모바일에서의 실시간 동작을 위한 딥러닝 기반 얼굴 검출 및 얼굴 랜드마크 알고리즘 분석 및 구현

손명규, 이상헌, 김현덕
대구경북과학기술원
smk@dgist.ac.kr, pobbylee@dgist.ac.kr, hyunduk00@dgist.ac.kr

Analysis and implementation of a deep learning system for face and its landmark detection on mobile applications

Myoung-Kyu Sohn, Hyunduk Kim, Sang-Heon Lee
DGIST

요 약

딥러닝을 이용한 영상인식은 많은 발전을 이루어왔고 상당한 결과를 보여주고 있다. 좀 더 좋은 인식률을 위하여 딥 네트워크는 점점 깊어지고 복잡해져 왔다. 이러한 복잡해진 네트워크는 GPU의 도움을 받아 데스크탑 상에서는 어느 정도의 실시간 동작을 지원해주고 있다. 하지만 GPU를 활용하지 못하는 대부분의 모바일 상황에서 데스크탑 상에서 설계된 네트워크를 그대로 사용하게 되면 실행 속도 측면에서 매우 낮은 성능을 보여주게 된다. 본 논문에서는 기존에 데스크탑 상에서 설계된 딥 네트워크의 구조를 크게 변경하지 않고 모바일 상에서 실시간으로 동작시키기 위하여 데스크탑과 모바일에서의 성능을 분석하고 최종적으로 분석 결과를 통하여 모바일 상에서 실시간으로 동작하는 인식기를 구현하였다.

Ⅰ. 서 론

본 논문에서는 기존의 데스크탑에서 실시간으로 동작하는 얼굴 검출 모듈을 모바일 상에서 실시간 동작하기 위해 최적화하기 위한 방법을 보여준다. 일반적으로 이러한 딥러닝 알고리즘은 GPU를 가진 데스크탑에서 학습이 되고 학습이 결과물로 나오는 모델파일을 이용하여 학습되지 않은 데이터에 대해 추론을 할 수 있게 된다. 복잡한 알고리즘 가진 인식 알고리즘[1]을 GPU가 없는 모바일 상에서 실시간으로 동작시키는 것은 어려운 일이다. 본 논문에서는 기존에 구현된 네트워크를 활용하여 모바일 상에서 실시간으로 동작시키기 위한 방법을 제안한다.

Ⅱ. 본론

그림 1은 본 논문에서 사용된 얼굴 검출 및 얼굴 랜드마크 검출기의 전체적인 아키텍처를 보여준다. 이 인식기는 입력된 영상에서 얼굴을 검출하고 얼굴에서 중요한 5개의 위치를 찾아주는 검출기이다. 5개의 랜드마크는 눈 2개, 코 1개, 입의 양쪽 끝 1개씩 모두 5개이다. 본 논문에서 구현하고 분석할 네트워크는 Deng[2]이 제안한 얼굴 검출기를 기반으로 하였다. 이 검출기를 구현하고 데스크탑과 모바일에서 각각의 성능을 평가하였다. 그리고 모바일 최적화 후에 성능을 다시 평가하여 비교하였다.

데스크탑에서 설계된 모델을 모바일 상에서 최적화하기 위하여 다음 두 단계를 수행하였다. 먼저 설계된 모델을 간략화 하는 것이다. 본 논문에서는 기존 네트워크의 아키텍처는 크게 변화시키지 않고 네트워크를 간략화 하는 방법을 선정하였다. [2]에서 구현된 네트워크는 백본 네트워크로써 MobileNet[3]을 사용하였는데 이 MobileNet의 네트워크의 구조는 그대로 가져가면서 컨볼루션(Convolution) 레이어 출력의 레이어 수를 줄임으로써 전체 네트워크를 경량화 하였다. 이때 추론 속도와 인식 성능의 트레이드 오프를 고려하여 최적화하는 것이 필요하다. 두 번째는 학습된 모델의 크기 자체를 줄여 계산 속도를 빨리하는 것이다. 이는 데스크탑에서 학습된 모델을 모바일로 변환 시에 적용할 수가 있는데 본 논문에서는 32비트의 파라미터를 가지는 모델을 16비트 파마미터를 가지는 모델로 변환하여 모델 파일의 크기를 감소하였다. 실험에서 이러한 최적화에 대한 추론 속도의 변화를 살펴볼 것이다.

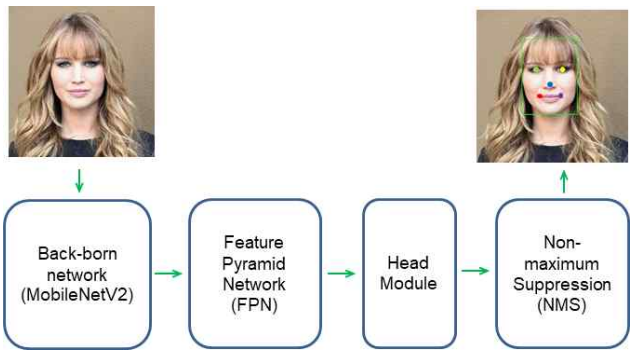

그림 1. 얼굴 검출 및 얼굴 랜드마크 검출기의 전체적인 구조

Ⅲ. 실험 및 결과

실험에 사용된 환경은 다음과 같다. 데스크탑은 Intel Xeon 2.6GHz의 CPU를 가지고 있으며 5120 CUDA코어를 가지는 Nvidia Titan V의 GPU를 장착하였다. 모바일의 경우, Exynos 9820을 가지는 삼성 갤럭시 S10이 사용되었다. CPU를 사용하는 경우 내부적으로 하나의 core만 사용하게 하였다. 딥러닝을 위한 학습 및 추론은 Tensorflow[4] 버전 2.3이 사용되었다.

