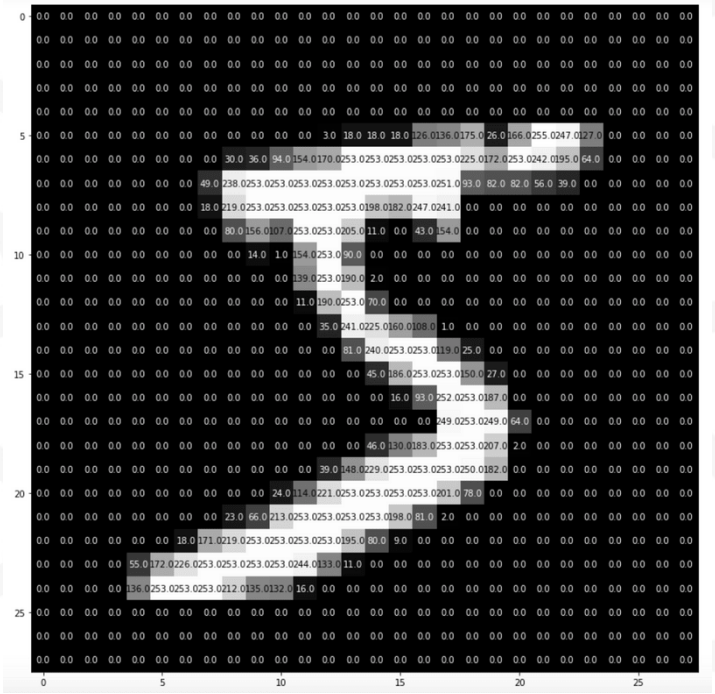
Convolutional Neural Networks

CNNs for Image Recognition



Convolutional Neural Networks (CNNs) are typically used in image recognition problems.

In image recognition, images are broken into pixels and each pixel has a corresponding numeric value.

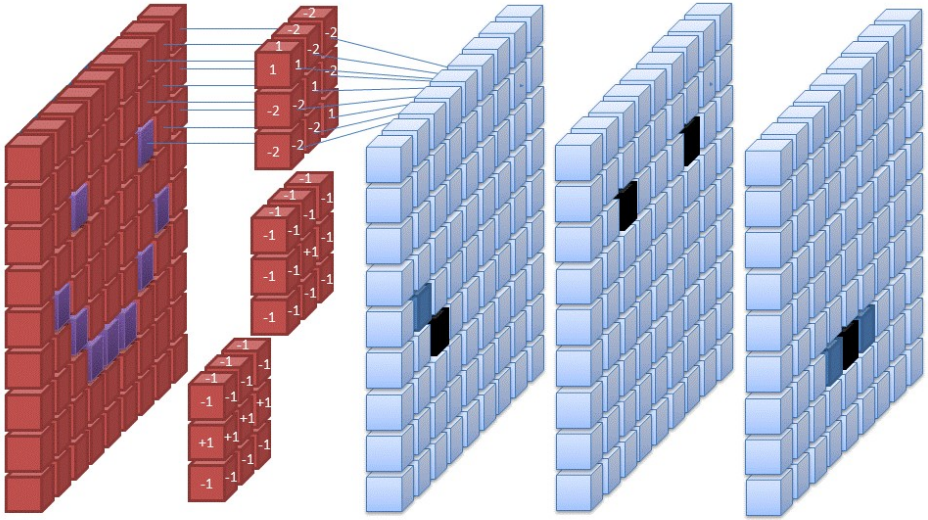
These pixels are used as input to the CNN.

Types of Layers

There are several types of layers that can be used in a neural network. We will focus on two common types:

- **Dense Layers** - Dense layers are used when connections can exist among any feature to any other feature in a data set. This corresponds to an ANN.
- **Convolutional Layers** - Convolutional layers can be used to detect patterns in an image. Early convolutional layers can detect things like edges or geometric shapes. Deeper convolutional layers can detect more sophisticated patterns like eyes, noses, etc. Convolutional layers are used when nearby connections among the features are important, but farther connections are not. This corresponds to a CNN.

Convolutional Layers



Example:
https://deeplizard.com/resource/pavq7noze2

Max Pooling

Max pooling is typically added to CNNs following individual convolutional layers to reduce the dimensionality of the image.

Example:
https://deeplizard.com/resource/pavq7noze3

CNN in Keras

```
model = Sequential()

# Define input layer & first hidden layer
model.add(Conv2D(64, (3,3), input_shape=(100,100,1), activation = 'relu'))
model.add(MaxPooling2D(pool_size=(2,2)))

# Define second hidden layer
model.add(Conv2D(64, (3,3), activation = 'relu'))
model.add(MaxPooling2D(pool_size=(2,2)))

# Flatten data to be used in output layer
model.add(Flatten())

# Define output layer
model.add(Dense(10, activation = 'softmax'))

# Compile model
model.compile(optimizer='adam', loss='sparse_categorical_crossentropy', metrics=['accuracy'])

# Fit model
model.fit(x_train2, y_train, epochs=8)
```

Conv2D - Specify layer, number of nodes, filter size (3,3), shape of input and activation function

MaxPooling2D - Specify pool size (2,2)

Need to flatten data before output layer

Output layer is a dense layer

Resources

Deep Lizard tutorials - https://deeplizard.com/learn/video/gZmobeGL0Yg

Keras - https://keras.io/

Tensorflow - https://www.tensorflow.org/