Convolutional Neural Nets

Devesh Nath

1 Introduction

Convolutional Neural Networks (CNNs) are a class of deep learning algorithms that have proven highly effective in areas such as image and video recognition, medical image analysis, and natural language processing. They are designed to automatically and adaptively learn spatial hierarchies of features from input data. This introduction provides an overview of the fundamental concepts and architectures of CNNs, highlighting their key components and the principles behind their operation.

1.1 Convolutions

Convolutions are the core building blocks of CNNs. They involve the application of a filter (or kernel) to an input to produce a feature map. This process helps in detecting patterns such as edges, textures, and shapes in the input data.

1.1.1 Padding

Padding is the process of adding extra pixels around the border of the input image. This is done to control the spatial dimensions of the output feature map. Common types of padding include 'valid' (no padding) and 'same' (padding such that the output has the same dimensions as the input).

1.1.2 Strides

Strides refer to the number of pixels by which the filter moves across the input image. A stride of 1 means the filter moves one pixel at a time, while a stride of 2 means it moves two pixels at a time. Larger strides result in smaller output feature maps.

1.1.3 Pooling

Pooling is a down-sampling operation that reduces the spatial dimensions of the feature maps. The most common types of pooling are max pooling, which takes the maximum value in each patch of the feature map, and average pooling, which takes the average value. Pooling helps in reducing the computational complexity and controlling overfitting.

1.1.4 Formula

The output dimensions of a convolutional layer can be calculated using the following formula:

$$W_{out} = \frac{W_{in} - F + 2P}{S} + 1$$

$$H_{out} = \frac{H_{in} - F + 2P}{S} + 1$$

where:

- W_{out} and H_{out} are the width and height of the output feature map.
- W_{in} and H_{in} are the width and height of the input image.
- F is the size of the filter (assumed to be square).
- P is the amount of padding.
- S is the stride.

2 Architectures

2.1 ResNets

Residual Networks (ResNets) are a type of CNN architecture that introduces skip connections or shortcuts to jump over some layers. This helps in addressing the vanishing gradient problem, allowing for the training of much deeper networks. The key idea is to learn residual functions with reference to the layer inputs, instead of learning unreferenced functions.

2.2 MobileNets

MobileNets are a class of efficient models designed for mobile and embedded vision applications. They use depthwise separable convolutions to reduce the number of parameters and computational cost. This makes them suitable for real-time applications on devices with limited computational resources.

2.3 InceptionNet

Inception Networks, also known as GoogLeNet, use a combination of convolutions with different filter sizes within the same layer. This allows the network to capture features at multiple scales. The Inception module concatenates these convolutions along the channel dimension, which helps in improving the network's performance without a significant increase in computational cost.

2.4 YOLO

You Only Look Once (YOLO) is a state-of-the-art, real-time object detection system. Unlike traditional object detection systems that apply the model to an image at multiple locations and scales, YOLO applies a single neural network to the full image. This network divides the image into regions and predicts bounding boxes and probabilities for each region simultaneously.

2.4.1 Architecture

The YOLO architecture consists of a single convolutional network that predicts multiple bounding boxes and class probabilities for those boxes simultaneously. The network divides the input image into an $S \times S$ grid. Each grid cell predicts B bounding boxes, confidence scores for those boxes, and C class probabilities.

- **Grid Cells**: The image is divided into an $S \times S$ grid. Each grid cell is responsible for detecting objects whose center falls within the cell.
- **Bounding Boxes**: Each grid cell predicts B bounding boxes, each with a confidence score. The confidence score reflects the accuracy of the bounding box and whether the bounding box actually contains an object.
- **Class Probabilities**: Each grid cell also predicts C conditional class probabilities, which represent the probability of each class given that an object is present in the cell.

The final predictions are obtained by combining the bounding box predictions with the class probabilities, resulting in a set of detected objects with associated bounding boxes and class labels.

2.5 Transformers in CNN Architectures

Modern architectures have started incorporating transformers, which were originally designed for natural language processing tasks, into CNNs for vision tasks. Vision Transformers (ViTs) treat image patches as sequences and apply transformer encoders to capture global context. This approach has shown promising results in various vision benchmarks, offering an alternative to traditional convolutional architectures.

2.5.1 Vision Transformers (ViTs)

Vision Transformers split an image into fixed-size patches, linearly embed each patch, and then process the sequence of embeddings using transformer encoders. This allows the model to capture long-range dependencies and global context more effectively than traditional CNNs.

2.5.2 Hybrid Architectures

Hybrid architectures combine the strengths of CNNs and transformers. For example, CNNs can be used to extract local features from images, which are then processed by transformer layers to capture global context. This combination leverages the advantages of both convolutional operations and self-attention mechanisms.