A Real-time Facial Expression Recognizer using Deep Neural Network

Jinwoo Jeon
Korea Advanced Institute of Science and Technology
291, Daehak-ro, Yuseong-gu, Daejeon, Korea
82-42-350-8172
jjw0514@kaist.ac.kr

Jun-Cheol Park
Korea Advanced Institute of Science and Technology
291, Daehak-ro, Yuseong-gu, Daejeon, Korea
82-42-350-8172
pakjce@kaist.ac.kr

YoungJoo Jo
Korea Advanced Institute of Science and Technology
291, Daehak-ro, Yuseong-gu, Daejeon, Korea
82-42-350-8172
ghjk7979@kaist.ac.kr

ChangMo Nam
Korea Advanced Institute of Science and Technology
291, Daehak-ro, Yuseong-gu, Daejeon, Korea
82-42-350-8172
ncm4114@kaist.ac.kr

Kyung-Hoon Bae
SK Telecom Future R&D Center
SK T-Tower, Jung-gu, Seoul
82-2-6100-1643
vc.lab@sk.com

Youngkyoo Hwang
SK Telecom Future R&D Center
SK T-Tower, Jung-gu, Seoul
82-2-6100-1643
youngkyoo.hwang@sk.com

Dae-Shik Kim
Korea Advanced Institute of Science and Technology
291, Daehak-ro, Yuseong-gu, Daejeon, Korea
82-42-350-3490
daeshik@kaist.ac.kr

ABSTRACT

With increasing boundaries of deep learning based recognition model, particular demands for real-time user state recognizer targeting smart home devices have emerged. This paper suggests real-time facial expression recognizer to satisfy them. We use HOG feature descriptor to detect a human face, correlation tracker to track detected face and deep Convolutional Neural Network (CNN) based recognizer on our model. Our CNN model is trained and tested with Kaggle facial expression recognition challenge dataset. The experimental result shows that high test accuracy and low computation time are achieved by our recognizer enabling real-time high-performance human expression recognition for mobile use.

Categories and Subject Descriptors

I.5.4 [**Pattern Recognition**]: Application – c*omputer vision.*

General Terms

Algorithms.

Keywords

Deep Learning, Machine Learning, Neural Network, Convolutional Neural Network, CNN, Real-time Recognition, Facial Expression Recognition, Computer Vision, Pattern Recognition

IMCOM '16, January 04-06, 2016, Danang, Viet Nam
© 2016 ACM. ISBN 978-1-4503-4142-4/16/01…$15.00
DOI: http://dx.doi.org/10.1145/2857546.2857642

1. INTRODUCTION

Facial expression is salient information to understand certain target's emotional situation. Most of human emotional expressions are able to be observed on their face than any other signs. Thus, for cutting-edge interfaces, which need to communicate with a human user, real-time facial expression recognition system have to be utilized for situation understanding. In this paper, we construct a real-time facial expression recognizer using a deep neural network which is invariant to subject.

Deep Neural Networks [1] have been widely used for pattern recognition and classification tasks. More than all, Convolutional Neural Networks (CNN) is shown better performance than other kinds of neural networks. CNN consist of several convolutional layers and fully-connected layers. Each convolutional layer extracts features from images in the training set. Extracted features are max-pooled for achieving spatial invariant. Then, features become more complex with successive convolution layers. In the highest convolutional layer, convolutional filters finally reflect objects' representative features. The outputs from several filters from convolutional layers are classified by fully-connected layers.

Histogram of Oriented Gradient (HOG) [2] is a feature descriptor, which is widely used for the object detection in computer vision. It is invariant to illumination or geometric transformation, but not for orientation. Input frame image is divided into small regions, cells. These cell pixel values are multiplied by the derivative kernel to calculate the gradient of the image. Sobel mask [3] or simple 1d derivative kernels are widely used for calculating the gradient. Edge orientation can be obtained by applying these filters with various angles. 8 bins with unit angle 45 degrees, 9 bins with 24 degrees or 24 bins with 15 degrees might be used for orientation quantization due to the purpose of feature description. Cells from generated orientation map which have edge orientation intensities are gathered and form a larger spatial unit, blocks. In this process, orientation intensity is contrast-normalized for better invariance to

