JANT 한국항행학회논문지
Journal of Advanced Navigation Technology

기반 기술

J. Adv. Navig. Technol. 24(5): 430-437, Oct. 2020

Caffe를 이용한 얼굴 인식 파이프라인 모델 구현

Implementation of Face Recognition Pipeline Model using Caffe

박 진 환 · 김 창 복*
가천대학교 에너지 IT학과

Jin-Hwan Park · Chang-Bok Kim*
Department of Energy IT, Gachon University, Gyeonggi-do, 13120, Korea

[요 약]

제안 모델은 얼굴 검출과 랜드마크 및 얼굴 인식 알고리즘을 이용하여 인공신경망으로 학습을 통해 얼굴 예측률과 인식률을 향상하는 모델을 구현하였다. 제안 모델은 특정 인물의 얼굴 영상에서 랜드마킹을 한 후, 기존에 학습된 Caffe 모델을 이용하여 얼굴 검출과 임베딩 벡터 128D를 추출하였다. 학습은 기계학습 알고리즘인 SVM (support vector machine)과 DNN (deep neural network)을 구축하여 학습하였다. 얼굴인식은 학습된 모델을 이용하여 학습된 인물 중 다른 얼굴 영상으로 테스트하였다. 실험 결과, SVM보다는 DNN으로 학습한 결과가 우수한 예측률과 인식률을 보였다. DNN의 중간층을 증가하게 되면 예측률은 높아지나 인식률이 감소하는 현상이 발생하였다. 이것은 인식하고자 하는 대상이 적음으로써 발생하는 과적합으로 판단된다. 제안 모델은 명확한 얼굴 영상을 추가하여 학습한 결과, 높은 예측률과 인식률의 결과를 얻을 수 있음을 확인할 수 있었다. 본 연구는 좀 더 많은 얼굴 영상 데이터를 이용함으로써 보다 효과적인 딥러닝 구축을 통해 보다 향상된 인식률과 예측률을 얻을 수 있을 것이다.

[Abstract]

The proposed model implements a model that improves the face prediction rate and recognition rate through learning with an artificial neural network using face detection, landmark and face recognition algorithms. After landmarking in the face images of a specific person, the proposed model use the previously learned Caffe model to extract face detection and embedding vector 128D. The learning is learned by building machine learning algorithms such as support vector machine (SVM) and deep neural network (DNN). Face recognition is tested with a face image different from the learned figure using the learned model. As a result of the experiment, the result of learning with DNN rather than SVM showed better prediction rate and recognition rate. However, when the hidden layer of DNN is increased, the prediction rate increases but the recognition rate decreases. This is judged as overfitting caused by a small number of objects to be recognized. As a result of learning by adding a clear face image to the proposed model, it is confirmed that the result of high prediction rate and recognition rate can be obtained. This research will be able to obtain better recognition and prediction rates through effective deep learning establishment by utilizing more face image data.

Key word : Face detection, Face alignment, Embedding vector, Face recognition, Caffe.

https://doi.org/10.12673/jant.2020.24.5.430

Received 14 September 2020; **Revised** 21 September 2020
Accepted (Publication) 26 October 2020 (30 October 2020)

***Corresponding Author : Chang-Bok Kim**

Tel : +82-10-8908-3946
E-mail : cbkim@gachon.ac.kr

Ⅰ. 서 론

인공 신경망은 하드웨어와 알고리즘의 발전으로 딥러닝(deep learning)으로 재조명받으며 컴퓨터 비전, 영상 처리 등의 분야에서 눈부신 성능향상을 보이고 있다. 특히 영상 분야에서 영상 인식, 예측 등을 이용하여 산업, 의료, 교육, 교통 등 거의 모든 분야에 적용되고 있는 현실이다. 딥러닝은 스스로 영상의 특징 패턴을 학습하여 환경 변화에 따른 가변적 요소에도 강인한 인식 성능을 보이고 있다[1]. 얼굴 인식은 패턴이 동일하여 개개인을 구별해야 하는 어려움이 있었으나 딥러닝의 발전과 더불어 많은 발전을 이루었으며, 보안, 엔터테인먼트, 모바일 서비스 등 다양한 분야에서 활용되고 있다.

얼굴 인식은 얼굴 검출, 랜드마크 검출을 통한 얼굴 정규화 그리고 얼굴 인식 모듈로 나누어져서 발전하고 있다. 이것은 각 모듈 단위로 필요한 분야가 있기 때문이지만 얼굴 인식은 이러한 모든 모듈이 필요하다.

