Introduction to CNNs

An overview of Convolutional Neural Networks and their applications in computer vision.

A

Understanding Convolutional Neural Networks

01

Definition of CNN

A Convolutional Neural Network (CNN) is a type of deep neural network used predominantly for image classification and recognition tasks.

02

Image Processing

CNNs are specifically designed to process data with grid-like topology, such as images, making them ideal for visual tasks.

03

Inspiration from Biology

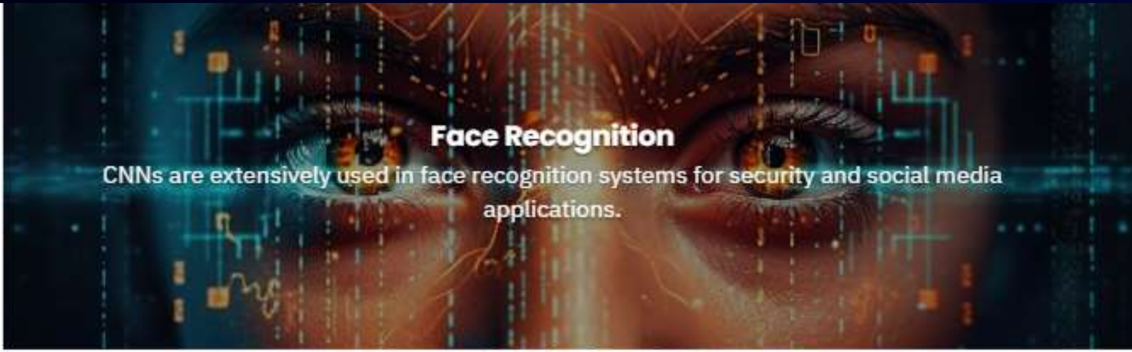
The architecture of CNNs is inspired by the human visual cortex, allowing them to effectively recognize patterns and features in images.

04

Common Applications

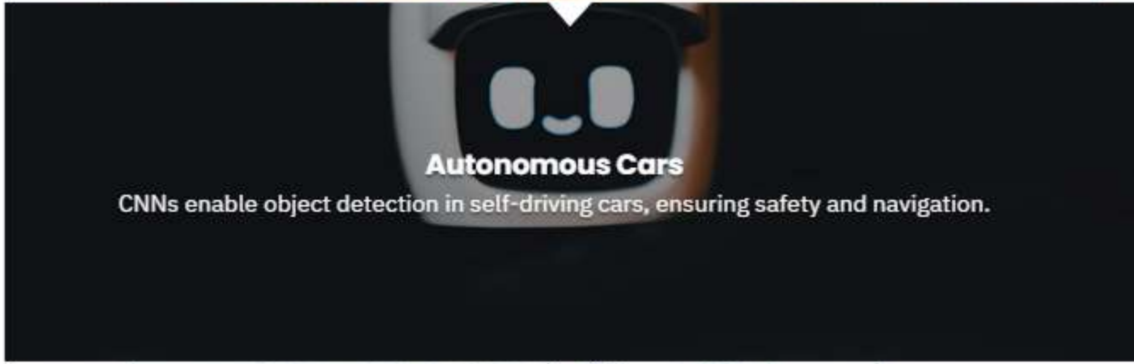
CNNs are widely used in various applications, including facial recognition, object detection, and medical image analysis, due to their high accuracy.

Real-World Applications of CNNs



Face Recognition

CNNs are extensively used in face recognition systems for security and social media applications.



Autonomous Cars

CNNs enable object detection in self-driving cars, ensuring safety and navigation.

Medical Imaging

CNNs assist in analyzing MRI and X-ray images, improving diagnostic accuracy.

Overview of CNN Architecture

Output Layer

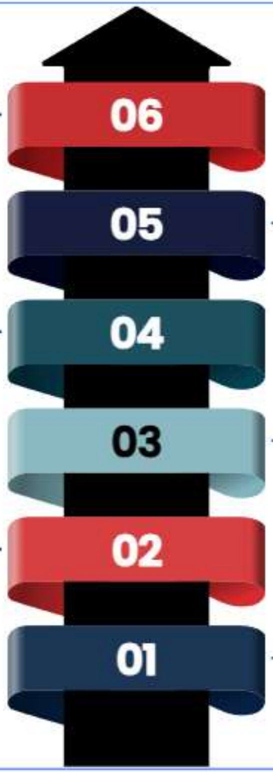
The **output layer** provides the final predictions or classifications based on the processed features from earlier layers.

Pooling Layer

The **pooling layer** reduces the spatial dimensions of the feature maps, maintaining essential information while decreasing computation.

