Facial Keypoints Detection Using Convolutional Neutral Network

Qinyao He
CST34
Tsinghua University
2012010548

Xiaocheng Yang
CST34
Tsinghua University
2013XXXXXX

Abstract

In this report we building an neural network for carrying out facial keypoints detection problem. We have tried several ways, but a problem of non convergence still need to be resolved, which till now we have no idea about that.

1. Introduction

This task comes originally from Kaggle[1], a online website holding many data science competitions. It's aim is to determine the location of several certain key points on face, given a photograph contains a human face.

This key points detection can form the building block for many applications. Such as:

* tracking faces in images and video
* analysing facial expressions
* detecting dysmorphic facial signs for medical diagnosis
* biometrics / face recognition

Detecing facial keypoints is a very challenging problem. Facial features vary greatly from one individual to another, and even for a single individual, there is a large amount of variation due to 3D pose, size, position, viewing angle, and illumination conditions. Computer vision research has come a long way in addressing these difficulties, but there remain many opportunities for improvement.

In this specific competition[2], we are required to localize 15 facial key points given a images. Each predicted point is specified as (x, y) real-valued pair in the space of pixel indices. The 15 keypoints is:

left_eye_center, right_eye_center, left_eye_inner_corner, left_eye_outer_corner, right_eye_inner_corner, right_eye_outer_corner, left_eyebrow_inner_end, left_eyebrow_outer_end, right_eyebrow_inner_end, right_eyebrow_outer_end, nose_tip, mouth_left_corner, mouth_right_corner, mouth_center_top_lip, mouth_center_bottom_lip

so 30 real value is going to be predicted for each input image.

Kaggle given a training data set of 7049 image with size of $96 * 96$, color is gray scale and represented as an integer from 0 to 255 for each pixel. And each training images is labeled with 30 real value for position of the 15 key points. In some examples, some of the target keypoint positions are misssing.

Also, a test set of 1783 images without label is given, for competitors to submit their predicted result to Kaggle for score.

Performance is graded by MSE(mean square error) for those 30 real values. A lower MSE indicate a better prediction.

In our work, we have carried out experiment on several models, include basic multilayer perceptron, convolutional neural network and cascaded convolutional network.

2. Related Work

3. Technical Approach

In this section we demonstrate some technical approach which, based on previous work, can improve performance of the network, accelerate convergence, and reduce overfitting.

3.1. Data Normalization

In the original training data, each pixel is represented by an integer from 0 to 255, and the real-value label range from 0 to 96. This large range of input and target data may force the neuron to saturate, which slow down convergence. Also, the non-error-centered target value make the full connected layer hard to get a ideal output.

In our implementation, gray scale from 0 to 255 are divided by 256, rescaled to range 0 to 1, and target value are rescaled and zero centered to range from -1 to 1. Let train_x,

[1] https://www.kaggle.com/

[2] https://www.kaggle.com/c/facial-keypoints-detection

train_y stand for the image data and label, the normalize preprocess looks like:

$$train\_x = \frac{train\_x}{256.0}$$
$$train\_y = \frac{train\_y - 48.0}{48.0}$$

3.2. ReLU Nonlinearity

When training deep feedforward neural network, gradient diminishing when back propagation often occurs and make it very slow to converge. This issue is common when using sigmoid-like activation function, since they have a so-called saturation regime where gradient almost vanished. According to experiment by Glorot [**?**], for a feedforward neural net with 4 layers and activate with sigmoid function, the last hidden layer quickly saturate with output value of 0, which slow down the process of learning. and after a hundred of epochs, it escaped, and the followed layer began to saturate.

Glorot hypothesize that this behavior is the result of combination of random initialization and the fact that an output of zero correspond to a saturated sigmoid. Unsupervised pre-training can successfully solve this issue.

3.3. Weight Initialization

3.4. Dropout

4. Network Structure

5. Experiment

6. Conclusion

References