얼굴 검출 알고리즘은 Haar-like 방식[2], AlexNet을 기반으로 한 방식[3]. 머리, 눈, 코, 입 그리고 수염 등을 개별적으로 검출하여 최종 얼굴로 판단하는 방식[4]. 고속 물체 검출기인 YOLO(you only look once)와 R-CNN(region convolution neural network)을 fine-tuning한 방식[5], [6], P-net, R-net, O-net이라는 세 CNN을 차례로 통과하는 MTCNN[7] 방식이 있다. 얼굴 랜드마크 검출은 cascade 기반의 CNN을 이용한 방식[8]. coarse-to-fine 방식의 개념을 이용하여 50×50 입력 영상에서 68개의 점을 검출하는 방식[9] 등이 있다. 얼굴 인식은 Facebook에서 발표한 Deepface[10], CVPR에서 발표된 DeepID1[11]과 성능을 개량한 DeepID2[12] 등이 있다.

본 연구는 얼굴 검출과 랜드마크 및 얼굴 인식 알고리즘을 이용하여, 인공 신경망으로 학습을 통해 얼굴 예측률과 인식률을 향상하는 모델을 구현하였다. 구현 모델은 얼굴 검출, 랜드마킹, 얼굴 임베딩, 학습 모듈, 얼굴 인식 모듈로 구성되어있다. 즉, 특정 인물의 얼굴 영상에서 랜드마킹을 한 후, 얼굴 검출과 임베딩은 랜드마킹으로 표준 정렬된 얼굴을 이용하여 딥러닝 영상처리 라이브러리인 Caffe를 사용하였으며, 얼굴 임베딩 후에 임베딩 벡터 128D를 추출하였다. 학습은 기계학습 알고리즘인 SVM(support vector machine)[13]와 DNN(deep neural network)을 구출하여 학습하였다. 이때, 임베딩 벡터 128D를 독립변수로써 입력으로 하고 실제값인 라벨은 임베딩 벡터에 해당하는 특정 인물을 one hot 인코딩하여 학습하였다. 얼굴 인식은 학습된 모델을 이용하여 학습된 인물 중 다른 얼굴 영상에서 동일한 방법으로 얼굴 검출, 랜드마킹, 얼굴 임베딩 후 임베딩 벡터 128D를 입력으로 하여 테스트하였다. 본 논문은 2장에서 관련 연구로서 얼굴 검출 및 인식 플랫폼 그리고 얼굴 인식 알고리즘을 서술하였으며, 3장에서 제안 모델을 제안하였다. 또한, 4장에서 구현된 모델의 실험 결과 및 비교 분석을 하였으며, 마지막으로 결론에 관해서 서술하였다.

Ⅱ. 얼굴 검출 및 인식

2-1 얼굴 검출 및 인식 플랫폼

Caffe (convolutional architecture for fast feature embedding)[14]는 오픈소스 플랫폼으로 알고리즘과 모델을 수정 가능하며, 컴퓨터 비전 및 음성 인식, 로보틱스, 뉴로 사이언스 등 활용영역을 넓혀가고 있다. Caffe는 데이터 저장소, 레이어, 네트워크, 학습 등의 구조로 구성된다.

데이터 저장소는 blob 이라고 하는 4차원 배열을 이용하여 데이터를 저장하고 모듈 간 통신하며, Google protocol buffers에 의해 저장된다. 레이어(layer)는 특정 프로세스를 나타내며, blob을 입출력으로 하여 필터, 풀링, 내적, 정규화, 시그모이드, 변형, 데이터 로딩, 정확도 계산, 로스 계산, 소프트 맥스 등 딥러닝 네트워크의 모든 연산처리를 제공한다. 네트워크는 하나의 딥러닝 단위로서 데이터를 로딩하는 데이터 레이어 부터 오류함수를 계산하는 오류 레이어까지 하나의 네트워크로 구성된다. 네트워크는 입력된 데이터가 출력으로 전달되는 forward 연산과 출력 결과를 이용하여 학습하는 backward 연산이 있다.

네트워크를 학습하기 위해서는 deploy, solver, train_val 등 3개의 prototxt 파일이 필요하다. train_val은 학습을 시키기 위한 구조고, solver는 네트워크 학습 방법을 명시하며, deply는 학습 완료된 모델을 이용하여 classifying에 사용하는 구조이다. 다음은 train_val.prototxt에서 네트워크 시작 부분을 나타냈다.

```
name: "CaffeNet"
layer {
  name: "data"
  type: "Data"
  top: "data"
  top: "label"
  include {
    phase: TRAIN
  }
  transform_param {
    mirror: true
    crop_size: 227
    mean_file: "data/ilsvrc12/imagenet_mean.binaryproto"
  }
}
  data_param {
    source: "examples/imagenet/ilsvrc12_train_lmdb"
    batch_size: 256
    backend: LMDB
  }
}
```

solver.prototxt는 학습을 위한 네트워크인 train_val.prototxt 파일의 경로와 학습률, 정책, 최대 학습 횟수 등을 지정한다.

Caffe는 내부의 주요 클래스와 함수를 파이선에서 사용할 수 있도록 파이선 API로 제공하고 있다. 즉. Caffe는 학습을 위한 solver 클래스, 네트워크 구성과 관련된 net 클래스 등 약 60개의 계층이 클래스로 정의되어 있다.

