Creating Face Filter Application

Mehmet Onur Aysel, Cafer Yahşi
Bilgisayar Mühendisliği Bölümü
Yıldız Teknik Üniversitesi, 34220 Istanbul, Türkiye
onur.aysel@std.yildiz.edu.tr
caferyi33@gmail.com

***Özetçe* —Bu belge, herhangi bir sosyal medya platformunda bulunan bir filtrenin nasıl çalıştığı, bu filtrelerin fotoğrafta ağzımızın veya burnumuzun yerini nasıl tespit ettiği ve bu uzuvlara nasıl filtre uyguladığı ve evrişimsel sinir ağları hakkında bilgiler içermektedir.**

Anahtar Kelimeler—evrişimsel sinir ağları, yapay sinir ağları, makine öğrenmesi, aktivasyon fonksiyonu, kayıp fonksiyonu, katman, çok katmanlı yapı, nöron, batch normalizasyonu, öznitelik tanımlama, ReLU, sigmoid fonksiyon, tanh fonksiyonu, softmax fonksiyonu, ortaklama, maksimum ortaklama, düzenleştirme, düzleştirme çıkarma(atma), padding işlemi

***Abstract*—This project includes such informations about how filtering process works which are in social media, how these filters can automatically detect face landmarks at works on these face parts such as mouth, nose and ear. It also applies face filtering process witj simple convolutional neural network model and by real time on image which comes from camera.**

Keywords—convolutional neural networks, artificial neural networks, activation function, loss function, error rate, layer,neuron, batch normalization, feature description, relu, sigmoid function, tanh function, softmax function, pooling, max pooling, regularization, flattening, dropout, padding

I. Introduction

II. CNN

Convolutional Neural Network is an algorithm that can detect unique aspects of a given picture. When you are shown a dog picture, you immediately compare your dog image to that picture and you understand the picture given is a dog picture. Convolutional Neural Network works like that aswell. The system that you uploaded the picture to creates an image of a dog. But this image is not like we create in our minds. The system creates something that we can't physically see like convex or concav shapes. System divides into layers to be able to do these identifications.

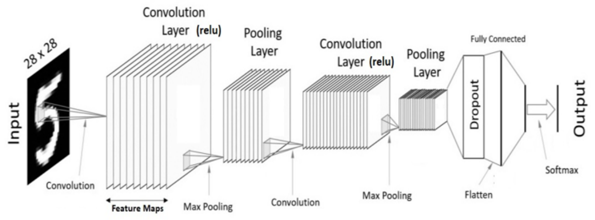

Figure 1 CNN

III. Convolution Layer

This layer is the main layer the algorithm takes it's name from. Also we need to decide what shape the picture would be. The main purpose of this layer is determining the shapes like convex or concav that we can't see. This procesess' name is Convolution Operation where it also got it's name from the layer itself. This operation is achieved by moving 3x3,5x5,7x7 matrices, the number of which varies, usually consisting of odd numbers, over the pixel values of the photograph. The problem here is that after the filter is applied, the size of the input variable decreases. To solve this problem, padding is done. Padding is the process of wrapping the pixel values of the photo like a frame before the convolution process is applied. The most used padding method is Same Padding and a frame of 0s is used.

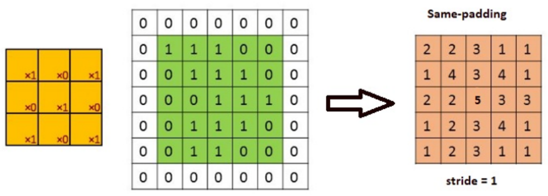

Figure 2 max padding

IV. Non - Linearity Layer

Another problem in neural networks is linearity. Usually, activation functions are used to solve this problem. These functions are functions that compress the values they receive between certain values. There are many activation functions such as relu, tanh, sigmoid, softmax. In general, the Rectifier(ReLu) function is used because complex structures are detected in the photos and it works more optimized.

V. Pooling(Downsampling) Layer

The main purpose of the pooling layer is to balance the number of parameters and solve the problem of overfitting the dataset of our model. In general, a Pooling layer is added to the model after each Convolutional layer. The pooling layer usually uses down-sapling or subsubling when solving these problems. Although the most used type is max pooling, there are also different types of pooling such as average, L2-norm pooling. Although 2x2 filters are generally used, this may change depending on the size of the data set. In max pooling, the maximum value in that frame is found according to the selected frame and only that number is taken for those 4 frames. Thus, the number of samples is reduced, that is, we make sub-sampling.

VI. Flattening Layer

In fact, every Convolutional Neural Network or Convolutional neural network structure model has an Artificial neural network structure at the end. Artificial neural network structure accepts only one-dimensional matrices as input, so after adding enough Convolutional, Pooling, Non-linearity layers, flattening should be done before entering the Artificial neural network structure. Flattening is 1x? is the process of transforming it into shape.

VII. Dropout

Although dropout is not counted as a layer, it is done after each layer. Dropout is the process of removing neurons below the specified value from the system. Its purpose is to reduce overfitting, ie memorizing the dataset, in the system.

