# Activation Functions & Non-Linearity: A Beginner's Guide with CIFAR-10

## 1 Introduction

Welcome to the exciting world of image recognition! In this book, we dive into how computers can "see" and understand images, just like humans. We explore a fundamental concept: **activation functions** and **non-linearity**, which are crucial for building intelligent image recognition systems.

Before we delve into technical details, let's start with a mission: **CIFAR-10**. Imagine you're tasked with building a program that can automatically identify objects in pictures. These objects include airplanes, automobiles, birds, cats, deer, dogs, frogs, horses, ships, and trucks. You have 60,000 images: 50,000 for training and 10,000 for testing.

Each image is a small, color (RGB) picture of size $32 \times 32$ pixels. Your goal is to train a model using the training images, then use the trained model to recognize objects in the test images, identifying which of the 10 object classes each image most likely belongs to.

This is the CIFAR-10 challenge, a perfect starting point for learning about image recognition. Along this journey, we will explore activation functions and non-linearity. Let's get started.

## 2 Revisiting the Basics

Before tackling the complexities of image recognition, let's recap some core concepts relevant to our task.

- **Neural Networks (NN):** Neural networks learn patterns from data through interconnected layers of "neurons." Data flows through these layers, and connections between neurons are adjusted during training to improve prediction accuracy.
- **Convolution:** Before feeding image data into a neural network, we extract information using convolution, a process that applies filters to capture important features.
- **Sequential:** In our code, `nn.Sequential` stacks neural network layers sequentially, like building with Lego bricks, each performing a specific task.
- **Layers:** Neural networks consist of various layers where data extraction and transformation occur.
- **Linearity:** Initial data extraction or transformation is performed linearly.

## 3 Convolutional Neural Networks (CNNs): A Glimpse

Since we are dealing with images, we use **Convolutional Neural Networks (CNNs)**. CNNs are neural networks designed to process images, leveraging their spatial structure to excel at tasks like object recognition. We refer to our CNN as a simple Convolutional Neural Network. CNNs use filters to extract features, which we will explore further.

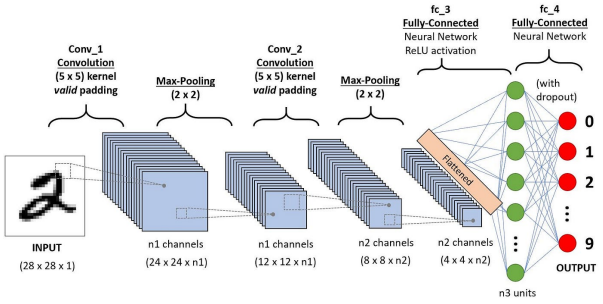


Figure 1: CNN

## 4 Data Preparation: Getting Ready for Action

Before training our CNN, we must prepare the CIFAR-10 dataset, which is conveniently available through libraries.

1. **Data Acquisition:** We fetch the CIFAR-10 dataset, containing 60,000 images (50,000 for training, 10,000 for testing).

2. **Normalization:** We normalize pixel values to a range between $-1$ and 1. This involves scaling values to $[0, 1]$ and then adjusting by a mean and standard deviation of 0.5, resulting in values between $-1$ and 1. This aids faster and more effective learning.

3. **Batching and Shuffling:** We divide training data into batches of 32 images and shuffle them to enhance training, similar to shuffling a deck of cards to avoid repetitive patterns.

4. **Creating Test Sets:** The test set is also batched into groups of 32 images.

## 5 Building Our CNN: Layer by Layer

Now, we define the architecture of our CNN by stacking layers to create a powerful image recognition system. The CIFAR-10 dataset is automatically downloaded by our library.

Here are the layers we use:

1. **Convolutional Layers (Conv2d):** These layers extract features from input images using filters.

    - *Input and Output:* We start with an input size of 3 (RGB channels) and an output size of 32 (for a $32 \times 32$ output).

- *Kernel Size:* We use a $3 \times 3$ kernel, determining the filter size that slides across the image.
- *Padding:* We apply padding of 1 to add extra pixels around the image border, preserving edge information.

*Note:* The input size is 3 due to RGB channels, and the output size is defined by the number of filters (e.g., 32).

2. **ReLU Activation:** The Rectified Linear Unit (ReLU) introduces **non-linearity** by setting negative values to zero and keeping positive values unchanged.
    - *Why Non-Linearity Matters:* Without non-linearity, neural networks can only learn linear relationships. Real-world data, like images, is rarely linear. Non-linear activation functions like ReLU enable learning complex patterns for accurate predictions.
3. **Max Pooling (MaxPool2d):** Pooling layers reduce image size, decreasing parameters and computation. A $32 \times 32$ image is downsized to $16 \times 16$ by halving its resolution, allowing exploration of new features with different filters.
4. **Flatten:** This layer converts 2D feature maps into a 1D vector, preparing data for fully connected layers.
5. **Linear Layers (Linear):** These fully connected layers perform linear transformations, similar to traditional neural networks.
6. **Final Linear Layer:** This layer maps the output to 10 classes, corresponding to the 10 object categories in CIFAR-10.

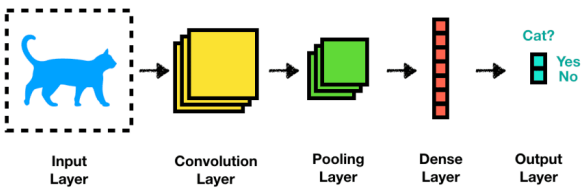


Figure 2: Structure of CNN

## 6 Training the Network: Learning to See

After defining the CNN, we train it by feeding training data and adjusting parameters to minimize prediction errors.

- **Loss Function:** We use `CrossEntropyLoss`, which measures the difference between predicted probabilities and true labels, incorporating a softmax function.
- **Optimizer:** We use the Adam optimizer to update parameters, leveraging backpropagation to minimize gradients.
- **Epochs:** An epoch is one pass through the training dataset. We train for multiple epochs (e.g., 5) to learn data patterns.

## 7 Diving Deeper: Understanding What's Happening Inside

With a single filter, the chain rule is applied during backpropagation to update weights. Filters determine the weights and gradients, shaping the learning process.

## 8 The Power of Pre-trained Models: ResNet

Training a CNN from scratch is time-consuming. Pre-trained models like **ResNet**, trained on massive datasets, offer powerful feature extraction, which we can leverage for faster and better results.

## 9 Looking Ahead: The World of CNNs

We've only scratched the surface of CNN capabilities. From object detection and image segmentation to facial recognition and medical imaging, CNNs are transforming various fields. As you continue, you'll discover more exciting applications and advanced techniques.