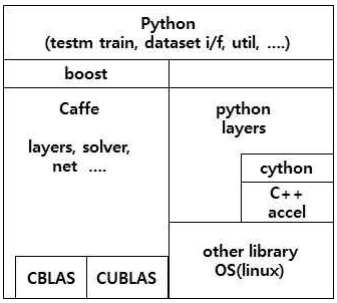


그림 1. Caffe 구조
Fig. 1. Caffe structure.

또한 고속 연산을 위해 multi-core를 활용하는 CBLAS(c basic linear algebra subroutines)나 GPU를 이용하는 CUBLAS (cuda blas)를 사용하고 있다.

2-2 얼굴 인식 알고리즘

컴퓨터 비전에서 metric 학습은 유클리언 거리를 기반으로 각 영상을 설명할 수 있는 특징을 추출하고 유사성을 검토하여 분리하는 것이다. CNN은 모양에 따른 서로 다른 분류는 우수하지만, 사람의 얼굴과 같이 유사한 모양에서 다른 패턴을 분류하는 것에 한계가 있다. triplet 오차 학습은 이러한 문제점을 해결하기 위한 알고리즘이다.

triplet 오차 학습은 3개의 고유한 얼굴 영상으로 구성되며, 기준이 되는 영상과 같은 사람의 얼굴 영상 그리고 다른 사람의 얼굴 영상이다. 그림 2에 triplet 오차 학습에 대해서 나타냈다.

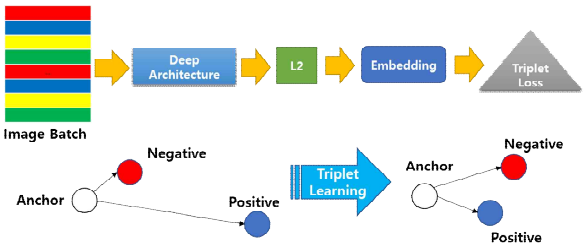


그림 2. triplet 오차 학습
Fig. 2. triplet error learning.

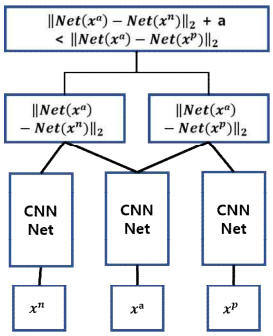


그림 3. triplet 오차 학습 알고리즘
Fig. 3. triplet error learning algorithm.

그림 3에 triplet 오차 학습 과정에 대해서 나타냈다. triplet 오차 학습을 위한 네트워크는 동일한 CNN 모델로서 3개의 얼굴 영상에 대해 128D 벡터를 생성하고, 기준 영상에 대해 긍정 영상과 부정 영상에 대한 각각의 유클리언 거리를 계산하고, L2 Norm을 적용한 뒤 두 거리 사이의 오차 값을 계산한다. 이때 마진 요소를 이용하여, 각각의 거릿값에 변화를 줌으로써 최적의 오차를 구할 수 있다.

$$\| f(x_i^a) - f(x_i^p) \|_2^2 + \alpha < \| f(x_i^a) - f(x_i^n) \|_2^2 \quad (1)$$
$$\forall (f(x_i^a), f(x_i^p), f(x_i^n)) \in T$$

여기서 x_i^a는 기준 영상이고, x^p는 긍정 영상이며, x^n는 부정 영상이다. 또한, f는 임베딩 함수이고, d는 거리 함수로 두 입력 간의 거리를 측정하는 함수이다. 위 식에서 첫 번째 항은 긍정 거리로서 0에 가까울수록 좋으며, 두 번째 항은 멀수록 좋을 것이다. 또한, α는 마진으로써 바이어스와 같은 개념이다. 따라서 이를 오차 함수로서 적용하려면 두 번째 항을 첫 번째 항으로 넘기면 되며, 모든 데이터셋의 대해 오차를 갱신하도록 구현된다. 다음은 triplet 오차 학습을 위한 오차를 나타냈다.

$$cost = \sum_i^N \| f(x_i^a) - f(x_i^p) \|_2^2 - \| f(x_i^a) - f(x_i^n) \|_2^2 + \alpha] \quad (2)$$

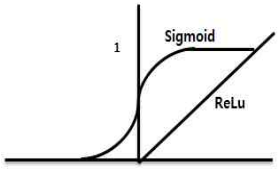
만약 마진이 1일 때, 만약 최적의 오차를 찾아냈다면 긍정 영상이 0이고, 부정 영상이 1이 되기 때문에 0 - 1 + 1이 되어서 최종적으로 오차가 0의 값을 출력하게 된다.

