ASLC : American Sign Language Classification

조원*, 한대찬*, 이미연*, 박준희*, 이규원**

세종대학교 지능기전공학부*, 컴퓨터공학과**

clockjow@gmail.com, immioriii@gmail.com, julia980406@gmail.com,

gkseocks2@gmail.com, duffufk1202@naver.com

요 약

본 논문에서는 청각 장애인들이 생활속에서 겪는 의사 소통의 문제를 해결하고자 연구를 진행하였다. Dataset은 자체 제작해 Single-RGB camera로 촬영 하였고 Dataset의 다양화를 위해 4명의 팀원이 각각 한 Class당 10개의 영상을 2번에 걸쳐 촬영했다. 영상은 각 4~7초로 110~200 frame으로 이루어져 있다. 추출된 영상은 openpose를 이용해 손의 위치를 추정한다. Openpose는 Human Pose Estimation 라이브러리로 손의 위치를 21개의 key point로 coordination을 전달 해 준다. 이렇게 얻게된 손의 위치로 손의 양 손의 local data 를 만들어 낸다. 제안하는 방법은 global data(전체 영상) 과 local data로 나뉜 영상들을 각각 8 frame 씩 겹쳐진 16 frame 씩의 batch로 나뉜다. 그 후 Conv3d Network를 통해 4096의 feature를 추출해 낸다. 이를 LSTM model에 넣고 fully connected layer를 거친 후 20차원의 class에 대한 점수와 ground truth의 차이를 최소화 시키는 방향으로 학습하였다. 하지만 성능이 높아질 것이라 예상했던 Global + Local이 성능이 조금 더 낮게 나왔다. 그 원인에 관한 분석, C3D 모델과 C3D + LSTM 모델의 정확도 차이점에 대해 결론을 내었다.

1. 서론

장애인들의 생활 개선 문제는 지속적인 사회적 이슈로 크게 대두되고 있다. 청각장애인의 경우 대부분의 업무에서 가장 중요시되는 의사소통에 문제가 되기 때문에 사회적으로 더욱 소외되는 경향이 있다. 그런 청각 장애인들과 비 장애인들과의 소통을 위해 수화를 단어로 바꾸는 방법을 제시한다.

수화를 단어로 바꾸는 기존 연구중 [2]를 참고하여 연구를 진행하였다. [2]에서는 3D Convolutional network 이하 C3D를 통해 Clip당 features를 추출하여 단어를 찾은 후 그 단어들을 문장으로 학습하는 데 LSTM을 사용하였다. 하지만 본 연구팀은 분류에 관한 문제를 해결 하고자 하므로 C3D이후를 clip을 LSTM에 넣어 마지막 fc layer 통해 grooud truth와의 loss를 구하는 방식으로 연구를 진행 하였다. 자세한 내용은 3절에서 다뤘다.

2. Dataset

본 논문에서 필요로 하는 Dataset은 RGB 영상에 OpenPose 활용을 위해서 손이 정확히 나오고, 변인으로 사람과 수화 동작의 종류만 작용하는 영상이다. 하지만 기존 인터넷에 있는 Dataset들은 Class 당 영상의 수가 적고, 필요하지 않은 변인이 등장하거나 영상이 본 논문에서 필요한 형태와 달랐다. 따라서 본 연구팀은 학습에 필요한 Dataset을 직접 촬영하기로 했다.

전반적인 Dataset의 처리 과정은 촬영한 영상들에서 왼손과 오른손의 위치를 각 Frame마다 저장하여 OpenPose를 통해서 detect하고, 마지막으로 영상을 정규화하여 학습 코드에 전해준다. 이 장에서는 Dataset의 촬영부터 활용, 전처리까지의 과정을 상세히 서술한다.

2.1 Dataset 촬영

그림 1. 촬영된 Dataset 영상 frame

Dataset 촬영하는데 가장 중요하게 고려되어야 하는 사항은 크게 2가지가 있다. 첫번째는 Data 영상에서의 변인을 조절하는 것이다. 모든 영상에서 사람의 형체는 상반신만 등장하도록 했고, 영상의 배경과 소품을 통일했다. 따라서 본 Dataset은 동일한 배경에서, 등장하는 사람과 각 Class에 따른 수화 동작만 변하는 영상을 촬영하여 활용하였다.

