**MARKET BASKET INSIGHTS**

**TensorFlow and Keras are popular tools for building Artificial Neural Networks (ANNs). ANNs are a type of machine learning model inspired by the structure of the human brain. Here's a brief overview:**

**TensorFlow:** TensorFlow is an open-source machine learning framework developed by Google. It provides a wide range of tools and libraries for various machine learning tasks, including neural networks. With TensorFlow, you can build and train neural networks at a low-level, which gives you more control over the architecture, or you can use high-level APIs like Keras.

### **Basic Artificial neural network will be in a form of,**

1. **Input layer**– To get the data from the user or a client or a server to analyze and give the result.
2. **Hidden layers** – This layer can be in any number and these layers will analyze the inputs with passing through them with different biases, weights, and activation functions to provide an output
3. **Output Layer** – This is where we can get the result from a neural network.

### **Some of the features that determine the quality of our neural network are:**

1. Layers
2. Activation function
3. Loss function
4. Optimizer

## **Layers**

Layers in a neural network are very important as we saw earlier an artificial neural network consists of 3 layers an input layer, hidden layer, output layer.

## **Activation function**

Activation functions are simply mathematical methods that bring all the values inside a range of 0 to 1 so that it will be very easier for the machine to learn the data in its process of analyzing the data. There are a variety of activation functions that are supported by the Tensor flow. Some of the commonly used functions are,

1. Sigmoid
2. Relu
3. Softmax
4. Swish
5. Linear

## **Losses**

Loss functions are a very important thing to notice while creating a neural network because loss functions in the neural network will calculate the difference between the predicted output and the actual result and greatly help the optimizers in the neural nets to update the weights on its backpropagation.

There are many loss functions that were supported by the TensorFlow library, and again commonly used few are,

1. Mean Absolute
2. MeanSquaredError
3. Binary Crossentropy
4. Categorical Crossentropy
5. Sparse Categorical Crossentropy

## **Optimizers**

Optimizers are a very important thing because this is the function that helps the neural network to change the weights on the backpropagation so that the difference between the actual and predicted result will decrease at a gradual pace and obtain that point where the loss is very minimum and the model is able to predict more accurate results.

Again TensorFlow supports many optimizers to mention a few,

1. Gradient descent
2. SDG – Stochastic Gradient Descent
3. Adagrad
4. Adam

**Keras:** Keras is an open-source high-level neural networks API that runs on top of TensorFlow (or other backends like Theano and CNTK). It provides a user-friendly interface for designing and training neural networks. Keras is often the preferred choice for building ANNs because of its ease of use and flexibility.To build an ANN using TensorFlow and Keras, you would typically follow these

steps:

**Data Preparation:** Gather and preprocess your data. This includes splitting your dataset into training and testing sets, normalizing or scaling the data, and converting it into a format suitable for training.

**Model Design:** In Keras, you can design your neural network model by stacking layers. You can choose from various layers like Dense (fully connected), Convolutional, Recurrent, and more, to define the architecture of your ANN. You specify the number of neurons in each layer, activation functions, and other hyperparameters.

**Compile the Model:** After defining the architecture, you compile the model. This involves specifying the optimizer, loss function, and metrics to be used during training.Training: Fit the model to your training data using the model.fit() method. The model will iterate through the training data and adjust its parameters (weights and biases) to minimize the specified loss function.

**Evaluation:** Evaluate the model's performance on a separate testing dataset to assess its accuracy, loss, and other relevant metrics.

**Prediction:** Use the trained model to make predictions on new, unseen data.Fine-tuning: Depending on the results, you may need to fine-tune your model by adjusting hyperparameters, modifying the architecture, or collecting more data.Both TensorFlow and Keras have extensive documentation and online resources for learning and using them effectively. If you have specific questions or need more details on any of these steps, feel free to ask.

**Convolutional Neural Networks (CNNs)** are a class of deep learning models designed for processing structured grid data, such as images and video. They're particularly effective in computer vision tasks. CNNs use convolutional layers to automatically learn and extract features from input data. These layers consist of filters that slide over the input, performing convolutions to detect patterns like edges, shapes, and textures. Pooling layers reduce spatial dimensions, and fully connected layers enable classification or regression tasks. CNNs have been crucial in image recognition, object detection, and other visual tasks.