본 연구에서는 triplet 오차 학습을 통해 최적의 128D의 임베딩 벡터를 구했으며, 이를 통해 각 얼굴 영상의 ID를 구하기 위해 SVM과 DNN을 사용하였다. SVM은 분류와 회귀 분석을 위해 사용되며, 패턴인식, 데이터 분석을 위한 지도학습 모델로서 서로 유사한 그룹끼리 초평면(hyperplane)으로 나누는 방식이다. 초평면은 다음과 같이 나타낼 수 있다.

$$H_1 : W^T x_i + w_0 \geq 1 \text{ for } y_i = +1 \quad (3)$$
$$H_2 : W^T x_i + w_0 \geq -1 \text{ for } y_i = -1 \quad (4)$$
$$y_i (W^T x_i + w_0) \geq 1, \ \forall i \quad (5)$$

초평면 H_1 과 H_2 위에 위치하는 모든 튜플은 위 식을 만족한다. 비선형 SVM은 선형적으로 분리할 수 없는 데이터에 적용할 수 있다. 비선형 SVM은 원 데이터를 비선형 매핑을 통해 고차원 공간으로 변환한다.

인공 신경망은 인간의 뇌 구조를 모방한 기계학습으로, 뇌의 뉴런과 유사한 형태인 시그모이드 비선형 활성화 함수를 사용하였다. 그림 4에 활성화 함수에 대해서 나타냈다.



$$\sigma(x) = \frac{1}{1+e^{-x}}$$

$$R(x) = (0, x)$$

그림 4. 활성화 함수
Fig. 4. Activation function.

시그모이드 활성화 함수는 지수함수 특징에 의한 기울기 사라짐, 지역 최솟값 등과 같은 문제와 이에 따른 학습 시간의 길어짐의 문제점이 있었다. 또한, 가중치 초기화 문제와 학습 데이터셋에 너무 가깝게 맞추어 학습하는 과적합의 문제가 있었다. DNN은 relu 활성화 함수를 사용하여, 심층에 의한 기울기 사라짐, 지역 최솟값 문제를 해결한다. 또한, Xavier 초깃값 등과 같은 가중치 초기화 기법을 이용해 가중치 초기화 문제를 해결하고, 과적합 방지를 위해 중간층 노드를 강제로 없애는 드롭아웃 방식을 이용하여 중간층을 깊게 한 심층 신경망이다.

Ⅲ. 제안 얼굴 인식 모델

제안 모델은 주어진 얼굴 영상에 대해서 영상 처리 패키지인 Open CV와 CNN 딥러닝 프레임워크인 Caffe, dlib 등을 이용하여 관심 영역(region of interesting)인 얼굴을 검출하였으며, 높은 인식률과 예측률을 위해서 랜드마킹 방법을 이용하여 추출된 얼굴을 정렬하였다. 또한, triplet 학습을 이용한 Caffe를 통해 정렬된 얼굴 영상의 특징을 추출하여 128D의 임베딩 벡터값을 추출하였다, 제안 모델은 학습을 위해 모든 사람에 대한 128D의 특징 추출 값을 입력으로 하였으며, 각 사람의 이름을 one-hot 인코딩하여 실제값인 라벨 생성하여 출력으로 하였다. 또한 얼굴 인식을 위한 학습 알고리즘은 SVM의 linear, poly, rbf 커널과 출력단의 활성화 함수를 softmax로 한다. 입력 및 분류 신경망과 중간층의 활성화 함수를 relu로 사용한 DNN을 이용하여 학습하였다. 최종적으로 얼굴 인식은 학습된 모델을 이용하여 학습된 인물 중 다른 얼굴 영상에서 동일한 방법으로 얼굴 검출, 랜드마킹, 얼굴 임베딩 후 임베딩 벡터 128D를 입력으로 하여 테스트하였다.

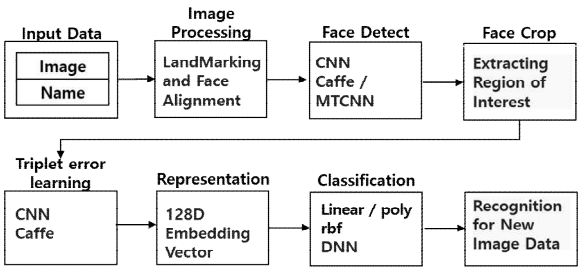


그림 5. 얼굴 인식 파이프라인
Fig. 5. Face recognition pipeline.

그림 5에 얼굴 인식 파이프라인 구조에 대해서 나타냈다. 제안 모델은 다음과 같이 구성된다.