두번째는, 영상의 수이다. 총 20개의 수화 동작을 각각의 Class로 지정하여 한 Class당 40개의 영상을 촬영했다. 모든 Class의 영상은 과적합(Overfitting) 을 피하고 Dataset의 다양화를 위해 4명의 팀원이 각각 한 Class당 10개의 영상을 2번에 걸쳐 촬영했다. 따라서 본 논문에서는 총 800개의 영상이 Data로 저장되었다. Data 영상 중에서 각 Class당 30개, 즉 600개의 영상을 train data로 학습시키는데 이용하였고, 나머지 200개의 영상을 test data로 활용하였다. 모든 Data 영상은 frame 단위로 나눠서 적용하였는데 frame 추출에 관련해서는 다음 절에서 다룬다.

2.2 Frame 추출

각 영상마다 Frame을 추출하기 위해서 opencv-python을 이용하였다. 4~7초가량의 Data 영상을 1초당 30 frame씩 추출하여 한 영상 당 약 110~개의 frame을 저장했다.

표 1. Data들의 저장 수

class 수	class내 영상 데이터 개수	총 frame 수
20	40	110~200

2.3 Openpose

최근 몇년 Computer Vision 학술 대회에서는 Human Pose Estimation에 관한 연구에 큰 진척이 있었다. 2017년 사람의 관절 key point를 추출해 전체적인 관절과 세부적으로는 손모양을 추출 해 낼 수 있는 Openpose가 발표되었다. 그 이후로 2018년엔 Densepose로 단지 single rgb 카메라로 depth 를 추정해 낼 수 있고, 최근 2019 CVPR에서는 가려진 신체부위의 자세를 거의 Real time 으로 추정해내는 모델까지 나왔다. 이러한 Human pose estimation 기술을 Sign Language Classification에 적용하기 좋은 이유는 수화의 의미전달은 Human Pose에서 기인하기 때문이다. Human pose에 Attention을 주는 방식으로 Accuracy를 높일 수 있을거라 기대로 도입했다. 현재 연구에서는 openpose는 hand position을 추출해 전체 모델에서 손의 모양에 attention 을 주기 위해, 사람에게서 손의 위치를 추론해 내는 데에 사용되었다. openpose 라이브러리로 수화를 하는 사람의 영상을 받아들이면고 손에 관한 정보를 얻으면, 손의 관절 key point를 21개의 2-dimensional coordinates를 왼손과 오른 손 양쪽을 따로 받아 낼 수 있다.

오른손 왼손이 따로 Tracking 되는것은 Human pose를 기반으로 tracking을 하기 때문에 갖는 특징이다. 다른 데이터셋을 이용한 학습 모델에서는 오른손 왼손의 파악이 어려워 팔이 교차되는 시점에 혼동하지 않기 위해 Hand Tracking 을 지속적으로 해야만 한다는 면에서 openpose는 차별화 된다. 21개의 coordination의 average coordination 을 구하게 되면 대략적인 양쪽 손의 중심을 알 수 있고 이 coordination을 바탕으로 attention을 줄 수 있는 local image를 얻을 수 있다.

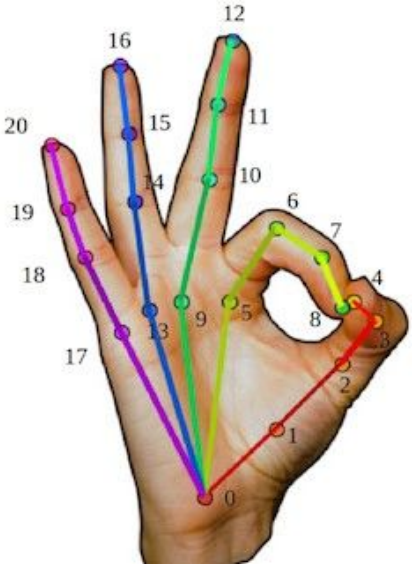


그림 2. Openpose hand-detect key point

본 연구에서는 pytorch-openpose 버전을 이용하여 hand-detect를 진행했다. Colab환경에서 영상 frame들을 Google Drive에 저장하고 이와 연동하여 frame별 detect를 실행했다. 각 Class마다 frame 별 hand-detect가 진행되면서 21개의 손 관절 key point에 대한 정보를 x,y좌표로 나타내어 출력하도록 했다. 이때, 모든 key point에 대한 좌표가 잡히지 않으면 정보가 없는 point의 좌표는 0으로 처리하였고, 좌표 정보를 frame별 hand-detect가 진행될 때 마다 csv 파일을 생성하여 저장했다. 저장된 csv 파일을 가지고 zero-mean regulation을 했는데 보다 자세한 데이터의 전처리 과정은 다음 절에서 다룬다. openpose hand-detect에 사용한 demo.py 코드는 [7]을 참고했다.