[Neural networks](https://www.ibm.com/topics/neural-networks) are a subset of machine learning, and they are at the heart of deep learning algorithms. They are comprised of node layers, containing an input layer, one or more hidden layers, and an output layer. Each node connects to another and has an associated weight and threshold. If the output of any individual node is above the specified threshold value, that node is activated, sending data to the next layer of the network. Otherwise, no data is passed along to the next layer of the network.

Convolutional neural networks are distinguished from other neural networks by their superior performance with image, speech, or audio signal inputs. They have three main types of layers, which are:

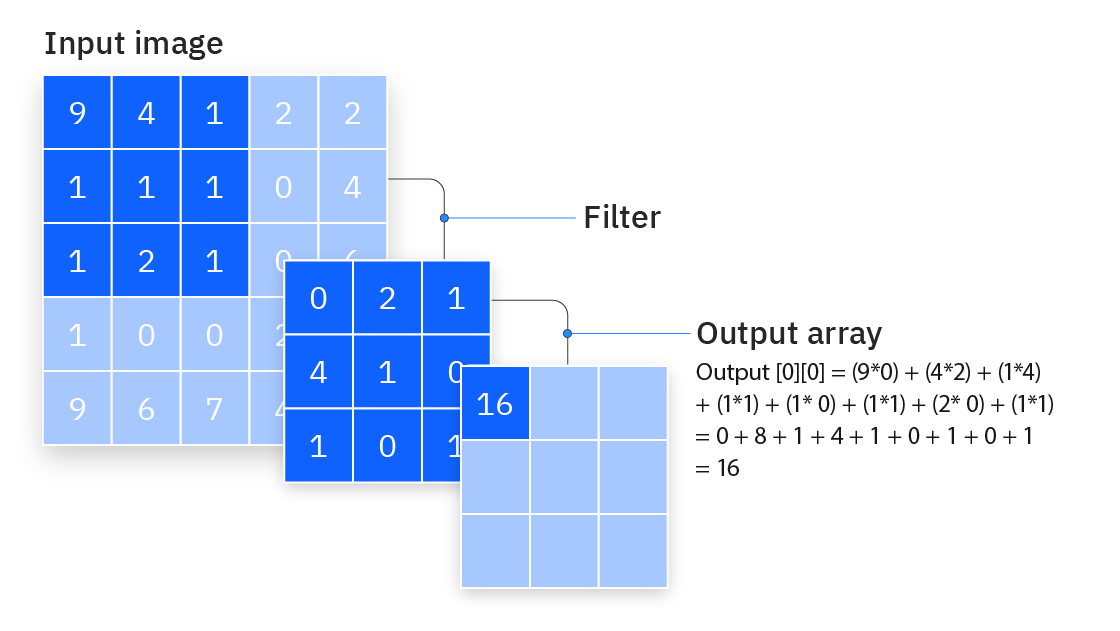
* Convolutional layer
* Pooling layer
* Fully-connected (FC) layer

1. The**number of filters**affects the depth of the output. For example, three distinct filters would yield three different feature maps, creating a depth of three.

2. **Stride** is the distance, or number of pixels, that the kernel moves over the input matrix. While stride values of two or greater is rare, a larger stride yields a smaller output.