1. 얼굴 교정을 위한 랜드마킹
2. 얼굴 검출 및 관심 영역 추출
2. triplet 오차 학습을 이용한 128D 임베딩 벡터 생성
3. 128D 임베딩 벡터와 라벨을 이용한 SVM 및 DNN 학습
4. 학습된 SVM 및 DNN 모델을 이용한 특정 ID 얼굴 예측 및 인식

입력 데이터는 특정 인물의 얼굴 영상과 이름이다. 얼굴 영상은 CNN의 입력 데이터이며, 이름은 one-hot으로 인코딩되어 실제 데이터인 라벨로 변환된다. 정교한 임베딩과 인식률 향상을 위해서는 얼굴의 정면 사진이 필요하므로, 랜드마크를 통한 얼굴 정렬 과정이 필요하다. 랜드마크는 얼굴 영상을 68개의 점으로 눈, 코, 입, 눈썹, 코, 턱선 등을 나타내는 것이다. 얼굴 정렬은 얼굴 영상에서 눈 부분만을 검출하여 눈동자의 중심과 얼굴의 기울어진 각도를 계산한 후 삼각함수를 이용하여 얼굴을 회전함으로써 표준 정렬하였다.

얼굴 검출은 관심 영역인 얼굴만을 검출하는 것이며, 딥러닝 얼굴 영상 처리 플랫폼인 Caffe를 이용하여 기존에 학습된 모델을 사용하였다. 학습 과정은 triplet 오차 학습 과정을 통해 임베딩 벡터 128D를 추출한다.

```
detector = cv2.dnn.readNetFromCaffe(prototxt, caffemodel)
detection = detector.detect_faces("얼굴영상")
embedder = cv2.dnn.readNetFromTorch(embedding_model])
```

detector 객체는 기존에 학습된 모델인 prototxt과 학습 결과 모델을 사용하였다. 즉, prototxt는 deploy.prototxt, caffemodel은 res10_300x300_ssd_iter_140000.caffemodel을 사용하였다.

detector 객체는 200개의 박스가 구해지고, 각 박스에 얼굴일 확률을 내림차순으로 정렬하여 출력된다. 이중 가장 확률이 높은 박스가 얼굴 객체로 인식하게 되며, 이 박스를 이용하여 얼굴 좌표를 구할 수 있다.

얼굴 특징 추출을 위한 embedder 객체는 128D 임베딩 벡터를 생성하는 Torch 딥러닝 모델인 openface_nn4.small2.v1.t7을 사용하였다. 또한 128D 임베딩 벡터와 이름을 수치화하여 피클 파일에 저장하였다.

```
embedder = cv2.dnn.readNetFromTorch(embedding_model])
embedder.setInput(faceBlob)
 vector  = embedder.forward()
```

학습은 피클 파일로 저장된 임베딩 벡터와 라벨을 이용하여, SVM, DNN을 사용하여 학습하였다. 이때, 임베딩 벡터 128D를

독립변수로써 입력으로 하고 실제값인 라벨은 임베딩 벡터에 해당하는 특정 인물을 one hot 인코딩하여 학습하였다. SVM 학습은 다음과 같다.

```
recognizer = SVC(C=1.0, kernel="linear", probability=True)
recognizer.fit(data["embeddings"], labels)
```

recognizer 객체는 linear, poly, rbf 커널을 사용하였다. 학습 결과 생성된 모델과 one hot 인코딩된 라벨은 피클 파일에 저장하였다. 또한, DNN1 학습은 다음과 같다.

```
model = Sequential()
model.add(Dense(10, input_dim = 128, activation = 'softmax'))
```

DNN1 학습의 입력은 128D 임베딩 벡터이며, 출력은 인식하고자 하는 특정 인물들의 라벨 수 10이다. 다중 분류를 위해 오차 함수를 categorical_crossentropy를 사용하였으며, optimizer를 adam으로 사용하였다. 본 모델에서는 학습 데이터가 많지 않기 때문에 validation_data를 학습과 테스트 데이터를 그대로 사용하였다. 또한, 학습률은 0.1, 배치 사이즈는 1 그리고 학습 횟수는 200으로 하였다. DNN2 학습은 다음과 같다.

```
model = Sequential()
model.add(Dense(128,  input_dim = 128, activation = 'relu'))
model.add(Dense(128, activation = 'relu'))
model.add(Dense(4,  kernel_initializer='uniform',  activation  =
'softmax'))
```

DNN2는 DNN1 학습과 마찬가지로 입력은 임베딩 벡터인 128D이며, 출력은 인식하고자 하는 특정 인물의 라벨 수 10이다. 그러나 중간층을 1층과 2층을 사용하였다. 1층과 2층의 노드 수는 128개이며, 중간층의 활성화 함수는 relu를 사용하였다. 오차 함수, optimizer, validation_data, 학습률, 배치 사이즈, 학습 횟수는 DNN1과 동일하였다. 그림 6에 중간층이 1층인 DNN1 구조에 대해서 나타냈다.

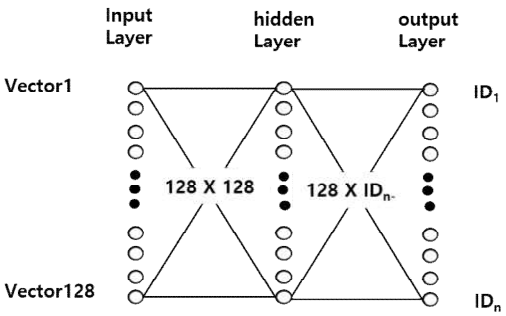


그림 6. DNN 구조
Fig. 6. DNN structure.