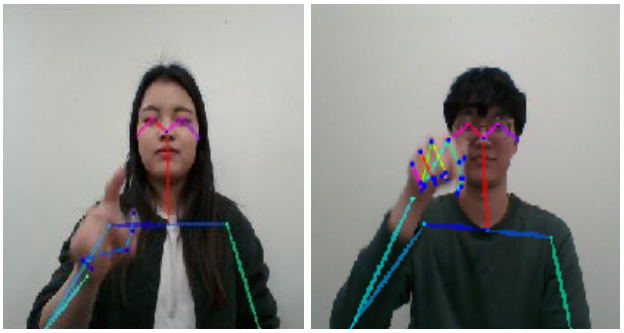


그림 3. Openpose를 통한 hand & body point 검출

2.4 데이터 전처리

얻어낸 영상의 해상도는 1280x720 크기이다. 우리는이를 그대로 사용하지 않고 [2]에서 처럼 128x128의 크기로 resize하였다. Model에 넣을 각 clip을 그림 4 처럼 영상을 1~16을 clip(1), 8~24를 clip(2) …..이렇게 영상을 겹쳐서 clip을 얻어 낸다.

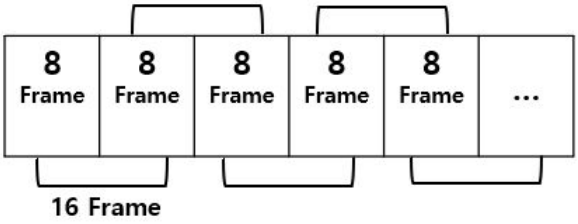


그림 4. one clip 제작법

이렇게 얻어진 clip으로 전처리를 마치게 되면 데이터 셋의 노이즈가 많아서 학습이 잘 되지 않는다. 그렇기에 우리 연구에서는 데이터 셋을 각각 해당 이미지의 모든 픽셀에 모든 픽셀에 대한 평균을 뺀 Zero-mean normalization을 이용했다.

3. 제안하는 방법

본 연구는 수화의 동작을 분류하는 게 목적이다. 수화는 동작의 순서에 따라서 의미가 바뀌기 때문에 우리는 공간의 정보만 담는 Conv2d이 아닌 시간에 대한 정보를 저장하며 학습할 수 있는 Conv3d Network를 선택하였다. 본 연구에서는 기존 3D-CNN 모델에 Sports 1M Dataset으로 기존 pretrain된 model에 촬영한 dataset을 추가로 pretrain 시켜서 우리 데이터 셋에 맞는 pretrained model을 만들었다. 본 pretrained model을 3D-CNN에 적용하여 4096 차원의 feature를 뽑아내고 이를 LSTM model에 넣고 fully connected layer를 거친 후 Logsoftmax activation 통해 20차원의 class에 대한 점수와 ground truth의 차이를 최소화 시키는 방향으로 학습하였다.

3.1 3D-CNN

Convolutional Neural Network는 일정 크기의 kernel을 input 값에 convolution 연산을 통하여 공간적인 정보를 output 으로 준다. 우리는 공간적인 정보 뿐만아니라 시간적인 정보도 output으로 전달할 수 있는 3D-CNN을 사용하였다. 3D-CNN model은 8개의 convolution layer와 5개의 pooling layer를 지니고 있으며 2개의 fully connected layer를 지니고 있다. 각 convolution layer의 kernel의 크기는 3x3x3 이며 이는 [1]에서 실험한 결과 3x3x3일 때 최고의 성능을 낸 것을 통해 정한 것이다. 이 3D-CNN을 통해 16 차원의 한 Clip을 넣어 4096 차원의 feature를 추출하게 된다.

3.2 LSTM

3D-CNN 부분에서 얻어낸 4096차원의 feature를 LSTM model에 넣어 256차원으로 [5]를 참조하여 model을 설계했다. 이렇게 나온 256차원의 feature를 한 층의 fully connect layer를 통해 Dataset의 class의 개수와 같은 수인 20 차원의 feature로 줄였다.