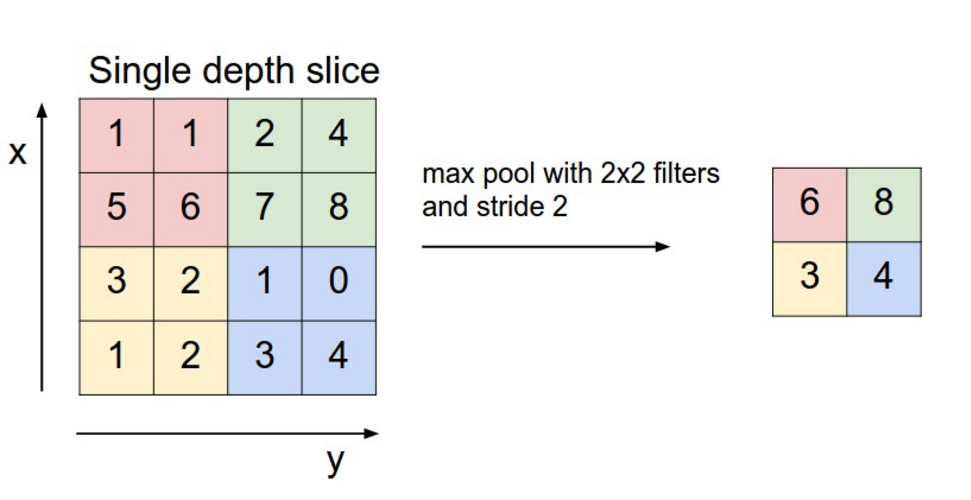
3. **Zero-padding** is usually used when the filters do not fit the input image. This sets all elements that fall outside of the input matrix to zero, producing a larger or equally sized output. There are three types of padding:

* **Valid padding:** This is also known as no padding. In this case, the last convolution is dropped if dimensions do not align.
* **Same padding:** This padding ensures that the output layer has the same size as the input layer
* **Full padding:**This type of padding increases the size of the output by adding zeros to the border of the input.



Additional convolutional layer

As we mentioned earlier, another convolution layer can follow the initial convolution layer. When this happens, the structure of the CNN can become hierarchical as the later layers can see the pixels within the receptive fields of prior layers.  As an example, let’s assume that we’re trying to determine if an image contains a bicycle. You can think of the bicycle as a sum of parts. It is comprised of a frame, handlebars, wheels, pedals, et cetera. Each individual part of the bicycle makes up a lower-level pattern in the neural net, and the combination of its parts represents a higher-level pattern, creating a feature hierarchy within the CNN. Ultimately, the convolutional layer converts the image into numerical values, allowing the neural network to interpret and extract relevant patterns.



### CNN architecture

Convolutional Neural Network consists of multiple layers like the input layer, Convolutional layer, Pooling layer, and fully connected layers.



*Simple CNN architecture*

The Convolutional layer applies filters to the input image to extract features, the Pooling layer downsamples the image to reduce computation, and the fully connected layer makes the final prediction. The network learns the optimal filters through backpropagation and gradient descent.

* **Marketing:** Social media platforms provide suggestions on who might be in photograph that has been posted on a profile, making it easier to tag friends in photo albums.
* **Healthcare:** Computer vision has been incorporated into radiology technology, enabling doctors to better identify cancerous tumors in healthy anatomy.
* **Retail:**Visual search has been incorporated into some e-commerce platforms, allowing brands to recommend items that would complement an existing wardrobe.
* **Automotive**: While the age of driverless cars hasn’t quite emerged, the underlying technology has started to make its way into automobiles, improving driver and passenger safety through features like lane line detection.

**OpenCV**, or Open Source Computer Vision Library, is an open-source computer vision and machine learning software library. It's designed to help developers create applications for tasks related to computer vision, image processing, and machine learning. OpenCV provides a wide range of functions and tools for working with images and videos, including features like object detection, face recognition, image stitching, and more. It supports various programming languages, including C++, Python, and Java, making it accessible for a broad community of developers. OpenCV has been widely used in applications ranging from robotics and autonomous vehicles to healthcare and augmented reality.

**OpenCV is a great tool for image processing and performing computer vision tasks. It is an open-source library that can be used to perform tasks like face detection, objection tracking, landmark detection, and much more.**

**OpenCV is open source and released under the Apache 2 License. It is free for commercial use**

OpenCV is written in the programming language C++, as is its primary interface, but it still retains a less comprehensive though extensive older C interface. All newer developments and algorithms appear in the C++ interface. There are language bindings in Python, Java, and MATLAB/Octave.

OpenCV is a software library that is used by programmers to create applications using computer vision. Modern computer vision tends to incorporate artificial intelligence techniques, so OpenCV contains a tool kit for the implementation of image based AI algorithms.