Camera-Based Table Tennis Posture Analysis
基於相機影像之桌球姿態分析

Hung-Yi Wu, Yu-Hsin Lin, Yu-Wei Chang

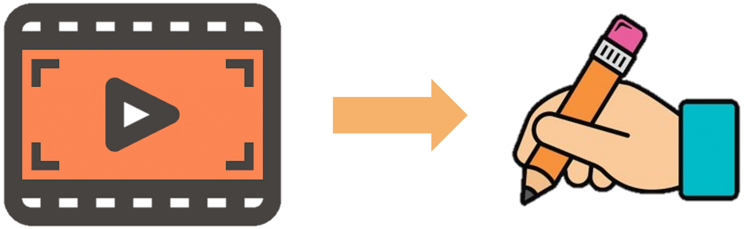
Advisor: Prof. I-Chen Lin

Outline

- The Problem
- Our Solution
- Methodologies
- Evaluations
- Conclusions
- Future Works
- Reference

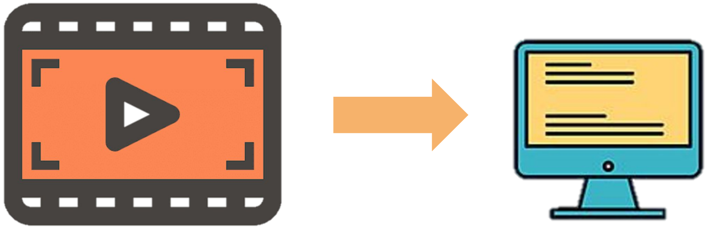
The Problem

- Table tennis players need analyses of their opponents' postures to optimize their game strategies
- Too **laborious** and **time-consuming** to calculate a player's postures **by hands**
- Existing models are **sensor-based**



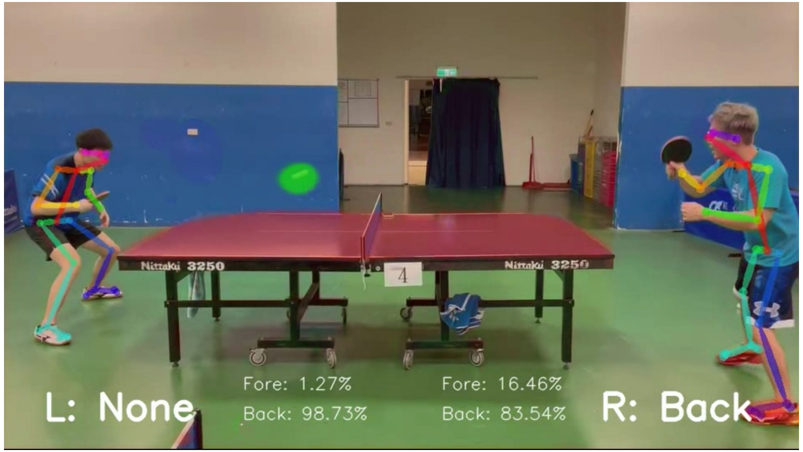
Our Solution

- We built a system to classify players' postures (forehand and backhand) automatically based on their past game and practice videos
- We calculate ratios of players' postures automatically based on the prediction from those classifiers



Methodologies

- Postures Analysis using machine learning algorithm
- Semantic Segmentation for Ball Tracking and Table Detection



Postures Analysis using ML algorithm - Outline

- Data Collection
- Data Preprocessing
  - OpenPose
  - Build Dataset
- SVM
- LSTM

Postures Analysis - Data Collection

- We recorded 8 videos of different players from the side of the tables at 30 fps
- The length of the videos are from 20 seconds to 100 seconds
- The camera moves slightly during the videos, and has different offsets between different videos