그 이후의 activation function은 Softmax함수대신 Loss funcition에서 옳지 않게 분류했을 때의 값 ground truth와의 차이를 더 크게 나타내어줄 수 있는 Logsoftmax activation function 을 사용했다. Loss function으로는 CrossEntropy와 MSE중 실험적으로 CrossEntropy를 선택하였다.

3.3 Local attention

우리 연구의 성능을 높이기 위해서 openpose를 이용한 hand detect 방식으로 손 영상을 얻어 냈다. 그리고 이 Local 영상과 원래 영상인 Global 영상을 각각 따로 3D-CNN 모델에 넣어 feature를 얻어내고 이를 concat하여 LSTM 모델에 넣어 분류 하였다.

4. 실험 결과 및 분석

실험은 Dual 20-Core intel Xeon CPU 와 32GB 메모리, tesla v100 GPU를 사용 했다. 알고리즘 구현을 위해서 Python과 Pytorch를 딥러닝 구현을 위해서 사용 했다. 데이터 셋은 삼성 노트북 10의 웹캠을 통해서 20 Class를 각 Class 당 40개씩 총 800개의 1280x720 영상을 촬영했다. 촬영한 영상들을 3:1의 비율로 train과 test로 나누어서 실험을 진행했다. 이 영상의 프레임들을 128x128 사이즈 줄이고 정규화 시켜 가공 하였다. OpenPose[10]로 Pretrain된 Model을 사용하여 손의 위치를 뽑아낸다.

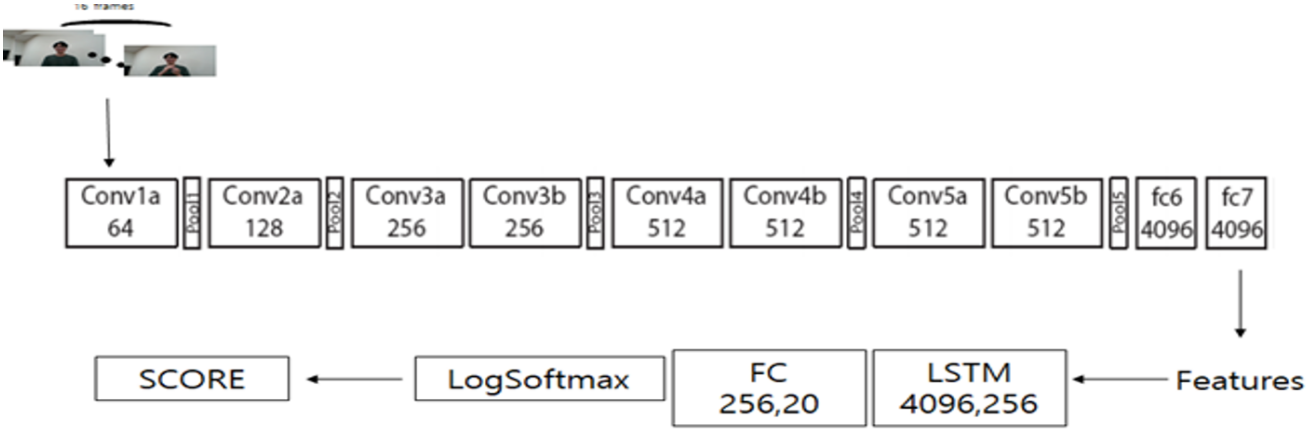
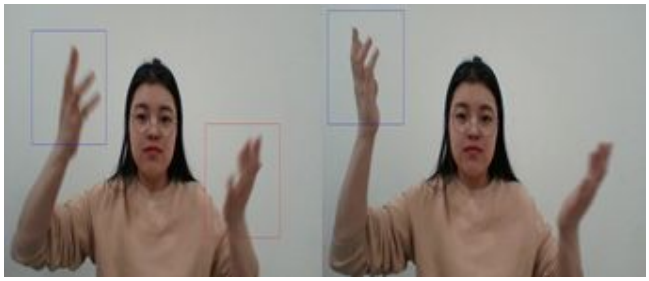


그림 5. CNN 모델



[가] [나]

그림 6. Hand-detect [가] 검출 성공 영상 [나] 검출 실패 영상

손을 뽑은 것을 보면 그림 6.가에서는 손을 정확히 추출했지만 그림 6.나에서는 왼손을 뽑아내지 못했다. 이렇게 손을 찾지 못한 부분들은 128x128 Zeros 로 주었다. C3D를 Base code를 통해서 short term에 대해서 촬영한 데이터셋에 맞게 학습시킨다.

그림 7. C3D pretrain

그림 7은 Short term 에 대해서 학습 시킨 결과 이다. Short-term에 대해 학습한 model을 정성적으로 결과를 확인해 보았다. 그림 8.가는 총 171 frames의 영상중에서 20번째 사진이고, 그림 8.나는 100번째 사진이다. 이 영상의 라벨은 DO_YOU_NEED_HELP 이지만 초반 동작이 얼마 없을 때는 잘못 맞추는 것을

[가](20/171) [나](100/171)

그림 8. C3D 정성적 결과

볼 수 있다. Short-term을 학습시킬때는 영상을 최대 프레임으로 나누지 않고 영상의 최대 프레임보다 약 ¼ 정도로 나누어 영상을 나누어서 학습을 시켜서 Accuracy가 잘 나왔지만 실제로 영상을 최대 프레임으로 나누어서 한 클립마다의 정확도를 봤을때 성능이 낮아지게 되었다. 수화는 동작의 전체를 봐야하므로 이렇게 순간 순간 나타내는 C3D만을 이용한 학습은 맞지 않다는 것을 확인 할 수 있다.

동작의 Long-term을 학습 시키기 위해서 먼저 사진의 Global영역의 feature들을 LSTM을 이용해서 실험을 진행했다. Activation function을 Logsoftmax와 softmax를 바꾸어서 학습을 해보고, Lossfunction을 MSE와 CrossEntropy를 바꿔서 학습해 본 결과 그림 9를 봤을때 와 같이 logsoftmax와 CrossEntropy가 성능이 제일 좋게 나왔다. logsoftmax를 이용했을때 False 값을 더욱 크게 보아서 False positive를 줄일 수 있고 ,Cross Entropy를 이용했을때 실험적 결과로 성능이 더욱 잘 나온 것을 볼 수 있다. model의 성능을 더욱 높이기 위해서 Local의 features를 LSTM에 Global features 와 합쳐서 위에서 성능을 확인한 Loss-funtion과 Activation-function을 사용해서 학습을 진행시켰다.

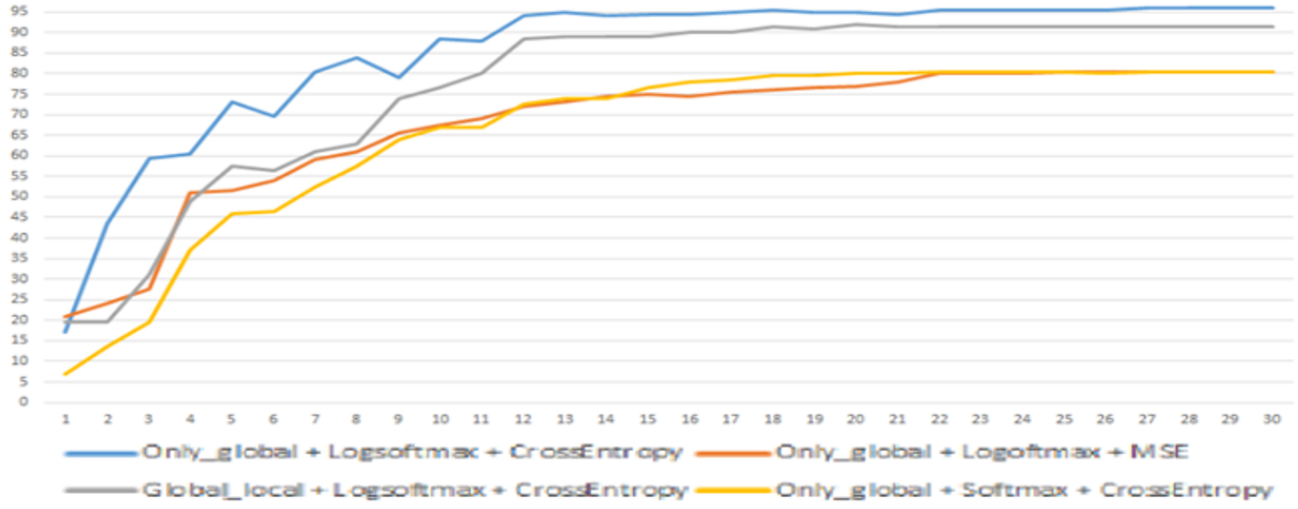


그림 9. Performance

하지만 성능이 높아질 것이라 예상했던 Global+local 이 성능이 조금 더 낮게 나왔다.그래서 local의 성능이 나오지 않은 이유로 3가지의 이유로 분석해 보았다.