VIII. Batch Normalization

Batch normalization allows each layer to learn at the same time, rather than the layers learning sequentially. In this way, it increases the training speed of the model and also prevents the gradients from disappearing in models with high learning rates. It makes the model more stable. It sets the mean of the distribution in the model to be 0 and the standard deviation to be 1. all values are compressed between -1 and 1. [1]

IX. Implementation

Our aim in the project is to apply one of the 8 face filters determined by your algorithm on the computer while our camera is open on the computer and to make it move with your face when you move. We will use the Convolutional Neural Network algorithm to perform this operation. First of all, let's get to know the dataset we will use to train our neural network. It consists of seven thousand black and white photographs of human faces. In addition, since our aim is to determine the landmarks on the human face, there are 15 points marked for each human face. Since each point consists of x and y axis, a total of 30 data are kept for each photograph. However, 30 points of all photos are not marked in total, so we will use the data marked right eye center, left eye center, nose tip and middle of the mouth, where we can apply a filter that will work for us, both because some points do not work for us and due to the missing data in some photos.

After selecting the data we will use, a variable with 8 columns holding 2 coordinates between 0-96 for each variable such as the values of the photograph in one variable, the center of the right eye, the center of the left eye, the tip of the nose and the middle of the mouth of that photograph in the other variable. We divide our dataset into 2 parts, 20% of which are testing and 80% are training. We first set our model to be Convolutional layer, Batch Normalization, Pooling and Dropout, repeating 4 times. In

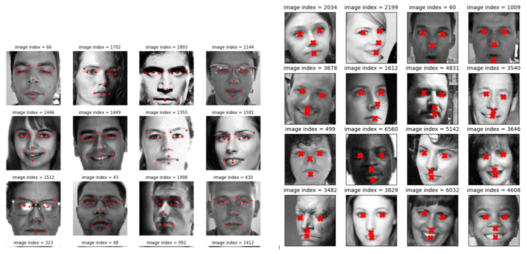
Figure 3 True Marks on Data

the Convolutional layer, our filter number usually changes to 32 or 64, we use the most used Max pooling algorithm as the padding algorithm. Since our photos are 96x96, our input shape is 96x96. As the activation function, we use the ReLU function as it is used more in complex systems. We used Batch Normalization because it solves the problem of simultaneous learning in the system, increasing the learning speed of the system and overfitting at the same time. We used ADAM, Adaptive Moment Estimation, as the optimizer. The reason why we can use ADAM optimizer is that it can increase or decrease the learning rate according to the course of the model.

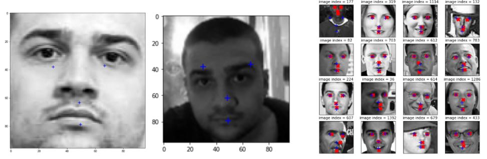
Figure 4 True Marks on Data

After creating our model, we use the data we reserved for testing. We cannot give an accuracy percentage because there is no clear class in the results. Therefore, we calculated the distance between the coordinates we found and the coordinates marked in the system using the Euclidean distance metric. We created a total of 4 graphs for each variable. In order to see how optimized it is in real life, we had our model make predictions for 2 photographs. Even though it is not highly optimized, the results are at a level that can be called good.

A. Performance

References

[1] J. Brownlee. (2019) A gentle introduction to batch normalization for deep neural networks. [Online]. Available: https://machinelearningmastery.com/batch-normalization-for-training-of-deep-neural-networks/

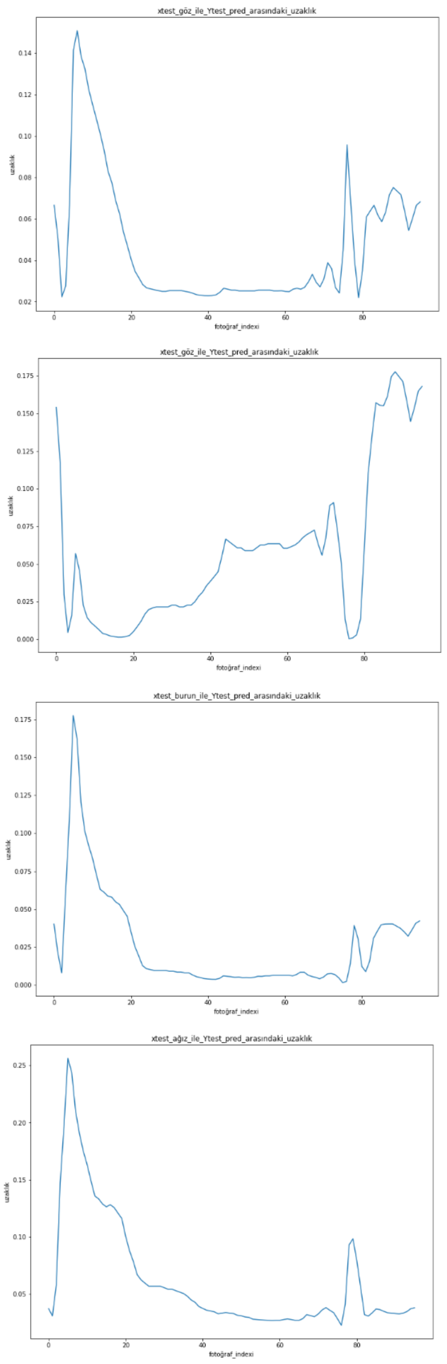

Figure 5 performance