YOLO 기반 영상 비식별화 도구 개발

신형환 [1], 박성완 [1], 박상현 [1], 오치민 [2], 김승원 [1]*
[1] 전남대학교 소프트웨어공학과
[2] 세이프모션
140407, 165513, 185905, seungwon.kim@jnu.ac.kr[1], oh@safemotion.kr [2]
*: 교신저자

YOLO-based Video Non-identification Tool Development

Hyeong-Hwan Shin[1], Sung-Wan Park[1], Sang-Hyun Park[1], Chi-Min Oh[2], Seungwon Kim[1]*
[1]Dept. of Software Engineering, JNU University
[2]Safemotion

요 약

영상 매체의 발달과 영상 미디어의 쉬운 공유는 많은 이점을 가지고 왔다. 하지만 영상이 인터넷 상에서 쉽게 공유되면서 개인이 원치 않는 모습 및 정보가 자신도 모르게 공개되는 초상권 문제나 사생활 침해 문제가 발생하고 있다. 이를 막기 위해 영상의 인물을 비식별화 하고 있지만 수작업으로 진행되는 영상의 비식별화는 많은 시간과 비용이 들어간다. 이에 본 논문에서는 자동으로 영상의 인물을 탐지, 추적하여 비식별화 영상처리를 진행할 수 있는 YOLO 기반 비식별화 시스템을 제안한다.

1. 서론

1.1 연구배경 및 목적

영상매체의 발달에 따라 각종 영상들이 언론에서 보도되며 다수의 기관에서 사용되고 있다. 영상을 통해 얻을 수 있는 정보량이 늘어났으며 많은 기관에서 녹화된 영상 등을 통해 사건 해결 및 영상이 찍혔던 당시의 상황을 파악하고 있다. 하지만 개인의 초상권 문제, 사생화 침해 문제등을 이유로 동영상 사용에 제약을 받고 있다. 이러한 문제를 해결하기 위해 영상의 비식별화가 필요하며 현재는 많은 시간과 비용이 들어가는 수작업으로 진행되고 있는 실정이다.

이에 본 논문에서는 영상에서 자동으로 인물을 탐색 및 추적하여 비식별화 처리를 할 수 있는 영상 인물 비식별화 시스템을 제안하고자 한다. YOLO 라이브러리와 Deep Sort 추적 알고리즘을 기반으로 객체의 탐색과 추적을 진행하며, Django 프레임워크를 사용하여 시스템을 사용자에게 비식별화 서비스를 제공할 수 있도록 구현한다.

2. 관련 연구

2.1 YOLO

YOLO(You Only Look Once)는 딥러닝 모델을 사용한 실시간 객체 탐지 시스템이다 [1]. 다른 딥러닝 모델들(CNN [2])과는 다르게 YOLO 는 이미지 전체를 한번만 살펴보기 때문에 실시간 탐지에 적합한 빠른 속도를 가지고 있다.

2.2 Deep Sort

딥솔트는 솔트 기법에 딥러닝 feature vector 을 합친 추적 알고리즘 [3]으로써 Kalman Filter 를 통한 객체 탐지 예측값[4]에 hungraian Algorithm [5]을 사용해 실시간 추적을 가능하게 한다 [6].

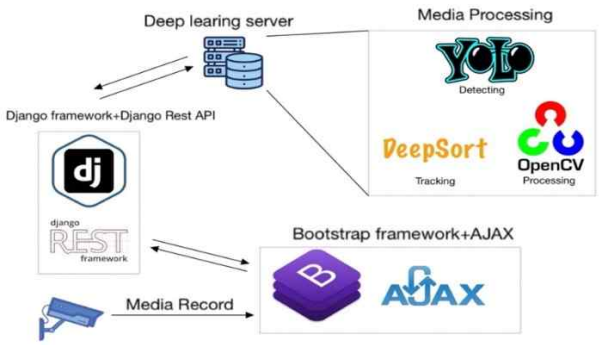
2.3 Django

파이썬 오픈소스 웹 프레임워크로써 데이터베이스 기반 웹 사이트를 쉽게 개발할 수 있고, 재사용성이 탁월하여 본 논문에서 설계한 비식별화 시스템을 쉽게 제공할 수 있다고 판단하였다 [7].