첫번째로 위에서 봤듯이 OpenPose를 활용한 손 detect가 중간 중간 안된 것들이 있는 이유이다. 이 detect가 안된 값들을 0으로 처리해버리는 것이 아닌 detect 된 사진들의 평균 값으로 넣어주거나 아님 마지막으로 찾은 손을 넣어주는 등의 처리를 했었어야 조금더 손의 정보를 정확하게 전달해 줄 수 있었을 것이다.

두번째로는 손의 feature 값 만을 넣어준 것이다. 수화는 손의 모양뿐만이 아닌 손의 위치 정보 또한 중요 한 것인데 그 정보를 고려하지 않은 점이다. 이 정보를 LSTM에 같이 넣어주는 방법을 실험 해보는 것도 좋은 제안이 될 것이다.

세번째로는 손의 feature를 C3D pretrain을 따로 시키지 않은 점이다. Global에 대한 영역은 pretrain 시켜서 더욱 데이터셋에 맞는 features를 뽑을 수 있었지만 손들은 Global에 대해 학습 시킨 model로 feature를 뽑아서 손에 맞는 feature를 학습에 넣을 수 없었다. local만 pretrain시킨 model로 feature를 뽑는 것이 맞는 방법이 될 것이다.

표 2. Performance

Methods	Accuracy
Only_global + Logsoftmax + CrossEntropy	0.96
Global_local + Logsoftmax + CrossEntropy	0.915
Only_global + Logsoftmax + MSE	0.805
Only_global+ Softmax + CrossEntropy	0.805

5.결론

본 논문은 OpenPose를 이용해 Local 추출한 영상을 C3D와 LSTM을 이용해 학습하는 수화 분류기를 제안한다. C3D만을 이용했을때 성능이 93%정도로 괜찮게 나오지만 LSTM을 통해 Global 에서의 성능을 96%까지 올려서 더욱 긴 영상에 대해 보다 정확한 성능으로 분류를 할 수 있는 방법이다 . 하지만 처음 제안한 local 을 이용한 방법은 best 약 92%로 기존 C3D와 성능이 살짝 낮지만 Long term 에 대해서 찾는다는 장점이 있다. Local을 이용한 방법의 문제점을 바꿔서 실험을 해본다면 성능을 높일 수 있을 것이라 생각한다.

감사의 글

본 연구는 이를 가능하게 2019 봄학기 인공지능 수업을 진행해 주시고, Github 의 issue 를 통해 지속적인 feedback 을 해주신 지능기전공학부 최유경 교수님의 지원을 바탕으로 수행이 되었습니다. 최유경 교수님께 진심으로 감사드립니다. 또한, 자신의 지식을 오픈소스로 공유함으로써, 연구하는 학생들에게 도움을 주시는 모든 선행 연구자분들에게 감사의 인사를 드립니다.

참고문헌

[1] Tran, D., Bourdev, L.D., Fergus, R., Torresani, L., Paluri, M.: Learning spatiotemporal features with 3d convolutional networks. In: ICCV. (2015) 4489–4497

[2] Huang, Jie; Zhou, Wengang; Zhang, Qilin; Li, Houqiang; Li, Weiping (2018). "Video-based Sign Language Recognition without Temporal Segmentation". arXiv:1801.10111 [cs.CV]

[3] https://github.com/jfzhang95/pytorch-video-recognition

[4] https://github.com/facebook/C3D

[5] https://github.com/ParitoshParmar/C3D-LSTM--PyTorch
[6] Du Tran1,2 , Lubomir Bourdev1 , Rob Fergus1 , Lorenzo Torresani2 , Manohar Paluri1 1Facebook AI Research, 2Dartmouth CollegeLearning “Spatiotemporal Features with 3D Convolutional Networks”
[7] open pose hand picture ref : https://github.com/Hzzone/pytorch-openpose
[8] https://github.com/amarlearning/Finger-Detection-and-Tracking
[9] Pei Xu, Department of Electrical and Computer Engineering, University of Minnesota, Twin Cities “ A Real-time Hand Gesture Recognition and Human-Computer
[10] OpenPose: Realtime Multi-Person 2D Pose Estimation using Part Affinity Fields Zhe Cao, Student Member, IEEE, Gines Hidalgo, Student Member, IEEE , Tomas Simon, Shih-En Wei, and Yaser Sheikh