DNN1과 DNN2의 학습 결과인 학습 모델은 가중치로서 model.h5 파일에 저장하였다.

얼굴 인식은 학습하지 않은 얼굴 영상을 인식하는 것이다. 인식 과정은 다음과 같다.

1. 전 단계에서 추출한 학습 결과인 학습 모델 파일과 라벨 파일을 로드
2. 테스트를 위한 얼굴 영상에 대해 얼굴 검출, 정렬, 자름, 128D 임베딩 벡터 생성
3. 128D 임베딩 벡터를 학습 모델에 입력하여 테스트 얼굴 영상에 대한 예측률 및 인식률을 구한다.

```
preds = recognizer.predict_proba(vec)[0]
preds = model.predict(np.array(vec))[0]
```

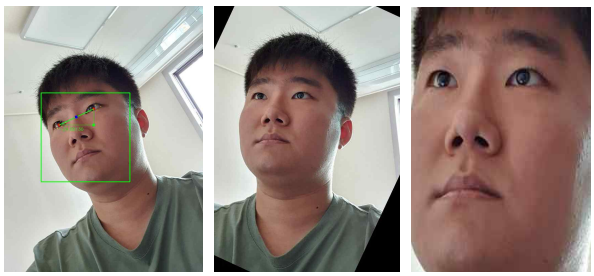
recognizer.predict_proba(vec)[0]는 SVM의 linear, poly, rbf 커널에 관한 결과이며, model.predict(np.array(vec))[0]은 DNN1과 DNN2의 결과이다.

Ⅳ. 결과 및 비교 분석

본 연구는 윈도우 10 기반에 모델 구축을 위한 코드는 파이선 3.8로 구축하였으며, 얼굴 검출과 128D 임베딩 벡터를 생성하기 위해서 OpenCV 4.3과 Caffe를 사용하였다. 또한, 딥러닝 학습을 위해 tensorfolw 2.3과 keras를 사용하였다.

학습을 위한 얼굴 영상 데이터는 9개의 폴더에 구축하고, 각 폴더명은 얼굴 영상의 인물 이름으로 저장하였다. 또한, 8개 폴더 안에는 동일한 인물의 사진을 10장씩 저장하고, 1개의 폴더는 9개 폴더 안의 인물과는 상관없는 인물의 사진을 저장하였다. 테스트를 위한 영상 데이터는 각 인물당 5장씩 40장의 사진을 이용하였다.

본 연구에서는 인식률 향상을 위해 얼굴 영상에 대해 좌우 눈을 랜드마킹과 얼굴 정렬을 하였다. 그림 7에 랜드마킹 및 얼굴 정렬에 대해서 나타냈다.



(a) (b) (c)

그림 7. 얼굴 정렬과 검출
Fig. 7. Face Alignment and detection.

그림 6과 같이 기울어진 얼굴을 68개의 점으로 랜드마킹하여 얼굴의 구성요소를 찾고, 그중에서 오른쪽 눈과 왼쪽 눈의 중심을 구하여 삼각형을 나타내면 기울어진 각도를 구할 수 있다(a). 따라서 각도만큼 얼굴을 회전하고 배율을 계산함으로써 얼굴을 정렬할 수 있다(b). 이를 바탕으로 얼굴을 검출할 수 있다(c). 다음은 128D 임베딩 벡터 결과에 대해서 나타냈다.

```
array([-0.02032343,  0.07803919, -0.07747763, -0.05468843,
0.04102494, 0.10063157, -0.00340556, 0.00961062, -0.06171633,
0.11349507, -0.00206275, 0.09730095, 0.0504887, -0.14516363,
-0.03343717,
.....
0.04221208, -0.04162304, -0.22655636, -0.00211026, 0.15095025,
0.05423078, 0.03935959, -0.01428195, 0.02969432, -0.0558367,
0.04571818, 0.0595092, 0.05269194], dtype=float32)]
```

본 연구에서는 이처럼 모든 얼굴 영상에 대해서 128D 임베딩 벡터와 폴더 이름을 ID로 하여 one-hot 인코딩을 구하여, SVM과 DNN의 입출력으로 하여 학습하였다. 또한 학습된 모델을 이용하여 테스트를 위해 새로운 ID의 얼굴 영상에 대해서 얼굴 정렬, 추출, 임베딩 벡터 추출하여 얼굴 인식을 하였다

본 연구는 특정 얼굴 영상을 테스트할 경우의 인식률과 예측률을 구하였다. 인식률은 특정 인물의 ID를 정상적으로 또는 비정상적으로 인식할 확률이며, 예측률은 특정 인물의 ID를 정상적으로 예측할 확률이다.