모바일 상에서의 성능 최적화에 앞서 인식기의 각 모듈별 실행 속도에 대한 성능을 분석하였다. 인식기는 그림 1에서 보이는 것처럼 4개의 모듈

로 구분할 수가 있다. 첫 번째 모듈은 컨볼루션이 주가 되어 이루어진 백본 네트워크이다. 여기에서는 MobileNetV2를 사용하였다. 두 번째 모듈은 얼굴과 랜드마크의 특징 추출을 위한 Feature Pyramid Network[5]으로 되어 있고, 세 번째 모듈은 학습시 파라미터를 업데이트 하기 위한 loss 함수가 정의되어 있다. 네 번째는 검출 결과의 여러 가지 후보 중에서 하나를 선택하는 Non-Max Suppression을 가지고 있다. 표 1은 각 모듈별 추론 시간을 보여준다.

표 1. 데스크탑에서 인식기의 각 모듈별 추론 속도 (ms)

	Back-born	FPN	Header	NMS	Total
with GPU	3.50	1.48	1.37	9.64	15.99
CPU only	105.40	1.72	12.32	43.47	162.90

전체 추론 속도를 보면 GPU를 사용하지 않고 CPU만을 사용했을 때는 GPU를 사용한 것에 비해 10배 이상 추론 시간이 소요되었다. 수행 속도의 차이는 백본 네트워크에서 가장 많이 차이가 나는데 이는 컨볼루션으로 이루어진 네트워크를 GPU가 아주 효율적으로 계산한다는 의미이며 반대로 말하면 GPU가 없이 CPU만을 사용할 때는 이 컨볼루션 네트워크를 최적화해야 전체적인 속도를 올릴 수 있다는 것을 말해준다. 컨볼루션 레이어에서의 속도를 올리기 위해 MobileNet에서 제공하는 width multiplier를 적용하였다. 이 multiplier를 적용하여 콘볼루션의 출력 레이어의 수를 줄여 추론 속도를 높일 수 있다. 표 2는 multiplier를 적용한 추론 시간의 결과를 보여준다.

네트워크 최적화전에 학습된 모델파일의 크기를 줄이는 것을 수행하였으며 본 논문에서는 32비트로 되어 있는 모델파일을 16비트로 줄이는 최적화를 진행하였다. 표 3은 모델 파일 크기의 최적화에 따른 모바일 상에서 백본 네트워크의 추론 시간을 보여준다.

표 2. 모바일을 위해 최적화 된 인식기의 추론 시간 (ms)

Model \ Platform	Desktop		Mobile	
Input size	640	320	640	320
MobileNetV2_1.0 (width multiplier = 1.0)	162.90	53.97	350	96
MobileNetV2_0.35 (width multiplier = 0.35)	92.95	31.97	138	38

표 3. 모바일 상에서 모델 파일 최적화에 따른 추론 시간 (ms)

	32비트 모델파일	16비트 모델파일
MobileNetV2	471.00	350.00

마지막으로 단말에서의 실시간 동작을 위하여 인식 성능을 보장하는 범위 내에서 입력 이미지에 따른 추론 속도의 변화를 측정하였으며 표 2에 나와 있다. 최종적으로, 최적화하기 전의 모델의 경우 모바일에서 471ms가 소요되었으며 최적화 이후에는 추론시간이 38ms로 나타났다. 그림 2는 최적화 전 데스크탑에서 GPU를 사용한 결과와 모바일에 최적화된 인식기의 결과 이미지를 보여준다.

그림 2. 얼굴 및 랜드마크 검출 결과 이미지 (왼쪽: 최적화 전 GPU를 사용한 데스크탑에서의 결과 이미지, 오른쪽: 모델 최적화 후 모바일 상에서의 결과 이미지)

IV. 결론

본 논문에서는 모바일에서 딥러닝 모델을 실시간으로 동작하기 위한 최적화를 통하여 얼굴 검출 및 얼굴 랜드마크 검출기를 구현하고 최적화하였다. 최적화를 위하여 인식기의 각 모듈별 추론 속도에 대한 성능을 분석하였다. 최종적으로 데스크탑에서 GPU를 사용하여야만 실시간으로 동작하는 인식기를 최적화를 통하여 모바일에서 CPU만을 사용하여 큰 인식 성능의 저하 없이 실시간으로 동작을 가능하게 하였다.

ACKNOWLEDGMENT

본 연구는 중소기업벤처부의 기술개발사업(S2860101)과 과학기술정보통신부에서 지원하는 대구경북과학기술원 기관고유사업 (21-IT-02)에 의해 수행되었습니다.

참 고 문 헌

[1] Szegedy, Christian, et al. "Going deeper with convolutions." Proceedings of the IEEE conference on computer vision and pattern recognition. 2015.

[2] Deng, Jiankang, et al. "Retinaface: Single-stage dense face localisation in the wild." arXiv preprint arXiv:1905.00641 (2019).

[3] Howard, Andrew G., et al. "Mobilenets: Efficient convolutional neural networks for mobile vision applications." arXiv preprint arXiv:1704.04861 (2017).

[4] https://www.tensorflow.org

[5] Ren, Shaoqing, et al. "Faster R-CNN: towards real-time object detection with region proposal networks." IEEE transactions on pattern analysis and machine intelligence 39.6 (2016): 1137-1149.