Department of Computer Engineering

MET ,BKC, Adgaon, Nashik

Group 1

Assignment No. 3

Title: Convolutional neural network (CNN):

Objective:

Students should be able to apply the technique of Convolutional Neural network for implementing a classifier.

Problem Statement:

Use MNIST Fashion Dataset and create a classifier to classify fashion clothing into categories.

Theory:

A Convolutional Neural Network, also known as CNN or ConvNet, is a class of neural networks that specializes in processing data that has a grid-like topology, such as an image. A digital image is a binary representation of visual data. It contains a series of pixels arranged in a grid-like fashion that contains pixel values to denote how bright and what color each pixel should be. The human brain processes a huge amount of information the second we see an image. Each neuron works in its own receptive field and is connected to other neurons in a way that they cover the entire visual field. Just as each neuron responds to stimuli only in the restricted region of the visual field called the receptive field in the biological vision system, each neuron in a CNN processes data only in its receptive field as well. The layers are arranged in such a way so that they detect simpler patterns first (lines, curves, etc.) and more complex patterns (faces, objects, etc.) further along. By using a CNN, one can enable sight to computers.

Convolutional Neural Network Architecture

A CNN typically has three layers: a convolutional layer, a pooling layer, and a fully connected layer.

Convolution Layer:

The convolution layer is the core building block of the CNN. It carries the main portion of the network's computational load. This layer performs a dot product between two matrices, where one matrix is the set of learnable parameters otherwise known as a kernel, and the other matrix is the restricted portion of the receptive field. The kernel is spatially smaller than an image but

is more in-depth. This means that, if the image is composed of three (RGB) channels, the kernel height and width will be spatially small, but the depth extends up to all three channels. During the forward pass, the kernel slides across the height and width of the image-producing the image representation of that receptive region. This produces a two-dimensional representation of the image known as an activation map that gives the response of the kernel at each spatial position of the image. The sliding size of the kernel is called a stride.

If we have an input of size W x W x D and Dout number of kernels with a spatial size of F with stride S and amount of padding P, then the size of output volume can be determined by the following formula:

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

Pooling Layer:

The pooling layer replaces the output of the network at certain locations by deriving a summary statistic of the nearby outputs. This helps in reducing the spatial size of the representation, which decreases the required amount of computation and weights. The pooling operation is processed on every slice of the representation individually. There are several pooling functions such as the average of the rectangular neighborhood, L2 norm of the rectangular neighborhood, and a weighted average based on the distance from the central pixel. However, the most popular process is max pooling, which reports the maximum output from the neighborhood.

If we have an activation map of size W x W x D, a pooling kernel of spatial size F, and stride S, then the size of output volume can be determined by the following formula:

$$W_{out} = \frac{W - F}{S} + 1$$

Fully Connected Layer

Neurons in this layer have full connectivity with all neurons in the preceding and succeeding layer as seen in regular FCNN. This is why it can be computed as usual by a matrix multiplication followed by a bias effect. The FC layer helps to map the representation between the input and the output.

Non-Linearity Layers

Since convolution is a linear operation and images are far from linear, non-linearity layers are often placed directly after the convolutional layer to introduce non-linearity to the activation map.

There are several types of non-linear operations, the popular ones being:

1. Sigmoid

The sigmoid non-linearity has the mathematical form $\sigma(\kappa) = 1/(1+e^{-}\kappa)$. It takes a real-valued number and "squashes" it into a range between 0 and 1. However, a very undesirable property of sigmoid is that when the activation is at either tail, the gradient becomes almost zero.

2. Tanh

Tanh squashes a real-valued number to the range [-1, 1]. Like sigmoid, the activation saturates, but — unlike the sigmoid neurons — its output is zero centered.

3. ReLU

The Rectified Linear Unit (ReLU) has become very popular in the last few years. It computes the function $f(\kappa)=\max(0,\kappa)$. In other words, the activation is simply threshold at zero. In comparison to sigmoid and tanh, ReLU is more reliable and accelerates the convergence by six times. Unfortunately, a con is that ReLU can be fragile during training. A large gradient flowing through it can update it in such a way that the neuron will never get further updated. However, we can work with this by setting a proper learning rate.

Designing a Convolutional Neural Network

Now that we understand the various components, we can build a convolutional neural network. We will be using Fashion-MNIST, which is a dataset of Zalando's article images consisting of a training set of 60,000 examples and a test set of 10,000 examples. Each example is a 28x28 grayscale image, associated with a label from 10 classes.

Our convolutional neural network has architecture as follows:

[INPUT]

→[CONV 1] → [BATCH NORM] → [ReLU] → [POOL 1]

→ [CONV 2] → [BATCH NORM] → [ReLU] → [POOL 2]

→ [FC LAYER] → [RESULT]

For both conv layers, we will use kernel of spatial size 5 x 5 with stride size 1 and padding of 2. For both pooling layers, we will use max pool operation with kernel size 2, stride 2, and zero padding.

CONV 1
Input Size (W_1 x H_1 x D_1) = 28 x 28 x 1
• Requires four hyperparameter: ○ Number of kernels, k = 16 ○ Spatial extend of each one, F = 5 ○ Stride Size, S = 1 ○ Amount of zero padding, P = 2
• Outputting volume of W_2 x H_2 x D_2 ○ W_2 = (28 – 5 + 2 (2)) / 1 + 1 = 28 ○ H_2 = (28 – 5 + 2 (2)) / 1 + 1 = 28 ○ D_2 = k
Output of Conv 1 (W_2 x H_2 x D_2) = 28 x 28 x 16

POOL 1
Input Size (W_2 x H_2 x D_2) = 28 x 28 x 16
• Requires two hyperparameter: ○ Spatial extend of each one, F = 2 ○ Stride Size, S = 2
• Outputting volume of W_3 x H_3 x D_2 ○ W_3 = (28 – 2) / 2 + 1 = 14 ○ H_3 = (28 – 2) / 2 + 1 = 14
Output of Pool 1 (W_3 x H_3 x D_2) = 14 x 14 x 16

CONV 2
Input Size (W_3 x H_3 x D_2) = 14 x 14 x 16
• Requires four hyperparameter: o Number of kernels, k = 32 o Spatial extend of each one, F = 5 o Stride Size, S = 1 o Amount of zero padding, P = 2
• Outputting volume of W_4 x H_4 x D_3 o W_4 = (14 – 5 + 2 (2)) / 1 + 1 = 14 o H_4 = (14 – 5 + 2 (2)) / 1 + 1 = 14 o D_3 = k
Output of Conv 2 (W_4 x H_4 x D_3) = 14 x 14 x 32

POOL 2
Input Size (W_4 x H_4 x D_3) = 14 x 14 x 32
• Requires two hyperparameter: ○ Spatial extend of each one, F = 2 ○ Stride Size, S = 2
• Outputting volume of W_5 x H_5 x D_3 ○ W_5 = (14 – 2) / 2 + 1 = 7 ○ H_5 = (14 – 2) / 2 + 1 = 7
Output of Pool 2 (W_5 x H_5 x D_3) = 7 x 7 x 32

FC Layer
Input Size (W_5 x H_5 x D_3) = 7 x 7 x 32
Output Size (Number of Classes) = 10

Conclusion: In this way we learn to implement MNIST Fashion Dataset and create a classifier to classify fashion clothing into categories.

Assignment Question

1. What is Convolutional Neural Network?
2. What are different types of CNN models?

3. What are the applications of CNN?

4. Describe all layers of CNN.