표 1. SVM과 DNN 예측률
Table 1. SVM and DNN prediction rate

ID	linear	poly	rbf	DNN1	DNN2
iID6_1	45.60%	32.35%	42.33%	62.62%	99.07%
iID6_2	50.68%	79.50%	75.26%	74.44%	99.99%
iID6_3	81.24%	76.23%	79.43%	97.96%	100%
iID6_4	54.77%	51.94%	54.61%	80.63%	72.10%
iID6_5	73.45%	87.41%	85.77%	96.30%	100%
iID7_1	44.51%	54.63%	49.87%	65.84%	99.97%
iID7_2	57.68%	62.10%	60.18%	84.50%	99.98%
iID7_3	68.81%	77.92%	76.67%	93.14%	100%
iID7_4	52.67%	49.79%	52.14%	77.58%	100%
iID7_5	65.26%	57.83%	62.06%	92.98%	99.99%
average	59.47%	62.97%	63.83%	82.60%	97.11%

표 2. SVM과 DNN 예측률
Table 2. SVM and DNN recognition rate

Name	SVM			DNN1	DNN2
	Linear	Poly	rbf		
ID1	100%	100%	100%	100%	75%
ID2	100%	100%	100%	100%	50%
ID4	100%	100%	100%	100%	100%
ID4	100%	75%	100%	100%	100%
ID5	75%	100%	100%	100%	100%
ID6	100%	100%	100%	100%	100%
ID7	100%	100%	100%	100%	100%
ID8	75%	100%	100%	100%	100%
Mean	93.75	96.88	100.00	100.00	90.63
Error / Total image	2/40	1/40	0/40	0/40	4/40

표 1에 두 사람에 대한 예측률 결과로서, 서로 다른 테스트 영상을 예측한 확률에 대해서 나타냈다. 두 사람에 대해서는 모든 테스트 영상을 제대로 예측하였으나 알고리즘마다 예측률이 달랐다. 즉, 예측률 평균이 SVM의 linear는 59.47%, poly는 62.97%, rbf는 63.83%, DNN1은 82.60%, DNN2는 97.11%였다. 본 실험으로 SVM보다는 DNN 알고리즘이 정상적인 예측에서 더 높은 예측률을 얻을 수 있음을 확인하였다.

표 2에 각 ID 별로 인식률 결과를 나타냈다. 8명에 대해서 알고리즘별로 학습을 하였으며, 각 사람 당 5장의 사진을 테스트함으로써 총 40장을 테스트하였다. 실험 결과로서 인식률 평균이 SVM의 linear는 93.75%, poly는 96.88%, rbf는 100%, DNN1은 100%, DNN2는 90.63%였다. 또한 오 인식 수는 총 40장 중에서 linear는 2장, poly는 1장, rbf는 0장, DNN1은 0장, DNN2는 4장이었다.

제안 모델은 SVM보다는 DNN으로 학습한 결과가 우수한 예측률과 인식률을 보였다. 그러나 DNN의 중간층을 증가하게 되면 예측률은 높아지나 인식률이 감소하는 현상이 발생하였다. 이것은 인식하고자 하는 대상이 적음으로써 발생하는 과적합으로 판단된다. 본 연구의 성능을 검증하기 위해 3명의 얼굴 영상을 추가하여 검증한 결과 첫 번째 얼굴 영상은 SVM의 linear는 90.34%, poly는 95.32%, rbf는 82.75%, DNN1은 97.27%, DNN2는 100%였으며, 두 번째 얼굴 영상은 SVM의 linear는 43.33%, poly는 54.61%, rbf는 52.59%, DNN1은

(a) linear (b) poly (c) rbf (d) DNN1 (e) DNN2

그림 8. 인식 결과
Fig. 8. Recognition result

84.76%, DNN2는 100%였다. 또한 세 번째 영상은 SVM의 linear는 31.21%, poly는 30.23%, rbf는 33.28%, DNN1은 45.49%, DNN2는 99.31%였다. 그림 8은 본 제안 모델에 첫 번째 영상의 결과를 나타냈다. 제안 모델은 실험을 통해 다음과 같은 결과를 얻을 수 있었다.

1. 더욱 좋은 영상과 많은 영상이 사용에 따라 예측률과 인식률의 변화가 다소 있었다. 영상 분야에서 좋은 영상과 많은 영상이 필요하다는 것을 알 수 있다.
2. 제안 모델은 파이프라인으로 구성함으로써 얼굴 검출, 임베딩 벡터 및 라벨 추출, 인공지능을 이용한 학습, 얼굴 인식 모듈을 분리함으로써 모듈별로 별도로 성능을 향상할 수 있다.
3. 얼굴 인식 부분은 전 단계에서 처리된 피클 파일을 이용하면 신속하게 인식처리를 할 수 있으므로 동영상이나 비디오 입력을 실시간으로 처리할 수 있다.
4. IOT 시대에 많이 사용하는 성능이 다소 떨어지는 임베디드 시스템에 적용할 수 있다. 즉, 처리 속도가 빠른 기존의 Harr cascade 얼굴 검출 알고리즘 등을 사용하면 버퍼링 없이 실시간으로 처리 가능한 IOT 시스템이 가능하다.

