Lecture Notes On Objective Function for CNNs (Part II)

Harry Li ‡, Ph.D.
Computer Engineering Department
College of Engineering, San Jose State University
San Jose, CA 95192, USA

Research and Development Center for Artificial Intelligence and Automation ‡
CTI Plus Corporation, 3679 Enochs Street
Santa Clara, CA 95051, USA

Email†: hua.li@sjsu.edu

***Abstract*—This note is part II from my lecture on convolutional neural networks. In particular, the techniques for objective functions for facenet and ResNet CNNs for facial recognitions.**

I. Objecgtive Functions

In this note, mathematical formulation of objective function is discussed below.

First, consider input neuron $\overrightarrow{\sigma}$ and output neuron $\overrightarrow{S}$, so we have an activation function connecting one input neuron to one output neuron

$$s_i = f(w_{i,k} \cdot \sigma_k) \tag{1}$$

Hence, for all input neurons connecting to one output neuron, we we have

$$s_i = f(\sum_{i=1}^{n_o} w_{i,k} \cdot s_k + \phi) = f(\sum_{i=1}^{n_o+1} w_{i,k} \cdot s_k) \tag{2}$$

Where the upper bound of the summation Note in case of convolutional NN, the nerons σ_k in the above equation is the output of 2D convolution layer. Denote 2D convolution as

$$\sigma_k = g_{\sigma_k}(l_i(x,y) * K(x,y); k_{u,v}) \tag{3}$$

where $l_i(x,y)$ is convolution layer i, $K(x,y)$ is a convolution kernel, whose coefficients are u and v such as $((u,v) \in \omega)$, e.g., in a k-by-k patch, k are odd numbers.

Therefore, for one input and one output neuron, we have

$$s_i = f(w_{i,k} \cdot g_{\sigma_k}(l_i(x,y) * K(x,y); k_{u,v} | u,v \in \omega)) \tag{4}$$

and for all input neurons n to one output neural, we have

$$s_i = f(\sum_{i=1}^{n+1} w_{i,k} \cdot g_{\sigma_k}(l_i(x,y) * K(x,y); k_{u,v} | u,v \in \omega)) \tag{5}$$

or in a simplified notation,

$$s_i = f(\sum_{i=1}^{n+1} w_{i,k} \cdot g_{\sigma_k}(l_i * K)) \tag{6}$$

With definition of a transfer function to handle the input neurons σ_k, we have

$$h_i = \sum_{i=1}^{n} w_{i,k}\sigma_k + \phi = \sum_{i=1}^{n+1} w_{i,k}\sigma_k \tag{7}$$

So the neuron output i is

$$s_i = f(h_i) = f(h_i(w_{i,k} \cdot \sigma_k)) \tag{8}$$

For CNN with convolution layer(s), we have

$$s_i = f(h_i) = f(h_i(w_{i,k} \cdot g_{\sigma_k}(l_i * K))) \tag{9}$$

Since l_i convolution data layer, we drop it from the notation and keep the kernel K in the notation for formulation of training. So,

$$s_i = f(h_i) = f(h_i(w_{i,k} \cdot g_{\sigma_k}(K))) \tag{10}$$

In Note I, we define an error at each neuron output for the experiment μ as

$$\zeta_i{}^{\mu} - s_i{}^{\mu} \tag{11}$$

where $\zeta_i{}^{\mu}$ (pronouced as 'zeta') is a desired (correct) output i at experiment μ, and $s_i{}^{\mu}$ is the actual output i at that experiment. In the context of our discussion in FaceNet, both $\zeta_i{}^{\mu}$ and $s_i{}^{\mu}$ are the embeddings of the CNN output.

Now, for CNN, the error function above can be written as

$$\zeta_i{}^{\mu} - f^{\mu}(h_i(w_{i,k} \cdot g_{\sigma_k}(K))) \tag{12}$$

where both $w_{i,k}$ and K have to be learned in the training of CNN.