Convolutional Layer

In the **convolutional layer**, filters scan over the input to detect features, producing feature maps.



Fully Connected Layer

In the **fully connected (FC) layer**, the flattened feature maps are connected to output classes, finalizing the decision-making process.

ReLU Activation

The **ReLU activation** function introduces non-linearity by transforming negative values to zero, enhancing learning.

Input Layer

The **input layer** receives the raw data, such as images, to be processed by the CNN.

Understanding Convolution Layers

Feature Extraction Mechanism

The **convolution layer** is essential for extracting features from input images, such as edges or textures, by applying a set of filters that slide across the image.

Activation Maps

The output of the convolution operation is called an activation map, which highlights the presence of specific features detected by the filters.

Sliding Mechanism

The sliding mechanism determines how the filter moves across the image, affecting the size of the output feature map and the amount of overlap.

Local Receptive Fields

Local receptive fields allow the convolution layer to focus on small regions of the input image, making it effective in capturing spatial hierarchies.

Filters and Kernels

Filters, also known as kernels, are small matrices that detect specific features in the image as they move over the input, producing feature maps.

Understanding Convolution Math

01 Convolution Equation

The convolution operation is mathematically defined as $(I*K)(i,j)=\sum m \sum n I(i+m,j+n)\cdot K(m,n)$, where I is the input image and K is the filter kernel. This equation summarizes how each pixel in the image is combined with its neighbors using the kernel.

02 3x3 Image Matrix

In this example, we will use a 3x3 image matrix as input for the convolution operation. Each value in the matrix represents a pixel intensity in the image.

03 2x2 Filter Kernel

A 2x2 filter is applied to the 3x3 image matrix. The filter will slide over the image and perform multiplication and summation to produce an output feature map.

04 Step-by-Step Calculation

This slide includes a table animation that breaks down the convolution steps, showing how the filter interacts with the image matrix at each position, leading to the final output values.

Understanding Padding & Stride

01

Definition of Padding

Padding refers to the addition of pixels around an image to maintain its spatial dimensions during convolution. This is essential when you want to retain the size of the output feature map.

02

Definition of Stride

Stride indicates the number of pixels the filter moves over the input image. A larger stride reduces the spatial dimensions of the output, leading to a more compact feature map.

04

1x1 vs 2x2 Stride Effect

Using a **1x1 stride** means the filter moves one pixel at a time, preserving detail, while a **2x2 stride** skips every other pixel, drastically reducing the output size and potentially losing information.

03

SAME vs VALID Padding

In **SAME** padding, the output size is preserved, while **VALID** padding does not add any pixels, resulting in smaller output dimensions. This impacts the feature extraction capability of the CNN.

Understanding ReLU Activation

Definition of ReLU

The **Rectified Linear Unit (ReLU)** is a widely used **activation function** in neural networks that introduces **non-linearity** into the model, allowing it to learn complex patterns.

Mathematical Equation

The equation for ReLU is given by **f(x) = max(0, x)**. This means that for any input value, the output is either the input itself (if positive) or zero (if negative).

Graphical Comparison

A comparison of the **ReLU** function against the **Sigmoid** function reveals that ReLU does not saturate for large values, which helps in faster training of deep networks.

Feature Map Application

In **Convolutional Neural Networks (CNNs)**, ReLU is applied to the **feature maps**, enhancing the model's ability to capture important features in the data.

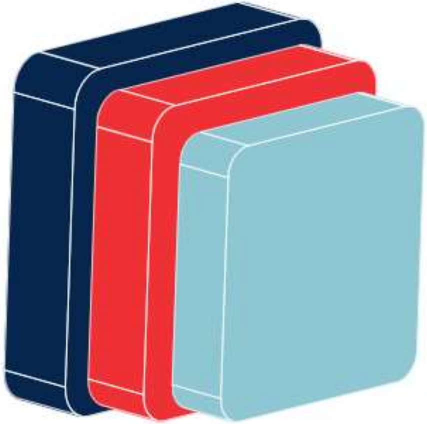
Flatten and Fully Connected Layers

Flatten Layer Functionality

The **Flatten** layer transforms the multi-dimensional output of a convolutional layer into a **1D vector**, making it compatible for subsequent layers. This crucial step prepares the data for classification tasks.

Softmax Activation

The **Softmax** function is applied at the final layer, converting the output scores from the FC layer into probabilities. This ensures the output sums to 1, facilitating multi-class classification.



Fully Connected Layers Role

Fully Connected (FC) layers take the flattened vector and map it to the output classes. Each neuron in an FC layer is connected to every neuron in the previous layer, enabling complex decision boundaries for classification.