3. YOLO 기반 영상 비식별화 도구 연구 설계

3.1 시스템 설계 및 흐름

그림 1 은 비식별화 시스템의 설계를 나타낸다.



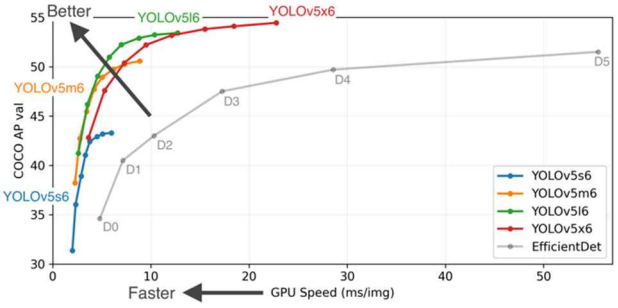
(그림 1) 시스템 구성도

YOLOv5 [7]와 딥소트 알고리즘으로 객체의 탐색과 추적을 진행하고 추적된 객체를 OpenCV 를 통해 영상처리하는 백엔드 서버를 Django 프레임워크를 사용해 설계하였고 해당 백엔드 서비스를 REST API 를 통해 제공하였다. 이를 부트스트랩으로 구현한 프론트엔드에서 사용자에게 재공한다.

사용자는 프론트엔드에서 API 를 통해 영상을 딥러닝 서버로 전송하면 백엔드 서버는 YOLO 와 딥소트 알고리즘을 사용해 탐색과 추적을 수행한다. 이후 사용자의 선택에 따라 특정 객체들의 비식별화 여부를 결정하여 영상을 비식별화 처리하여 API 를 통해 사용자에게 되돌려준다.

3.2 YOLO 모델 설정

시스템은 웹에서 제공하고 서버의 자원을 소모하므로 YOLO 의 네트워크 모델은 가볍고 빨라야 한다. 그림 2 은 각각의 Pretrained 모델을 비교하고 있다.



(그림 2) Pretrained 모델의 비교

이러한 자료를 바탕으로 성능과 속도가 좋은 YOLOv5l6 모델을 시스템에 사용하기로 하였다.

4. 시스템 구현

4.1 시스템 구현 환경

본 시스템은 소프트웨어 이식성을 증가시키기 위해 도커 환경에서 구현하였다. 도커에서 공식 이미지로 지원해주는 python3.7 이미지 파일 위에 python 언어를 기반으로 하는 Django 프레임워크를 설치하여 백엔드 API 시스템을 구현하였다.

4.2 Media Processing 구현

YOLO 시스템을 사용해서 객체의 Bounding box 좌표를 탐지할 수 있고, 이를 Deep Sort 알고리즘에 대입하여 객체를 추적할 수 있도록 하였다.

추적된 고유의 id 와 box 좌표는 Opencv 라이브러리에서 비식별화 과정을 거친다.

4.3 비식별화 구현

이미지의 크기를 줄였다가 다시 본래의 크기로 늘리면 해당 이미지가 줄어들었을 때 사라진 픽셀을 복구할 수 없기 때문에 손상이 가해진다. 이 원리를 Opencv 라이브러리의 resize 메소드를 통해 구현하였다 .