In Note I, we have defined total error for all neuron outputs n_o, and for all the experiments *m*.

$$D = \frac{1}{2}\sum_{\mu=1}^{m}\sum_{i=1}^{n_o}(\zeta_i{}^{\mu} - s_i{}^{\mu})^2 \tag{13}$$

In case of CNN, we will have the above equation to count the convolution kernels, so we have

$$D = \frac{1}{2}\sum_{\mu=1}^{M}\sum_{i=1}^{n_o}(\zeta_i{}^{\mu} - f^{\mu}(h_i(w_{i,k}\cdot g_{\sigma_k}(K))))^2 \tag{14}$$

Now, re-write the above equation by introducing l_2 mode, we can drop the scalong factor 1 by 2 in front of the equation without affecting the final outcome of the training. So,

$$D = \sum_{\mu=1}^{M}\sum_{i=1}^{n_o}||(\zeta_i{}^{\mu} - f^{\mu}(h_i(w_{i,k}\cdot g_{\sigma_k}(K))))||_2^2 \tag{15}$$

II. Objective Function in Facical Recognition

Now, we will discuss the objective function in Face Net. First, let's define an image x as an input layer, and the FaceNet CNN output for the image feature extraction as function f. So, we have???

Obviously, the above objecitve function is also a loss function.

Denote all output neurons errors from i equal to 1, 2, ..., n_o as

$$D = \sum_{\mu=1}^{M}||(\zeta^{\mu} - f^{\mu}(h(l)))||_2^2 \tag{16}$$

$$= \sum_{\mu=1}^{M}||(\zeta^{\mu} - f^{\mu}(l))||_2^2 \tag{17}$$

where ζ_i^{μ} for i = 1,2, ..., n_o, for the errors at each individual dimenson, is replaced by the summation of all the errors at the experiment μ, e.g., we denote

$$\zeta^{\mu} = \sum_{i=1}^{n_o}||(\zeta_i{}^{\mu} - f^{\mu}(h(l))||_2^2 \tag{18}$$

In order to make comparison with Google team's mathematical notation, let input image layer l

$$l_i(u, v) = x_i \tag{19}$$

for images of i=1,2, ..., n. And $x_i \in R^3$, the outpout of CNN of the faceNet defines feature vectors, e.g., embeddgings of the input image. The outpout is denoted as

$$f : R^3 \to R^N \tag{20}$$

for N=68 facial land marks, see google facenet reference and Sandburg github repo (detector_dlib.py). Using this notation to rewrite the objective function above,

$$D = \sum_{\mu=1}^{M}||(\zeta^{\mu}(x) - f^{\mu}(x))||_2^2 \tag{21}$$

Replacing index μ by i, we now have

$$D = \sum_{i=1}^{M}||(\zeta(x_i) - f(x_i))||_2^2 \tag{22}$$

Or, simply, let $\zeta(x_i) = f_{true}(x_i)$, we have

$$D = \sum_{i=1}^{M}||(f_{true}(x_i) - f(x_i))||_2^2 \tag{23}$$

Lemma 1. ***Classes from the image data set.*** Suppose there are P classes of faces (persons), where each class of the images counts for $M_1, M_2, ..., M_i, ...M_P$, so the total number of images involved in the training for P classes is

$$M = \sum_{i=1}^{P} M_i \tag{24}$$

The recognition of a person from class i can be formulated as a training process of class i against the rest of all other classes. We call these rest of all other classes as the negative class and it is

$$M_n = \sum_{j=1, j\neq i}^{P} M_j \tag{25}$$

Lemma 2. ***The recognition of class i.*** For the recognition of class i from the rest of all other classes j out of P classes, we have objective function as

$$\begin{aligned} D \quad &= \sum_{j=1}^{M_1}||(f_{true}(x_j) - f(x_j))||_2^2 + \\ &\sum_{j=M_1+1}^{M_2}||(f_{true}(x_j) - f(x_j))||_2^2 + ... \\ &... + \sum_{j=M_{P-1}+1}^{M_P}||(f_{true}(x_j) - f(x_j))||_2^2. \end{aligned} \tag{27}$$

Definition 1. ***P-classes objective function.*** P classes objective function takes the form of

$$\begin{aligned} D \quad &= \alpha_1\sum_{j=1}^{M_1}||(f_{true}(x_j) - f(x_j))||_2^2 + \\ &\alpha_2\sum_{j=M_1+1}^{M_2}||(f_{true}(x_j) - f(x_j))||_2^2 + ... \\ &... + \alpha_p\sum_{j=M_{P-1}+1}^{M_P}||(f_{true}(x_j) - f(x_j))||_2^2. \end{aligned} \tag{29}$$

where

$$\sum_{i=1}^{p}\alpha_i = 1. \tag{30}$$

(END)

References

[1] Harry Li, Lecture notes on Back Propagation Techniques, https://github.com/hualili/opencv/blob/master/deep-learning-2020S/10-2020F-106a-back-prop3-hl-2020-10-1.pdf

[2] David Sandburg, *github repo of FaceNet python code*: https://github.com/davidsandberg/facenet, 2018.