Ⅴ. 결 론

제안 모델은 딥러닝 플랫폼인 Caffe을 이용하여 얼굴 인식 파이프라인을 구축하였다. 얼굴 검출 부분과 얼굴 특징 추출 부분은 기존 CNN 플랫폼을 이용하였으나 특징 추출 데이터를 학습을 딥러닝으로 구축하여 학습하였다. 또한, 이를 기존의 SVM과 비교 하였다, 실험 결과 SVM보다는 DNN으로 학습한 결과가 다소 우수한 예측률과 인식률을 보였다. 그러나 실험에서 DNN의 중간층을 증가하게 되면 예측률은 높아지나 인식률이 감소하는 현상이 발생하였다. 이것은 인식하고자 하는 대상이 적음으로써 발생하는 과적합으로 판단된다. 얼굴 인식 분야는 앞으로도 계속 진화할 것으로 예상되며, 같은 패턴을 갖고 있는 얼굴 영상에서보다 더 정교한 특징 추출 알고리즘에 관한 연구와 특징 데이터를 효과적으로 학습할 수 있는 딥러닝 구조에 관한 연구가 필요하다.

References

[1] L. Yann, Y. Bengio, and G. Hinton, “Deep learning,” *Nature*, Vol. 521, No. 7553, pp. 436-444, 2015.

[2] V. Paul, and M. Jones, “Rapid object detection using a boosted cascade of simple features,” in *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Kauai: Hi, Vol. 1, pp. 511-58, 2001.

[3] S. S. Farface, M. Saberian, and L. J. Li, “Multi-view face detection using deep convolutional neural networks,” in *Proceedings of the 5th ACM on International Conference on Multimedia Retrieval*, Seattle: WA, pp.643-650, 2015.

[4] H. Li, Z. Lin, X. Shen, J. Brandt, and G. Hua, “A convolutional neural network cascade for face detection,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Boston: MA, pp. 5325-5334, 2015.

[5] R. Joseph and A. Farhadi, “Yolo9000: better, faster, stronger,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Honolulu: HI, pp. 7263-7271, 2017.

[6] S. Ren, K. He, R. Girshick, and J. Sun, “Faster R-CNN: towards real-time object detection with region proposal networks,” *Advances in Neural Information Processing Systems*, Montreal: Canada, pp. 91-99, 2015.

[7] K. Zhang, Z. Zhang, Z. Li, and Y. Qiao, “Joint face detection and alignment using multitask cascaded convolutional networks,” *IEEE Signal Processing Letters*, Vol. 23, pp. 10, 1499-1503, 2016.

[8] Y. Sun, X. Wang, and X. Tang, “Deep convolutional network cascade for facial point detection,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Portland : OR, pp. 3476-3483, 2013.

[9] J. Zhang, S. Shan, M. Kan, and X. Chen, “Coarse-to-fine Auto-encoder networks (CFAN) for real-time face alignment,” *European Conference on Computer Vision*, Zurich: Switzerland, pp.1-16, 2014.

[10] Y. Taigman, M. Yang, M. Ranzato, L. Wolf, “Deepface: closing the gap to human-level performance in face verification,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Columbus : Ohio, pp. 1701-1708, 2014.

[11] Y. Sun, X. Wang, X. Tang, “Deep learning face representation from predicting 10,000 classes,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Columbus: Ohio, pp. 1891-1898, 2014.

[12] Y. Sun, X. Wang, X. Tang, “Deep learning face representation by joint identification-verification,” A*dvances in Neural Information Processing Systems*, Montreal: Canada, pp. 1988-1996, 2014.

[13] E. A. Zanaty, S. H. Aljahdali, and R. J. Cripps, “Accurate support vector machines for data classification,” *International Journal of Rapid Manufacturing,* Vol. 1, No. 2, pp. 114-127, 2009.

[14] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell, “Caffe: convolutional architecture for fast feature embedding,” in *Proceedings of the 22nd ACM International Conference on Multimedia*, Orlando: Florida, pp. 675-678, 2014.

박 진 환 (Jin-Hwan Park)
2014 3월 ~ 현재: 가천대학교 에너지IT학과 재학
관심분야 : 딥러닝, IoT, 임베디드 시스템

김 창 복 (Chang-Bok Kim)
1986 2월 : 단국대학교 전자공학화 졸업(공학사)
1989년 2월 : 단국대학교 전자공학과 (공학석사)
2009년 2월 : 인천대학교 컴퓨터 공학과(공학박사)
1994년 ~ 현재 : 가천대학교 IT대학 에너지 IT학과 교수
※관심분야 : 데이터 마이닝, 딥러닝, 강화학습, 사물인터넷