4.4 백엔드 딥러닝 서비스 API 구현

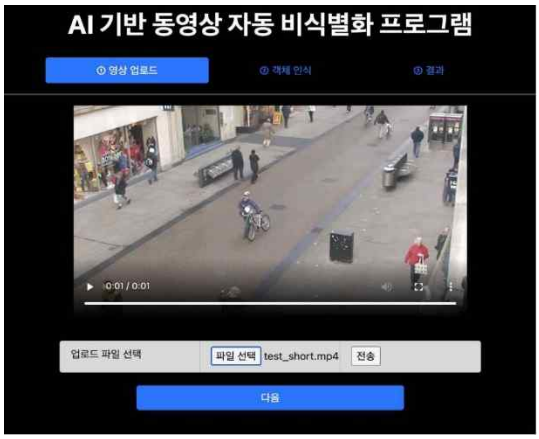
Django REST Framework 를 이용해 프론트엔드에서 사용할 수 있는 4 가지 API 를 구현하였고, API 의 기능은 다음 표 2 와 같다.

<표 2> API 의 기능

API name	Discription
upload	영상을 사용자로부터 업로드(POST)
detector	업로드된 영상을 비식별화 모듈을 통해 탐색, 추적하여 id 부여 후 id 별 이미지 리턴(POST)
non_idt	비식별화 할 id 를 form 데이터로 입력(POST)
download	form 데이터로 입력받은 id 를 바탕으로 비식별화 처리 후 영상 파일을 리턴(GET)

4.5 프론트엔드 구현

부트스트랩 프레임워크를 사용해 백엔드 API 를 사용할 수 있는 3 개의 화면으로 구성된 간단한 웹 페이지를 만들었다 (그림 3-1, 3-2, 3-3 참조).



(그림 3-1) upload.html

사용자가 비식별화 처리를 진행할 영상을 API 를 통해 업로드하여 백엔드 서버로 전송한다.

(그림 3-2) list.html

영상을 전송받은 백엔드 서버는 탐색된 객체에 고유한 id 를 부여하여 사용자에게 어떤 객체를 비식별화 할 것인지 선택하게 한다.

(그림 3-3) download.html

사용자의 선택에 따라 비식별화 처리가 완료된 영상을 확인하고 다운로드 받을 수 있다.

5. 결론

영상의 비식별화는 수작업으로 진행되고 해당 과정에서 많은 시간과 비용손실이 발생한다. 이에 본 논문에서는 비식별화 작업을 자동화 할 수 있는 YOLO 기반 비식별화 시스템을 개발하였다.

본 시스템을 사용할 경우 본래 수작업으로 처리하여야 할 비식별화를 사용자가 선택만 하면 영상에서 해당 선택에 따라 자동으로 비식별화가 진행되므로 시간과 비용면에서 수작업보다 많은 이득을 볼 수 있을것이다.

구현된 결과를 바탕으로 탐색할 수 있는 객체의 범위를 늘리고 나아가 OCR(문자인식) 기능을 추가하여 차량의 번호판이나 전화번호 등 인물 이외의 정보를 비식별화 할 수 있는 시스템을 만드는 연구를 진행할 것이다.

사사표기

“본 연구는 2021년 과학기술정보통신부 및

정보통신기획평가원의 SW중심대학의 연구결과

수행되었음”(2021-0-01409)

참고문헌

[1] Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). You only look once: Unified, real-time object detection. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 779-788).

[2] Li, Y., Hao, Z. B., & Lei, H. (2016). Survey of convolutional neural network. Journal of Computer Applications, 36(9), 2508-2515.

[3] Hwang, J. J., & Liu, T. L. (2015). Pixel-wise deep learning for contour detection. arXiv preprint arXiv:1504.01989.

[4] Welch, G., & Bishop, G. (1995). An introduction to the Kalman filter.

[5] Wright, M. B. (1990). Speeding up the Hungarian algorithm. Computers & Operations Research, 17(1), 95-96.

[6] Nicolai Wojke “SIMPLE ONLINE AND REALTIME TRACKING WITH A DEEP ASSOCIATION METRIC”

[7] Forcier, J., Bissex, P., & Chun, W. J. (2008). *Python web development with Django*. Addison-Wesley Professional.

[8] Alexey Bochkovskiy "YOLOv4: Optimal Speed and Accuracy of Object Detection "