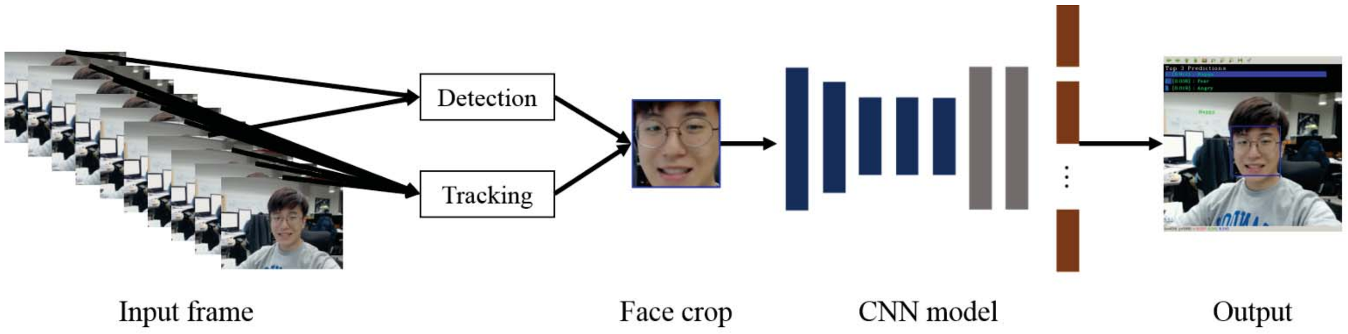

Figure 1. Recognizer Model Overview

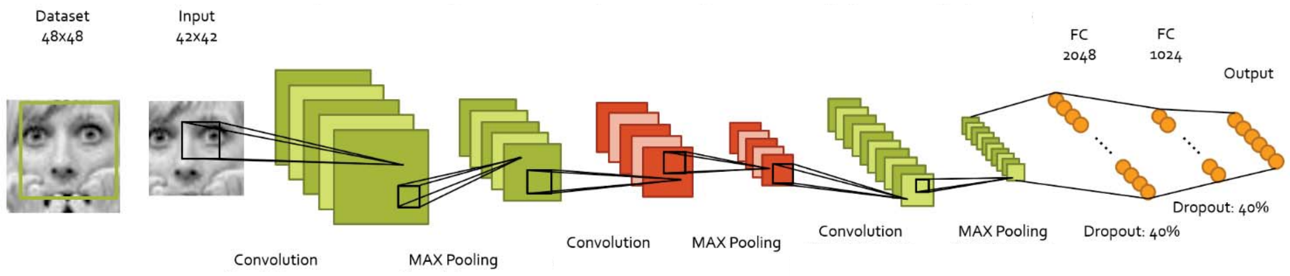

Figure 2. Our CNN Model Architecture

illumination and shadowing. The histogram is obtained by collecting intensity from cells in the block and form 1-D data. Conventional SVM classifier is used for the final step of detection with overlapping grids of HOG descriptor.

CNN is used for facial expression recognition task with Tang [4] and Bergstra [5] and achieved the best performance on Kaggle facial expression recognition challenge (2013). Tang used Convolutional Neural Network model with Linear-SVM instead of softmax layer in classification phase. His model performed the best accuracy 69.77% on the challenge. Bergstra used Null model consists of convolution, Principal Component Analysis (PCA), pooling, normalization and linear readout. His model is concentrated to hyperparameter optimization.

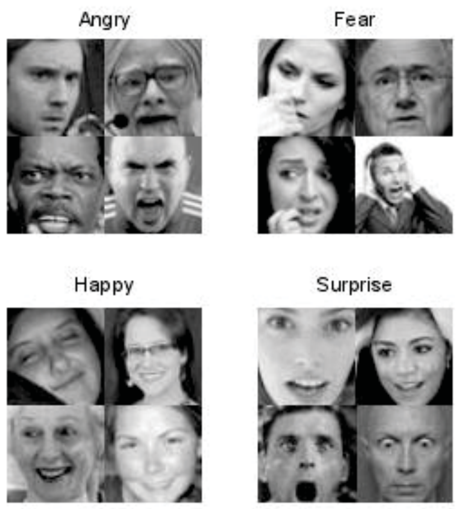

Figure 3. Example images of Kaggle database

2. OVERVIEW OF THE MODEL

Our model detects a human face and recognizes his facial expression by two steps: face detection & tracking and CNN based recognition.

Face detection is performed with HOG feature descriptor combined with a linear classifier. This type of detector is general and suitable for not only human face detector, but also semi-rigid objects. Face localization only with face detection takes too long calculation time. Thus, in order to perform real-time recognizer, the face location is initiated by face detector with the first frame and use correlation tracker [6] to track the face for the rest of frames. Re-initialization is done after certain frame of tracking period to prevent tracking out-of-sight object and compensate tracking error (see the left side of Figure 1).

After localizing face, facial part is cropped and its dimension is reduced to 42px×42px which is input dimension of our CNN model. Features from the input image are extracted and classified by our CNN model. Then, finally we get real-time facial expression recognition result (see the right side of Figure 1 and Figure 2).

