Report of Machine Learning Project

Zhixin Ling
516021910495
Shanghai Jiao Tong University
Shanghai, China
1069066484@qq.com

Yifeng Gao
0000000000
Shanghai Jiao Tong University
Shanghai, China
gaoyifengsmail

I. Main Ideas

In this project we use deep methods to perform classification work on Fer2013 dataset. We classify people's face images into seven categories: Angry, Disgust, Fear, Happy, Sad, Surprise and Neutral. We try different network architectures and select network parameters that result in the best accuracy on the given public test set. Finally, we test our networks and parameters on the given private test set to investigate the performance. Also, we try cropping the face segment of the images first to eliminate some unnecessary noises to see whether the classification can improve.

We find networks of ResNet18 and VGG19 achieve the best performance on the given private test dataset with the selected parameters without face cropping. The best resulted accuracy reaches 0.73, reaching the state-of-art achievement. With face cropping, the overall resulted accuracies reduce by about 0.03. However, on several classes, the resulted classification accuracies improve greater than 0.02.

II. Methods

A. LeNet

LeNet is a classical neural network. It is the first complete proposed neural network architecture, including convolution layers, pooling layers, fully connected layers and neuron activation mechanism. The network is simple but can be very effective for some requirements. LeNet's training needs far less computation compared with some later complicate networks. So, LeNet can always worth a try in our classification work. Compared with the first proposed version of LeNet, we use RELU instead of sigmoid function for neuron activation and also apply batch normalization to ensure a fast and stable convergence.

B. VGG

VGG, short for Visual Geometry Group Network, is far deeper than LeNet. More convolution kernels are applied; methods of neuron dropout and batch normalization are introduced; larger fully connected layers are used. We try several architectures of VGG of different depths to explore whether a deeper network brings a better classification performance.

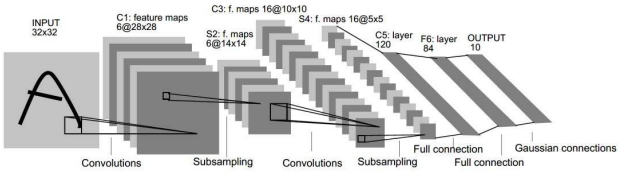

Fig. 1. Theory of SA

C. ResNet

ResNet, short for residual neural network, won the first prize in ILSRVC2015. The special feature of ResNet is that it creatively proposed shortcut connections that allow a former layer to have additional connections, bypassing adjacent layers, leading to the layers behind. The idea greatly ease decreasing of accuracies caused by increasing of network depths, thus allowing the network as deep as possible. As a result, ResNet have far more convolution layers than VGG and also bigger convolution kernels.

D. Face Detection And Cropping

The samples of our dataset are not all well collected. Faces in some images may be misplaced in a corner and the main contents of the image can be something uncorrelated to the classification work. So we try to detect the area in a image where the face exists and crop the area as a sample. The old image is discarded, of course. For the work of face detection, we use traditional methods. We use a trained classifier cascaded from strong subclassifiers trained with extracted Haar features of images using Adaboost algorithms.

E. Data Augmentation

Data augmentation is used to enlarge the training dataset, creating more training samples, to make the trained model more robust and ease overfitting. Symmetry and random cropping are used for data augmentation.

III. Algorithms

A. Preprocessing

Before training, we need to preprocess our data. This includes face detection and cropping, if required, and data augmentation. The algorithm is elaborated in 1.

Algorithm 1: Preprocess

Input:

An $b \times m$ matrix B, a batch of raw data of size b.

F_{DC}, boolean, whether or not to detect and crop the face part of the image.

l', side length of output image.

Output:

A $b \times l' \times l'$ matrix X

for *Every row R in B* **do**
  /* Reshape R to be an image. */
  Reshape R to be an $l \times l$ matrix, where $l = \sqrt{m}$;
  if F_{DC} *is true* **then**
    Detect the area where the face exists in R;
    Discard the rest area of R;
    Resize R to be $l \times l$;
  /* Perform data augmentation. */
  Randomly select a $l' \times l'$ area in R;
  $R \leftarrow$ the selected part;
  Randomly decide whether or not to flip the image horizontally with a chance of 0.5;
$X \leftarrow B$;
return X;

B. Network Training

Because of hardware limitations, we train our model in batches. And learning rate annealing is also applied to stabilize the convergence. After each epoch, we estimate the classification accuracy on the public test set and decide whether or not to save the current network parameters. The algorithm is elaborated in 2.

IV. Experimental Settings

Before going on, we present our common notations to avoid confusion in table I. The middle column Fixed Value indicates we have a fixed value for the parameter and we use $-$ to indicate the value is not fixed.

V. Results

Results.

Algorithm 2: Network Training

Input:

(X^{tr}, Y^{tr}), the training set.

$(X^{te}_{pub}, Y^{te}_{pub})$, the public test set.

F_{DC}, boolean, whether or not to detect and crop the face part of the image.

θ, parameters of the given network.

Ep_m, the maximum epoch.

l', side length of image for training.

Output:

$\theta*$, trained parameters of the given network.

Initialize the θ and $\theta^* \leftarrow \theta$;
Initialize learning rate l_r;
$acc^{te*}_{pub} \leftarrow 0.0$;
for $e = 1 \rightarrow Ep_m$ **do**
  for *Each batch i in* (X^{tr}, Y^{tr}) **do**
    Select a batch (X^{tr}_i, Y^{tr}_i) from (X^{tr}, Y^{tr});
    $Preprocess(X^{tr}_i, F_{DC}, l')$;
    Calculate array q with $q_j \leftarrow f(\theta, X^{tr}_{i,j})$, where $j = 1, 2, \cdots, b$ and f denotes the function corresponding to the given network;
    $q \leftarrow softmax(q)$;
    $loss \leftarrow -\frac{1}{b}\sum_{j=1}^{b} Y^{tr}_{i,j} \log q_j$;
    $\theta \leftarrow -l_r \times \frac{\partial loss}{\partial \theta}$;
  Calculate q for all batches of $(X^{te}_{pub}, Y^{te}_{pub})$;
  $acc^{te}_{pub} \leftarrow \frac{1}{s^{te}_{pub}} \sum_j argmax(q_j) == Y^{te}_{pub,j}$, where s^{te}_{pub} is total size of public training set;
  if $acc^{te}_{pub} > acc^{te*}_{pub}$ **then**
    $acc^{te*}_{pub} \leftarrow acc^{te}_{pub}$;
    $\theta^* \leftarrow \theta$;
  Anneal l_r;
return θ^*;

TABLE I
CONVENTION OF NOTATIONS

Notation	Fixed Value	Description
l	48	side length of raw input images.
l'	44	cropped side length of input images.
F_{DC}	-	whether or not to detect and crop the face part of the image.
D_o	0.5	Neurons' drop out rate when training(if a neuron is allowed so).
Ep_m	250	Maximum number of epochs.
$l_r(e)$	$0.01 \times 0.9^{e/5}$	Learning rate in a given epoch e.
b	-	Batch size.
$S_{tr} = (X^{tr}, Y^{tr})$	✓	Training set.
$S_{pub} = (X^{te}_{pub}, Y^{te}_{pub})$	✓	Public test set.
$S_{pri} = (X^{te}_{pri}, Y^{te}_{pri})$	✓	Private test set.
$acc^{tr}(e)$	-	Training accuracy in a given epoch e.
$acc^{te}_{pub}(e)$	-	Private test accuracy in a given epoch e.
$acc^{te}_{pri}(e)$	-	Public test accuracy in a given epoch e.