Postures Analysis - Data Preprocessing

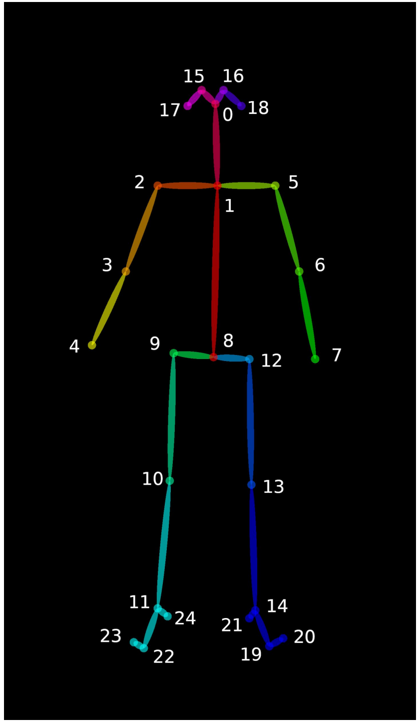
- Training on **images** is not the best approach
    - **Costly and time-consuming** because of the high dimension
    - Easy to be distracted because too many other information are irrelevant to postures
- We train models on data containing **keypoints of body** obtained by OpenPose
    - More **efficient** on both training costs and time
    - **Concentrated** on players’ body motions

OpenPose

- An open source tools authored by CMU Perceptual Computing Lab, and it is the first real-time multi-person system to jointly detect human body, hand, facial, and foot keypoints
- Able to handle image and video inputs
- Able to output labeled images, videos, and JSON files that record key points of human body

OpenPose - Output Skeletons

OpenPose - Output Key Points in JSON Files

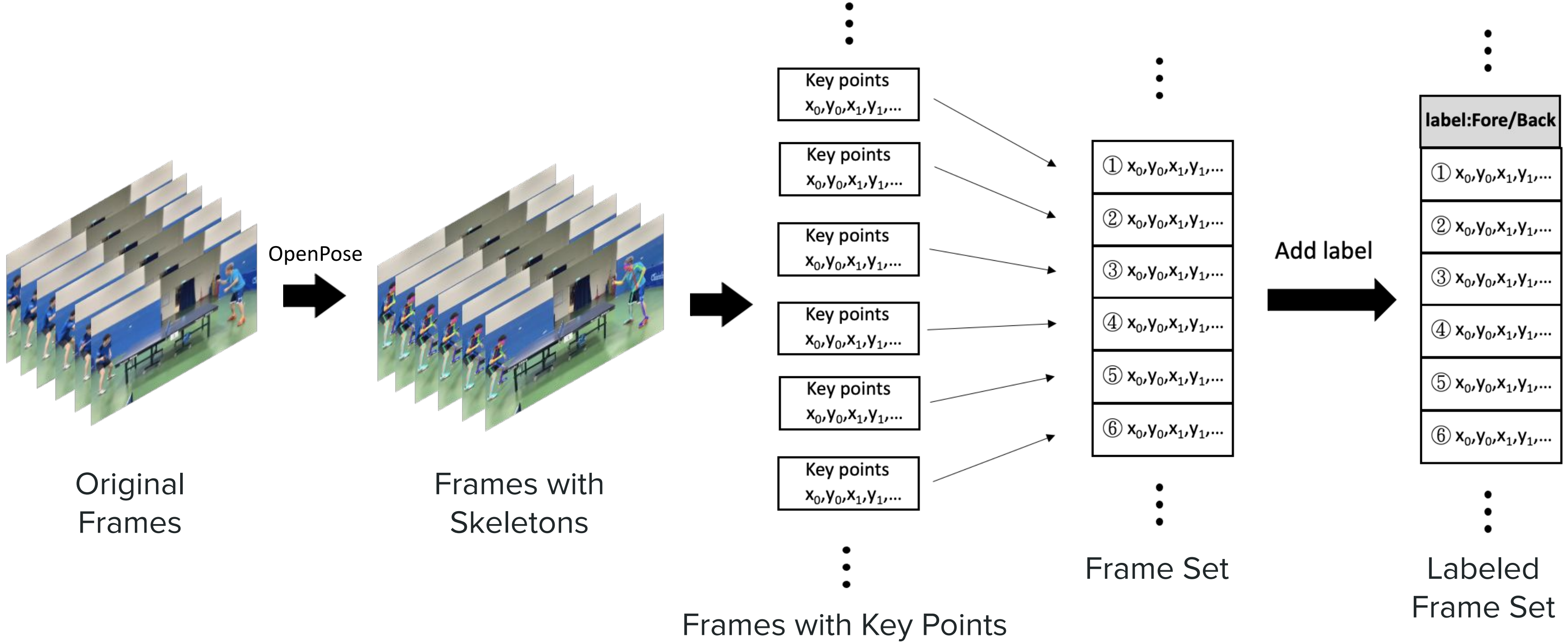



```
"version":1.1,
"people":[
    {
        "pose_keypoints_2d":[582.349,507.866,0.845918,746.975,631.307,0.587007,...],
```