3. DATASET AND METHODOLOGY

3.1 Dataset for CNN Model

Kaggle [7] facial expression recognition challenge database is used for training and testing performance. This dataset has 7 facial expression categories (angry, disgust, fear, happy, sad, surprise and neutral), 28,709 training images, 3,589 validation images and 3,589 test images with grayscale image size 48px×48px (see Figure 3). This dataset contains human frontal face with various illumination, poses, and domains, even cartoon characters are included.

3.2 Face Detection & Tracking

When the model initialized, face detection is implemented to find a face for every frame. Then once it succeeds to detect a face, detected face position and the pixel data inside the detected area is

Table 1. Our CNN model architecture

type	kernel	stride	pad	output	dropout
input				42×42×1	
convolution1	5×5	1	2	42×42×32	
pooling1	3×3	2		21×21×32	
convolution2	4×4	1	1	20×20×32	
pooling2	3×3	2		10×10×32	
convolution3	5×5	1	2	10×10×64	
pooling3	3×3	2		5×5×64	
fully-connected1				1×1×2048	40%
fully-connected2				1×1×1024	40%
output				1×1×7	

used for face tracking module. On this phase, only one frame out of 20 frames of the sequential input frame is used for detection and rest of frame are used for tracking module for reducing computation time on real-time recognizer.

We used public C++ library *dlib* [8] for HOG feature-based face detection and correlation tracking.

3.3 CNN Based Recognition

3.3.1 CNN model

Our CNN model consists of three convolutional layers with max-pooling layers and two fully-connected layers. Detail specification of parameters is listed on Table 1. Implementation of CNN was done with public C++ deep learning library *Caffe* [9].

3.3.2 Training phase

On training phase, we first removed faulty images in the training set. 9 images of which pixels are all black were eliminated in this procedure.

To train more data into our CNN model, we performed data augmentation. We used 42px×42px randomly cropped images for training data input (see Figure 4). By this method, 8 times more data is generated for training and spatial invariance is induced.

We trained our model shown in Figure 2 with augmented data. Order of images in the training set is randomly shuffled before training started. The model is trained by minimizing loss function with back-propagation [10] and stochastic gradient descent method. To avoid overfitting, dropout [11] is applied on fully connected layers.

3.3.3 Testing phase

On testing phase, we used an averaging method (see Figure 5) to reduce outliers. Four corners cropped images and their mirrored images were used for testing data. Then, their probability is averaged for the final output. By this procedure, we could decrease classification error a bit.

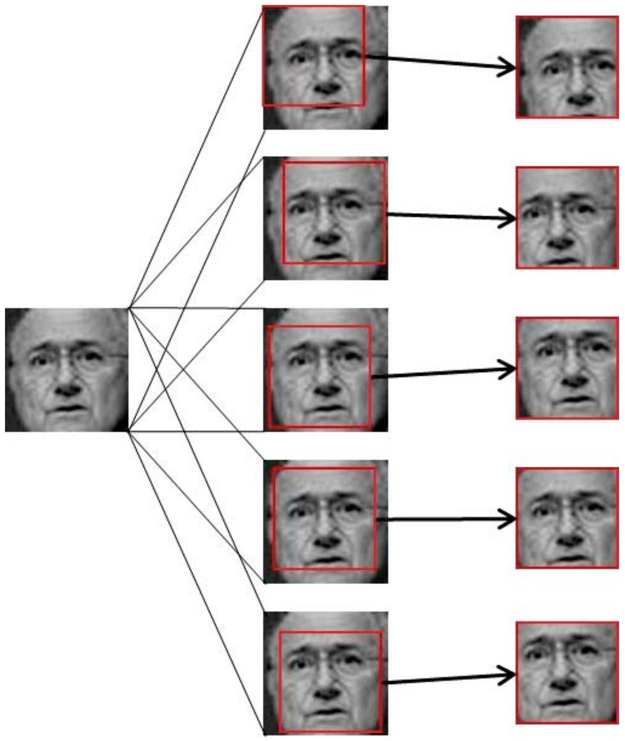

Figure 4. Data augmentation

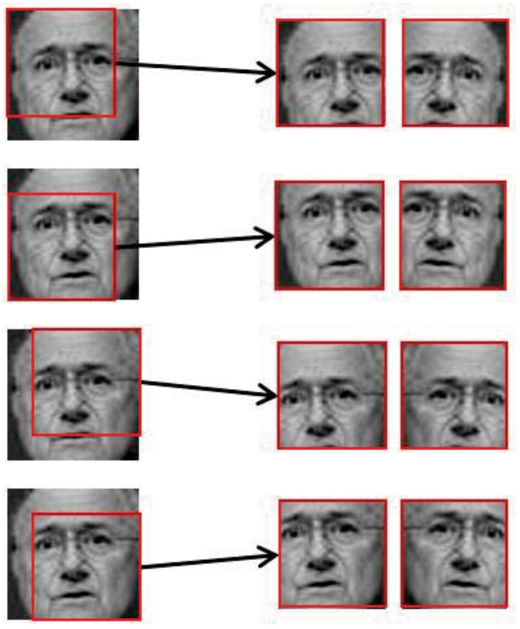

Figure 5. Averaging method

4. PERFORMANCE

4.1 Processing time

When the face is not detected, only detection model is activated. Although CNN model is not executed, it takes 110ms (9.1fps) to process a single frame. Which means in the worst case, our model takes 110ms in response to unexpected subject appearance.

Table 2. Test accuracy confusion matrix of our CNN model

	Angry	Disgust	Fear	Happy	Sad	Surprise	Neutral
Angry	**61.1**	1.0	6.3	3.5	15.3	1.6	11.2
Disgust	23.6	**67.3**	1.8	0.0	1.8	1.8	3.6
Fear	10.6	0.2	**49.5**	3.0	19.7	8.1	8.9
Happy	1.4	0.0	0.8	**88.7**	3.9	1.6	3.6
Sad	6.1	0.0	8.1	5.1	**65.2**	0.8	14.8
Surprise	2.2	0.0	5.8	3.6	2.9	**83.2**	2.4
Neutral	3.6	0.3	3.7	7.3	14.4	1.5	**69.2**

After it detects a face, process time drops to 43.7ms (22.9fps) average to process a single frame. It is because face tracking and classification takes much shorter processing time than face detection. Processing times were measured on an Nvidia GeForce GTX 650 Ti GPU.

4.2 CNN Based Recognition

Classification accuracy in testing phase is shown on Table 2. The average accuracy for all categories was 70.74%. Accuracy for the happy and surprise category was higher than the others, but accuracy for the fear category was poor.

5. ACKNOWLEDGMENTS

This research was supported by: (1) the Brain Korea 21+ Project, BK Electronics and Communications Technology Division, KAIST in 2015, (2) SK Telecom, (3) POONGSAN and (4) the MSIP, Korea, under the G-ITRC support program (IITP-2015-R6812-15-0001) supervised by the IITP.

6. CONCLUSION

This paper suggests real-time facial expression recognizer based on the deep neural network model. We show that our recognizer can detect, track and classify human face expression with high frame rate and classification accuracy.

The reason for a low testing accuracy of some category is considered that the number of images for the category is imbalanced. In Kaggle facial recognition challenge training dataset, 7215 images are in the happy category and 436 images are in disgust category. Considering that there are averagely 4101 images for a single category, this imbalanced category size is enough to cause classification error. Using more data from another dataset on the corresponding category or from data augmentation might increase the performance of our CNN model.

7. REFERENCES

[1] A. Krizhevsky, I. Sutskever, and G. Hinton. 2012. ImageNet classification with deep convolutional neural networks. In *NIPS*.

[2] Dalal, Navneet, and Bill Triggs. 2005. Histograms of oriented gradients for human detection. *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on. Vol. 1. IEEE.*

[3] Sobel, Irwin. 2014. History and definition of the sobel operator. *Retrieved from the World Wide Web*.

[4] Tang, Yichuan. 2013. Deep learning using linear support vector machines. *arXiv preprint arXiv:1306.0239*

[5] Bergstra, James, and David D. Cox. 2013. Hyperparameter Optimization and Boosting for Classifying Facial Expressions: How good can a" Null" Model be?. *arXiv preprint arXiv:1306.3476*

[6] Danelljan, Martin, et al. 2014. Accurate scale estimation for robust visual tracking. *British Machine Vision Conference*, Nottingham, September 1-5, 2014. BMVA Press.

[7] Goodfellow, Ian J., et al. 2013. Challenges in representation learning: A report on three machine learning contests. *Neural Information Processing*. Springer Berlin Heidelberg.

[8] Davis E. King. 2009. Dlib-ml: A Machine Learning Toolkit. *Journal of Machine Learning Research* 10, pp. 1755-1758

[9] Jia, Yangqing, et al. 2014. Caffe: Convolutional architecture for fast feature embedding. *Proceedings of the ACM International Conference on Multimedia. ACM.*

[10] Y. LeCun, B. Boser, J. Denker, D. Henderson, R. Howard, W. Hubbard, and L. Jackel. 1989. Backpropagation applied to handwritten zip code recognition. *Neural Comp*.

[11] Srivastava, Nitish, et al. 2014. Dropout: A simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research* 15.1: 1929-1958.