add(Flatten())

    model.add(Dense(128, activation='relu'))

    model.add(Dropout(0.5))  *# Adding dropout for regularization*

    model.add(Dense(num\_classes, activation='softmax'))

*# Compile the model*

    model.compile(optimizer=Adam(learning\_rate=0.001),

                loss='categorical\_crossentropy',

                metrics=['accuracy'])

*# Train the model on the rotated and shifted images*

    history = model.fit(train\_images, train\_labels,

                        epochs=10, batch\_size=128,

                        validation\_data=(test\_images, test\_labels))

Normal: 99.2%

Shifted: 97.15%

Rotated: 97.16%

Both: 88.43%