format: x1,y1,c1,x2,y2,c2,...

- x, y: body part locations
- c: confidence in the range [0,1]

Postures Analysis - Data Preprocessing



Build Datasets - Handle Outputs from OpenPose

- OpenPose outputs images with skeletons and key points of players' body in JSON files per frame
- Each human body in each frame has 25 key points where each key point has 3 dimensions, x, y, c
- We collect only x and y from each of the 25 key points, so one person in a frame consists of 2 x 25 features

Build Datasets - Creating Frame Sets

- An action contains multiple continuous frames, so we should convert the data into frame sets, so we combined key points across 6 continuous frames with a same posture into a piece of data, and each frame has 50 (2 x 25) features, so the shape becomes (number of frame sets, 6, 50)
- We then label the data with two classes of forehand and backhand based on a frame sets

Models

- We did experiments on **SVM, CNN, LSTM, and many other models**, but CNN and other models performed not well. The accuracies of those models approximated to the baseline and some were even lower than it
- We chose SVM, which performed the best both on training and test data, and LSTM, which performed well at training data but test data, as the finalist

Support Vector Machine

- We used the SVM to find the maximised margin to fit the data
- We utilized 3 different kernels:
    - RBF
    - Sigmoid
    - Linear

LSTM-Based Model

- We stacked 2 LSTM models and 2 fully connected layers follow
- We used dropout to reduce overfitting

Evaluation - Accuracy

Model	Left Model Accuracy	Right Model Accuracy
SVM-RBF	**89%**	75%
SVM-Sigmoid	75%	57%
SVM-Linear	82%	**95%**
LSTM	88%	57%

Evaluation - Reasoning

- CNN/LSTM model didn’t perforom well
-

Evaluation - Reasoning

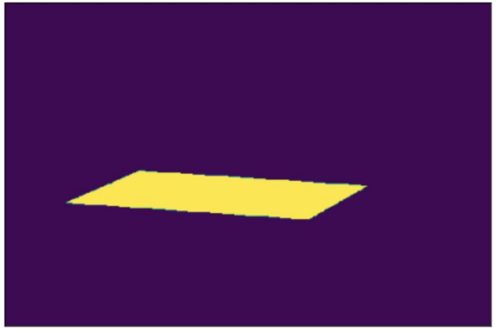
- Most models has better performance on the left side model than on the right side model
- The reason is that players' right arms on the right side are easily blocked by their body (most players are right-handed), so the skeletons are often incomplete

Semantic Segmentation for Balls and Tables - Outline

- Data Preprocessing
  - Data Labeling
  - Data Augmentation
- Transfer Learning
  - EfficientNet

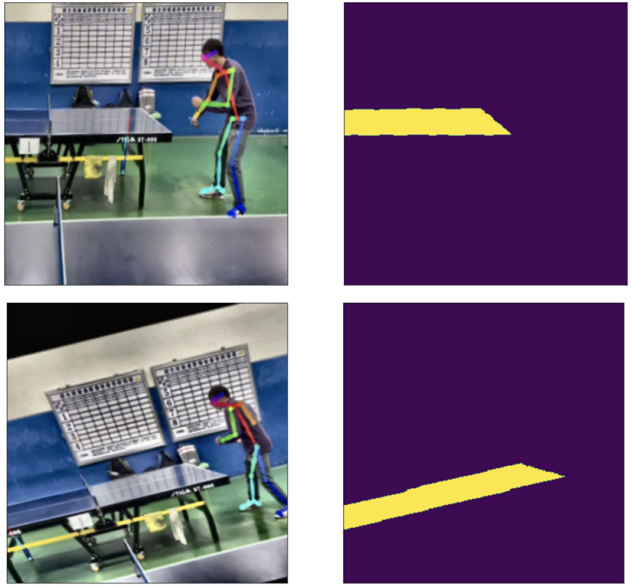
Semantic Segmentation - Data Labeling

- Pairs of images and masks
- images: original images, shape: (H, W, 3)
- masks: labeled masks for table areas with VIA, a labeling tool, shape: (H, W, 1)



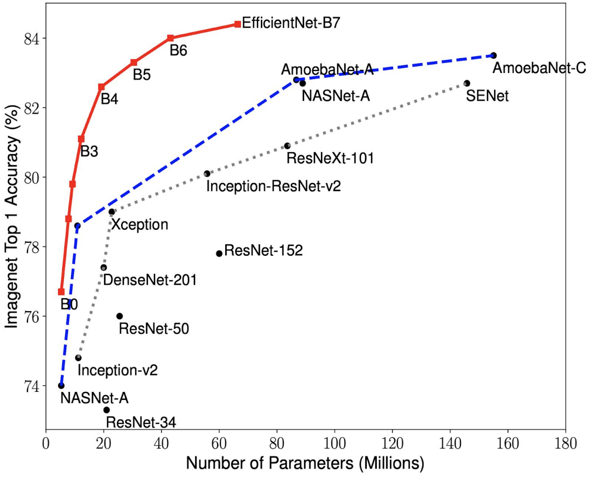
Data Augmentation

- We utilized Albumentations to obtain more data
    - Flip / Rotate / Scale / Crop
    - Guassian Noise
    - Perspective
    - Brightness



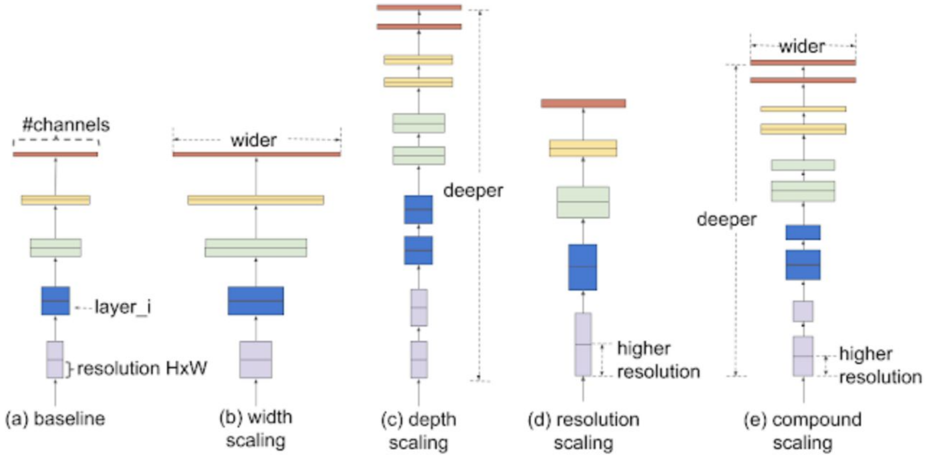
EfficientNet

- proposed by Google AI in 2019
- It uses a simple but highly effective compound coefficient to uniformly scales all dimensions of width, depth, and resolution
- B0-B7: Trade off between number of parameters and the performance



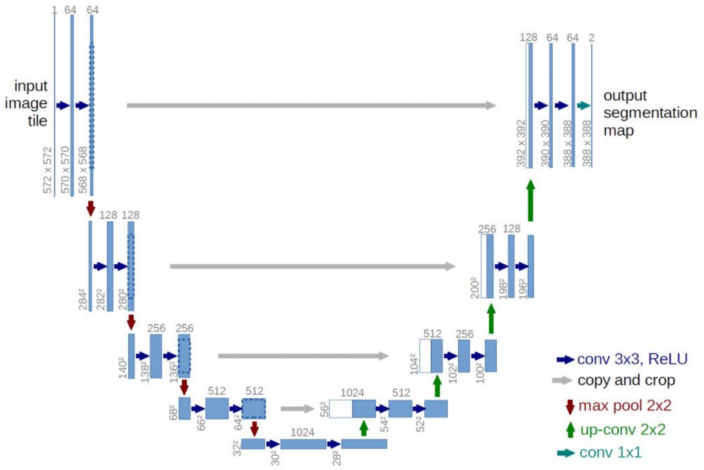
EfficientNet - Compound Scaling

- Unlike (b) - (d) that arbitrary scale a single dimension of the network, the **compound scaling** method uniformly scales up all dimensions in a principled way

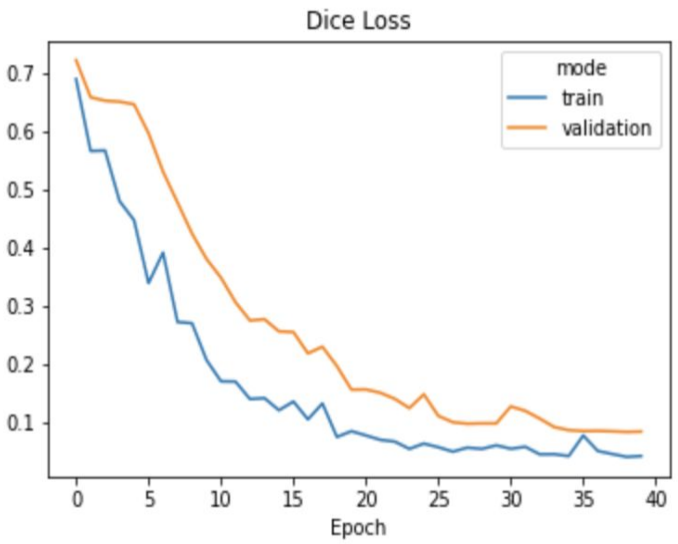


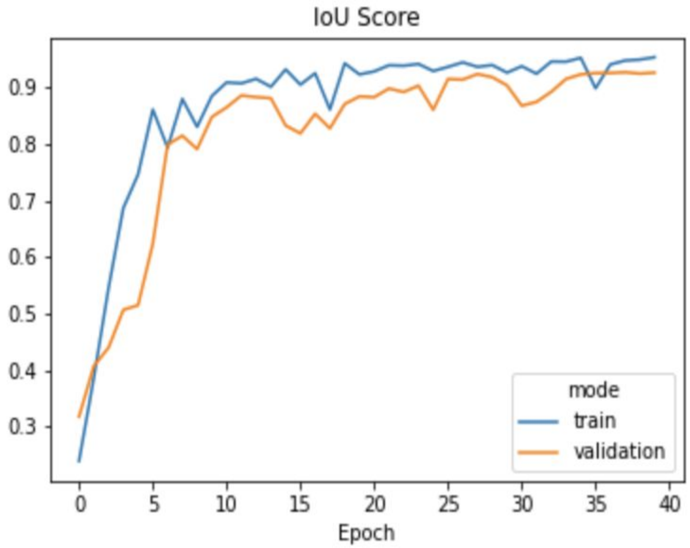
U-Net Architecture

- This architecture allows us to use a pre-trained model that has been used for a classification task - on a dataset such as ImageNet - as our encoder
- We use EfficientNet as the U-Net's encoder

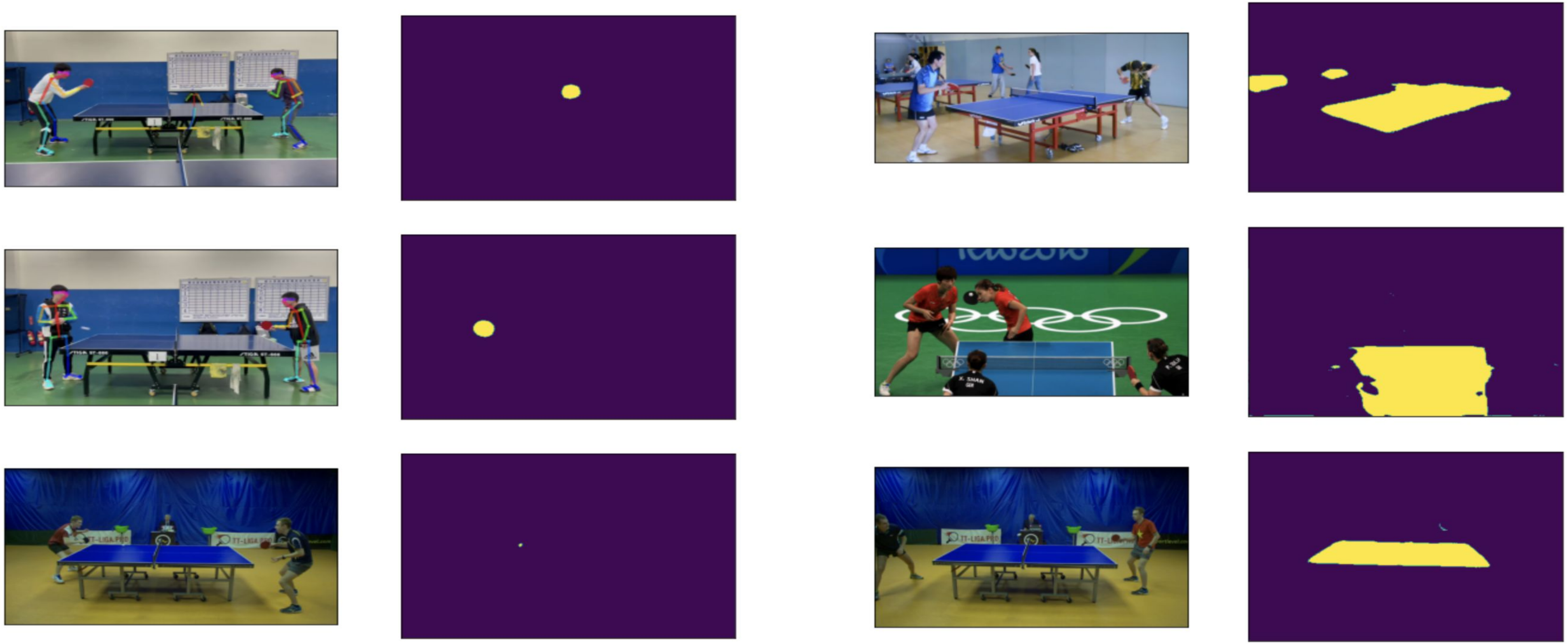


Evaluation - Training and Validation Performance





Evaluation - Segmentation Results

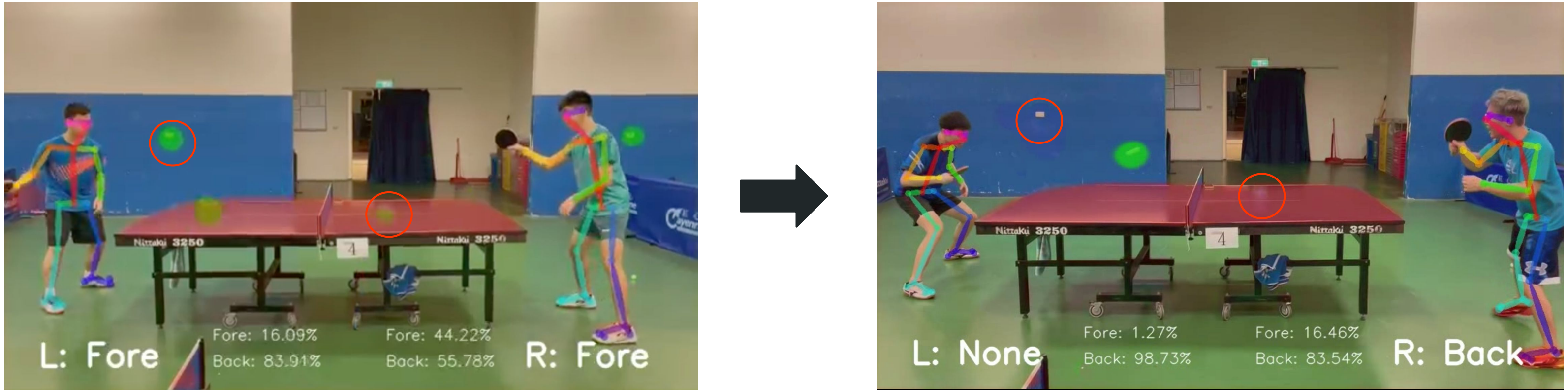


Evaluation - Analysis

- The semantic segmentation for both balls and tables performs well with **90% average IoU score**
- The areas of the segmentations of balls vary, which may be caused by the **afterimages** of balls due to the fast moving speeds
- Some of the edges of the segmentations of tables have burrs, we may need to do **edge detection** to enhance the performance

Video Optimization

- White points in backgrounds may be detected as balls
- We recover pixels that be detected as balls at 70% of all the frames in a video



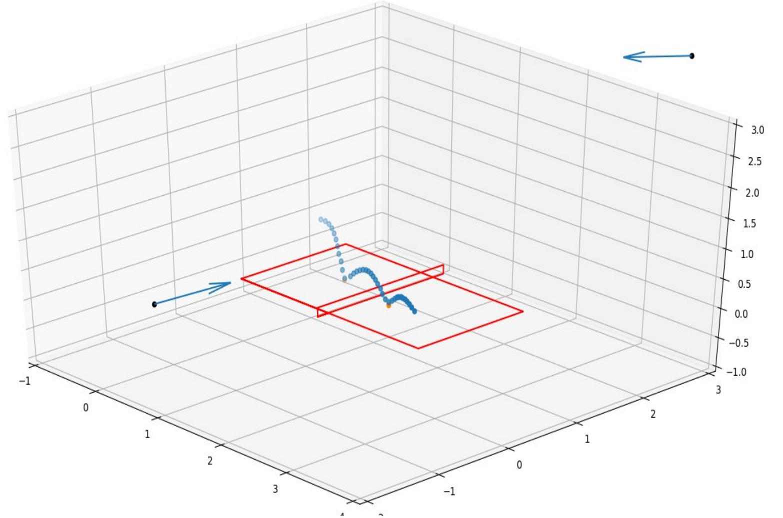

Nittaku 3250
Nittaku 3250
4
Fore: 4.48%
Back: 95.52%
L: None
Fore: 21.72%
Back: 78.28%
R: None

Conclusions

- We developed a table tennis posture analysis system only using camera images. Successfully achieved 89% accuracy on left and 95% accuracy on right
- We successfully increased the efficiency of video analyses by automatically calculating the postures distributions and automatically breaking down videos

Future Work

- 3D Ball Tracing and Location (cannot finish due to Covid-19)
- Strategy Analysis and Generation
- Automated Scoreboard System
- Real-Time Version



References

[1] R. Voeikov, N.Falaleev, R. Baikulov. TTNet: Real-time temporal and spatial video analysis of table tennis. *CVPR.* 2020.

[2] C. B. Lin, