

ISSN: 2508-7894 © 2017 KAIA. <http://www.kjai.or.kr>

Doi: <http://dx.doi.org/10.24225/kjai.2017.5.2.15>

A Study on the Recognition of Face Based on CNN Algorithms

CNN 알고리즘을 기반한 얼굴인식에 관한 연구

¹ Da-Yeon Son(손다연), ² Kwang-Keun Lee(이광근)

¹, First Author School of Medical IT Industry, Eulji University, Korea, Tel: +82-31-740-7392,
E-mail: gad03364@naver.com

², Corresponding Author School of Silver Industry, Kyung Dong University, Korea, Tel: +82-33-639-0348,
E-mail: kankun@k1.ac.kr

Abstract

Recently, technologies are being developed to recognize and authenticate users using bioinformatics to solve information security issues. Biometric information includes face, fingerprint, iris, voice, and vein. Among them, face recognition technology occupies a large part. Face recognition technology is applied in various fields. For example, it can be used for identity verification, such as a personal identification card, passport, credit card, security system, and personnel data.

In addition, it can be used for security, including crime suspect search, unsafe zone monitoring, vehicle tracking crime. In this thesis, we conducted a study to recognize faces by detecting the areas of the face through a computer webcam. The purpose of this study was to contribute to the improvement in the accuracy of Recognition of Face Based on CNN Algorithms. For this purpose, We used data files provided by github to build a face recognition model. We also created data using CNN algorithms, which are widely used for image recognition. Various photos were learned by CNN algorithm. The study found that the accuracy of face recognition based on CNN algorithms was 77%. Based on the results of the study, We carried out recognition of the face according to the distance. Research findings may be useful if face recognition is required in a variety of situations. Research based on this study is also expected to improve the accuracy of face recognition.

Keywords: CNN algorithms, Recognition of Face, Deep learning, Tensorflow.

1. 서론

최근 정보 보안 문제를 해결하기 위하여 생체 정보를 이용하여 사용자 인식 및 인증을 하는 기술들이 개발되고 있다. 생체 정보를 이용하여 인증 및 인식하는 기술은 생체 정보를 가지고 사람을 구별하는 기술을 말하며, 현재 연구되고 있는 생체 정보로는 얼굴, 지문, 홍채, 음성, 정맥 등이 있다. 이 가운데 얼굴 인식 기술은 큰 비중을 차지하고 있으며, 여러 분야에서 적용되고 있다. 예를 들면 주민등록증, 여권, 신용카드, 보안시스템, 인사자료 등 신원 확인용으로 사용될 수 있으며, 범죄용의자 검색, 우범지대 감시, 범죄 차량 추적 등 치안용으로도 사용할 수 있다. 이러한 얼굴 인식을 위해서는 먼저 얼굴검출이 되어야 한다. 얼굴검출은 얼굴이라는 객체를 찾아내 영상에서 위치를 확인해내는 것이다. 하지만 얼굴검출은 검출하고자 하는 대상의 얼굴상태(표정, 수염, 피부색, 안경, 머리 모양)와 주변 환경(조명의 변화, 거리) 등 외부 요인에 의해 영향을 많이 받는다. 이러한 이유로 얼굴검출은 얼굴영상정보를 이용하여 영상인식을 수행하는 연구자들 사이에서 가장 민감한 요소로 주목 받고 있으며, 얼굴인식분야에서 가장 중요한 요소인 사용자 얼굴영역정보의 정확한 검출 이야말로, 얼굴인식 시스템의 성능을 좌우한다고 할 수 있을 것이다. 얼굴인식의 장점은 다른 생체인식 기술이 사용자로 하여금 일정한 동작을 취하도록 요구하는 것과 달리 비 접촉하면서 자연스럽게 확인할 수 있는 것이며, 사용자는 자신이 현재 검사 당하고 있다는 사실을 인지하지 못하는 상태에서 수행되므로 거부감이 적다. 그 중에서도 인간의 시각 처리 과정을 모방하기 위해 개발된 CNN(convolutional neural networks)은 영상인식 분야에 다양하게 적용되어 높은 성능을 보이고 있다. 따라서 본 논문에서는 CNN 알고리즘을 통해 얼굴 영역을 검출하였다. 얼굴 영역 검출은 카메라로부터 입력되는 영상에서 움직임이 감지되는 영역에 관해서 특징 점을 추출하여 얼굴영역을 식별한다. 얼굴검출에 있어서 특징 점은 눈, 코, 입과 같은 얼굴의 요소 및 특정 위치나 다양한 형태의 고유 요소를 의미한다. 본 논문에서는 이러한 연구배경을 바탕으로 컴퓨터 webcam을 통해서 얼굴 영역을 검출하여 얼굴을 인식하는 연구를 실행하였다. 얼굴 인식 모델을 만들기 위하여 github에서 제공되는 데이터 파일을 사용했으며, 이미지 인식에 많이 사용되는 CNN 알고리즘을 이용하여 데이터를 훈련한 모델을 만들었다. 본 연구의 목적은 이러한 연구한 결과를 바탕으로 webcam에 훈련된 모델을 적용하여 카메라를 이용하여 상황에 따라 얼굴을 인식함으로써, 향후 얼굴인식을 통한 보안 시스템에 활용될 수 있도록 하기 위하여 시도되었다.

2. 관련연구

2.1. 영상인식

영상인식은 영상 데이터에서 의미 있는 정보를 추출하는 문제를 의미한다. 영상은 조명 변화나 주변 환경에 영향을 많이 받으며 물체는 관찰 각에 따라 형상이 다양하기 때문에 이를 수학적으로 모델링하기는 쉽지 않다. 일반적으로 물체를 표현하는 특징을 추출하기 위해 경계 혹은 경계의 히스토그램을 계산하거나 물체의 요소를 개별적으로 모델링하고 요소 사이의 관계를 정의하는 파트 모델

등이 사용된다. 추출된 특징을 모델과 비교하여 물체 여부를 판단하는 역할은 기계학습 알고리즘 즉, 딥러닝(Deep learning)이 담당한다.

2.2. 얼굴검출

영상을 기반으로 얼굴을 검출하는 연구는 크게 지식 기반방법(Knowledge-based method), 특징 기반 방법(Feature-based method), 템플릿 매칭 방법 (Template-matching method), 외형기반 방법 (Appearance-based method)으로 구분된다. 지식기반 방법은 얼굴의 눈, 코, 입 등의 얼굴 구성요소의 일정한 거리와 위치 관계를 전제로 얼굴을 검출하는 연구로서, 간단한 연산을 통해 빠르게 얼굴 검출이 가능하지만, 복잡한 배경을 가지거나 다수의 얼굴 검출을 수행하기에는 오류가 많으며, 사전에 규칙을 정하기가 매우 어렵다는 문제점이 존재한다. 특징기반 방법은 얼굴 요소들의 크기나 모양, 상호 연관성, 얼굴의 색상과 질감 정보, 또는 이러한 요소들의 혼합된 형태의 정보를 이용해 얼굴을 검출하는 방법이다. 이 방식은 얼굴 부위의 잡음이 존재하거나 주변 환경의 밝기 변화에 민감할 뿐 아니라 다른 사물에 의해 가려지는 폐색(occlusion)이 발생할 경우 얼굴의 특징을 추출하기가 어렵기 때문에 얼굴 검출 성능이 낮아지게 된다는 문제점이 있다. 템플릿 매칭 방법은 얼굴에 대해 표준 템플릿을 생성한 후, 입력 영상과 비교하여 얼굴을 검출하는 방법으로 상대적으로 구현하기 쉬우나, 얼굴 영상의 크기, 모양, 포즈 등의 다양한 변화에 대해 효율적으로 처리하기 어렵다.

2.3. Harr

Harr는 영상에서 영역 간의 밝기 차를 이용한 특징으로 여러 형태의 기본 특징들이 존재하며 이 특징들을 다양한 크기와 위치로 조합하여 물체에 대한 특징을 추출하는 방법이다. 기본 특징에 대한 특징 값은 특징의 흰색 부분에 해당하는 영상 픽셀들의 밝기 합에서 검은색 부분의 밝기 합을 뺀 차로 계산된다. 그리고 특징을 이용한 대상의 식별은 계산된 영역의 밝기 차가 특징에 부여된 임계값 보다 큰지 작은지 여부를 이용한다. 물론 하나의 특징을 사용하는 것이 아니라 다수의 특징을 조합하여 사용하게 되며 같은 종류의 특징이라 할지라도 물체 내에서의 위치 및 크기에 따라 서로 다른 특징으로 간주하기 때문에 거의 무한대에 가까운 특징 조합이 가능하다. 사람의 얼굴 같은 경우에는 머리카락, 눈썹, 눈동자, 입술 등이 특징적인 밝기 차를 가지기 때문에 Harr 특징을 적용하기에 비교적 적합한 대상으로 볼 수 있다. Harr 특징은 기본적으로 물체의 기하학적 정보를 유지하며 물체의 형태 변화 및 약간의 위치 변화를 어느 정도 커버 할 수 있는 특성을 가진다. 하지만 영상의 contrast 변화, 광원의 방향 변화에 따른 영상 밝기 변화에 영향을 받으며 물체가 회전하는 경우에는 검출이 힘들다는 단점이 있다. Harr 특징은 영상에서 얼굴과 비슷한 특징이 있으면 얼굴로 검출하는 단점이 있기 때문에 다양한 전처리 방법과 함께 사용하여 검출 정확도를 높인다.

2.4. 딥러닝 (Deep learning)

딥러닝(Deep learning)은 깊은 신경망 알고리즘과 이를 학습하는 방법을 의미한다. 깊은 신경망은 입력 층과 출력 층을 제외한 은닉 층이 2개 이상인 구조의 신경망으로 1980년대에 처음 제안되었으나 학습에 오랜 시간이 걸리고 학습 데이터에 과 적합 되는 단점 때문에 일반적인 문제에 사용 할 수 없었는데 이러한 문제들을 2000년대 이후 병렬 연산이 가능한GPU(Graphics Processing Unit)의 대거 등장과 과 적합을 방지할 수 있는 기법이 제안되며 해결되었다. 이 후 딥러닝은 급속도로 발전하였고 현재는 음성인식, 장면인식, 얼굴인식 등의 다양한 분야에서 딥러닝이 사용되고 있다.

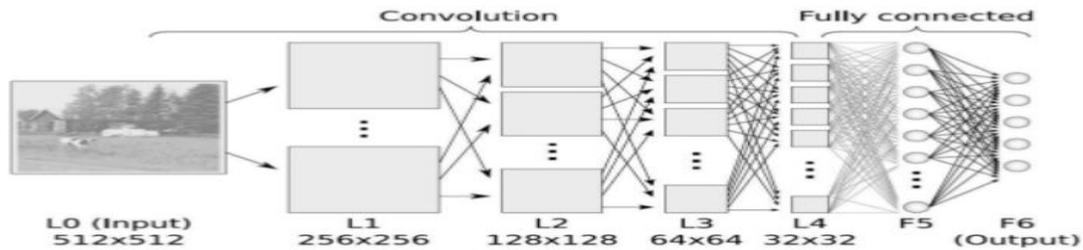
2.5. 텐서플로우(Tensorflow)

텐서플로우(Tensorflow)는 구글에서 오픈소스로 공개한 기계학습 라이브러리로 딥러닝과 기계학습 분야를 일반인들도 사용하기 쉽도록 다양한 기능들을 제공한다. 하이 레벨 프로그래밍 언어로 알려진 파이썬을 활용하여 연산처리를 작성할 수 있다. 다른 언어들도 대부분 지원하지만 파이썬 관련 자료가 가장 많다. 따라서 공개된지 그리 오래되지 않았음에도 불구하고 텐서플로우(TensorFlow)가 다양한 분야에서 활용되고 있다. 텐서플로우(TensorFlow)는 데이터플로우 그래프(Data flow graph)를 사용하여 수치 연산을 하는 소프트웨어로 그래프의 노드(Node)는 수치연산을 나타내고, 엣지(edge)는 노드사이를 이동하는 다차원 데이터 배열(tensor)을 나타낸다. 유연한 아키텍처로 구성되어 있어 코드 수정 없이 데스크탑, 서버 혹은 모바일 디바이스에서 CPU나 GPU를 사용하여 연산을 구동시킬 수 있다. 텐서플로우는 머신러닝과 딥 뉴럴 네트워크 연구를 목적으로 개발되었으며, 여러 다른 분야에도 충분히 적용될 수 있다. 텐서플로우(TensorFlow)를 사용하기 위한 언어는 파이썬이 대표적이며, 수기 숫자 분류와 이미지인식, 워드 임베딩(Word Embedding), RNN(Recurrent Neural Network), 기계번역을 위한 (Sequence to Sequence)모델, 자연어처리, PDE(Partial Differential Equation)기반의 시뮬레이션에 대한 애플리케이션 등이 포함 되어 있다.

2.6. CNN(Convolution Neural Network)

합성곱 신경망(Convolutional Neural Network, 이하, CNN)은 최소한의 전처리(preprocess)를 사용하도록 설계된 다계층 퍼셉트론(multilayer perceptrons)의 한 종류이다. Figure 1에서 보여지는 것처럼 CNN은 하나 또는 여러 개의 합성곱 계층과 그 위에 올려진 일반적인 인공 신경망 계층들로 이루어져 있으며, 가중치와 통합 계층(pooling layer)들을 추가로 활용한다. 이러한 구조 덕분에 CNN은 2차원 구조의 입력 데이터를 충분히 활용할 수 있다. 다른 딥러닝 구조들과 비교해서, CNN은 영상, 음성

분야 모두에서 좋은 성능을 보여준다. CNN은 또한 표준 역전달을 통해 훈련될 수 있다. CNN은 다른 피드포워드 인공신경망 기법들보다 쉽게 훈련되는 편이고 적은 수의 매개변수를 사용한다는 이점이 있다. 최근 딥 러닝에서는 합성곱 심층 신뢰 신경망 (Convolutional Deep Belief Network, CDBN)이 개발되었는데, 기존 CNN과 구조적으로 매우 비슷해서 그림의 2차원 구조를 잘 이용할 수 있으며 그와 동시에 심층 신뢰 신경망 (Deep Belief Network, DBN)에서의 선훈련에 의한 장점도 취할 수 있다. CDBN은 다양한 영상과 신호 처리 기법에 사용될 수 있는 일반적인 구조를 제공하며 CIFAR[25] 와 같은 표준 이미지 데이터에 대한 여러 벤치마크 결과에 사용되고 있다.



<Figure 1> CNN 알고리즘 구조

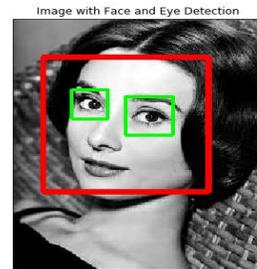
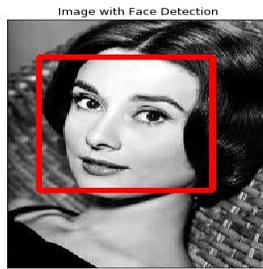
3. 연구방법

3.1. 얼굴검출 및 인식

본 논문에서는 컴퓨터 화면에 장착되어 있는 카메라를 이용하여 사람 얼굴 이미지에서 얼굴 영역을 검출한 후 얼굴을 인식하는 연구를 수행하였으며 다양한 보안 분야에서 활용할 수 있다. Figure 2에서와 같이 github에서 제공되는 사진 데이터로 텐서플로우 라이브러리를 이용하여 CNN 알고리즘을 적용시켜 얼굴 인식 모델을 만들었다. 정확한 얼굴 검출을 위하여 96*96 rgb 얼굴 이미지를 그레이 스케일로 변환하였다. 얼굴 인식 모델을 학습 시키는데 파이썬 프로그래밍 언어를 사용하였으며, 언어 프로그램으로는 jupyter notebook을 사용했다. Figure 3에서의 얼굴 영역 검출은 입력 영상에서 사람의 얼굴을 찾아 그 위치 정보를 추출하였다. 이 경우, 얼굴을 인식하기 위해서 검출된 얼굴영상에서 각 사람의 얼굴을 잘 표현하는 특징의 추출이 필요하다. 또한 이 특징들을 잘 분류할 수 있는 분류기를 통해 다른 얼굴 특징과의 유사도를 비교하는 인식과정이 수행된다. 따라서 본 논문에서는 검출한 얼굴 영역에서 특징 요소인 눈 영역을 Harr 알고리즘을 이용하여 검출하였으며, 위에서 검출한 얼굴 영역의 이미지를 사용하여 Figure 4의 CNN 이미지 인식 과정을 거쳐서 1712개의 샘플데이터를 4개의 Convolution 계층과 2개의 Fully connected 계층 하나의 Dropout 계층을 사용하여 얼굴 인식을 수행하였다. 본 실험에서는 Convolution계층에서는 CannyEdge 필터를 사용하였으며 Max Pooling layer를 통해 얼굴인식을 수행하였다.



<Figure 2> 학습용 얼굴 데이터



<Figure 3> 얼굴 영역 검출 & 특정 영역 추출



<Figure 4> CNN 알고리즘 얼굴 인식 과정

<Table 1> CNN 알고리즘 소스코드

```

image_size = 100
image_size_flat = image_size * image_size
image_shape = (image_size, image_size)
num_channels=1
num_labels = label_count #Currently there are 30 Labels
print(num_labels)
num_iterations = 10000
batch_size = 100
alpha = 0.001;
# Convolutional Layer 1.
filter_size1 = 5
num_filters1 = 16
filter_size2 = 4
num_filters2 = 32
  
```

```

conv_stride2=[1,1,1,1]
# Convolutional Layer 3.
filter_size3 = 3
num_filters3 = 48
# Fully-connected layer.
fc_size = 160

def variable_summaries(var):
    """Attach a lot of summaries to a Tensor (for TensorBoard visualization)."""
    with tf.name_scope('summaries'):
        mean = tf.reduce_mean(var)
        tf.summary.scalar('mean', mean)
        with tf.name_scope('stddev'):
            stddev = tf.sqrt(tf.reduce_mean(tf.square(var - mean)))
            tf.summary.scalar('stddev', stddev)
            tf.summary.scalar('max', tf.reduce_max(var))
            tf.summary.scalar('min', tf.reduce_min(var))
            tf.summary.histogram('histogram', var)

def new_weights(shape):
    return tf.Variable(tf.truncated_normal(shape, stddev=0.05))
def new_biases(length):
    return tf.Variable(tf.constant(0.05, shape=[length]))
def new_conv_layer(name,input,
num_input_channels,
filter_size,
num_filters,
list_strides,
use_pooling=True
):
    with tf.name_scope(name):
        shape = [filter_size, filter_size, num_input_channels, num_filters]
        with tf.name_scope('ConvWeights'):
            weights = new_weights(shape=shape)
            variable_summaries(weights)
        with tf.name_scope('ConvBiases'):
            biases = new_biases(length=num_filters)
            variable_summaries(weights)
        layer = tf.nn.conv2d(input=input,
filter=weights,
strides=list_strides,
padding='SAME',
use_cudnn_on_gpu=True)
        layer = tf.nn.max_pool(value=layer,
ksize=[1, 2, 2, 1],
strides=[1, 2, 2, 1],

```

```

padding='SAME')
layer = tf.nn.relu(layer)

return layer, weights

def flatten_layer(layer):
layer_shape = layer.get_shape()

num_features = layer_shape[1:4].num_elements()
layer_flat = tf.reshape(layer, [-1, num_features])

return layer_flat, num_features
def new_fc_layer(name,input,
num_inputs,
num_outputs,
use_relu=True):

with tf.name_scope(name):
with tf.name_scope('Weights'):
weights = new_weights(shape=[num_inputs, num_outputs])
variable_summaries(weights)
with tf.name_scope('Biases'):
biases = new_biases(length=num_outputs)
variable_summaries(biases)
layer = tf.matmul(input, weights) + biases
if use_relu:
layer = tf.nn.relu(layer)

return layer
x = tf.placeholder(tf.float32, shape=[None, image_size_flat], name='x')
x_image = tf.reshape(x, [-1, image_size, image_size, num_channels])
y_true = tf.placeholder(tf.int64, shape=[batch_size], name='y_true')
#y_true_cls = tf.argmax(y_true, dimension=1)
#Convolutional Layer 1
layer_conv1, weights_conv1 = \
new_conv_layer('ConvLayer1',input=x_image,
num_input_channels=num_channels,
filter_size=filter_size1,
num_filters=num_filters1,
list_strides=conv_stride1,
use_pooling=True
)
#Convolutional Layer 2
layer_conv2, weights_conv2 = \
new_conv_layer('ConvLayer2',input=layer_conv1,
num_input_channels=num_filters1,
filter_size=filter_size2,

```

```

num_filters=num_filters2,
list_strides=conv_stride2,
use_pooling=True
)
#Convolutional Layer 3
layer_conv3, weights_conv3 = \
new_conv_layer('ConvLayer3',input=layer_conv2,
num_input_channels=num_filters2,
filter_size=filter_size3,
num_filters=num_filters3,
list_strides=conv_stride3,
use_pooling=True
)

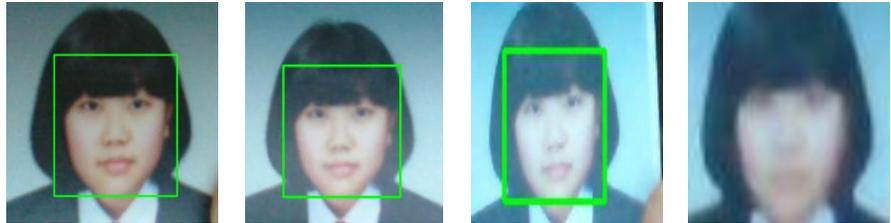
```

3.2. 거리별 얼굴인식 정확도

Figure 5는 거리별 얼굴 인식을 위한 입력 이미지이다. 본 논문에서는 Figure 5의 이미지를 이용하여 컴퓨터에서 제공하는 카메라인 webcam을 통해서 얼굴 인식을 수행하였다. Figure 6은 입력영상의 거리를 1m, 2m, 3m, 5m로 두고 인식하였다.



<Figure 5> 입력 이미지



<Figure 6> 거리에 따른 인식 정확도

4. 연구결과

4.1. 얼굴검출 및 인식

본 논문에서 실행한 연구는 github에서 제공된 데이터셋이며, CNN 알고리즘을 적용시켜 얼굴 인식 모델을 수행하였다. Figure 7은 CNN 알고리즘을 통해 5000번 반복 학습한 데이터의 정확도를 나타낸다. 반복 학습 중 가장 높은 정확도는 77%를 보여주었다.

```

Initialized
--> Validation Accuracy 7.333333333333
7.333333333333
Minibatch loss at step 0: 3.415160
mini batch accuracy: 0.0
--> Validation Accuracy 49.833333333333
49.833333333333
Minibatch loss at step 500: 1.146062
mini batch accuracy: 0.66
--> Validation Accuracy 51.333333333333
51.333333333333
Minibatch loss at step 1000: 0.602483
mini batch accuracy: 0.82
--> Validation Accuracy 55.166666666667
55.166666666667
Minibatch loss at step 1500: 0.033983
mini batch accuracy: 0.99
--> Validation Accuracy 56.0
56.0
Minibatch loss at step 2000: 0.009965
mini batch accuracy: 1.0
--> Validation Accuracy 56.833333333333
56.833333333333
Minibatch loss at step 2500: 0.000290
mini batch accuracy: 1.0
--> Validation Accuracy 56.833333333333
Minibatch loss at step 3000: 0.000152
mini batch accuracy: 1.0
--> Validation Accuracy 57.0
57.0
Minibatch loss at step 3500: 0.000089
mini batch accuracy: 1.0
--> Validation Accuracy 57.333333333333
57.333333333333

```

<Figure 7> 얼굴 인식 학습 정확도

4.2. 거리별 얼굴인식 정확도

Figure 6에 따라 인식의 주변 환경 중 하나인 거리를 다르게 두어 인식한 결과로, 4.5*3.5 크기의 이미지를 1m, 2m, 3m, 5m의 차이를 두어 인식하였을 때, 5m가 넘어가는 거리에서 인식한 결과 얼굴 영역을 검출하지 못해 이미지를 인식하지 못하였다.

5. 결론

본 논문에서는 사용자의 얼굴 이미지 인식 정확도를 향상시켜 나가는데 기여하기 위하여 얼굴 이미지를 인식하는 연구를 시도하였다. 연구목적을 달성하기 위해, 필자는 여러 사진을 CNN 알고리즘을 통해 학습시켜본 결과 77%의 정확도 모델을 만들었다. 또한 이 모델을 기반으로 현재 얼굴 인식에 있어서 문제가 되고 있는 조건 중 하나인 거리에 따른 얼굴인식에 관해서도 수행하였다. 본 연구결과는 다양한 상황에서 얼굴을 인식해야 하는 경우에 유용하게 활용될 수 있을 것이며, 나아가 이를 바탕으로 지속적인 연구가 이루어진다면 얼굴인식 정확도를 높여 나가는데 많은 기여를 할 것으로 기대된다. 아울러, 최근 딥러닝 분야의 연구가 늘어나고 있으며, 생체 인식 중 얼굴 인식을 통한 사용자 인식 기술이 늘어나고 있는 추세이므로, 향후에는 얼굴인식에 오류를 제공하는 조건을 해결하여 얼굴인식 정확도를 높이기 위한 연구가 필요하다고 판단된다.

References

- Byeon, Young-Hyun, & Kwag, Geun-Chang (2014). A Video-based Face Recognition Using 3D Convolutional Neural Networks Under Robot Environments, Proceedings of KIIT Summer Conference, pp.26-29.
- Choi, In-Kyu (2016). CNN-based facial expression recognition. Proceedings of Conference of KIBME, No.6, pp. 320-321.
- Hong, Sung-eun (2016).Deep CNN-based Person Identification using Facial and Clothing Features, Proceedings of Conference of IEIE, pp. 2204-2207.
- Hwang, Won-Jun (2017). A Study on the Research Trends of Deep Learning Based Face Detection, Landmark Detection and Face Recognition Techniques. *Broadcasting and Media Magazine*. 22(4), 41-49.
- Jeon, Young-Cheol, Lee, Keon-Ik, & Kim, Kang(2008). Simply Separation of Head and Face Region and Extraction of Facial Features for Image Security. *Journal of the Korea Society of Computer and Information* , 13(5), 125-133.
- Kang, Dea-Sung, & Kong, Ki-Ho (2017). A Study of Face Detection Algorithm Using CNN Based on Symmetry-LGP & Uniform-LGP and the Skin Color. *Journal Of Advanced Information Technology And Convergence*, 15(1), 107-113.
- Kim, Nam-Ho, & Bang, Kwang-Min (2006). The Face Recognition in continuous video stream, Proceedings of Conference of KMMS, No.2, pp.892-896.
- Kim, Chan, & Kwon, Young-Su (2017). Understanding Image Recognition using Deep-Learning : using Caffe, Py-Caffe and the Faster-RCNN, Proceedings of Conference of IEIE, No.6, pp.1115-1118.
- Kim, Yeon-Ho, Park,Sung- Wook, & Kim, Do- Yeon(2017). Research on Robust Face Recognition against Lighting Variation using CNN. *The Journal Of Korean Institute Of Communications And Information Sciences*, 12(2),pp. 325-330.
- Mun, Hae-Min, & Ban, Sung-Bum (2013). Performance Analysis of Face Recognition by Distance according to Image Normalization and Face Recognition Algorithm. *Journal of the Korea Institute of Information Security and Cryptology*, 23(4), 737-742.
- Park, Jun-Gyu (2016). A Implementation of Face Recognition System Using CNN Algorithm Based on Symmetry Uniform Local Gradient Pattern, Thesis for master's in Dong-A University, Busan, Korea.
- Sin, Jin-Woo (2017). Introducing the Latest Deep Learning Algorithm for Image Recognition. *Journal of The Korean Institute of Communication Sciences*, 34(7), 25-30.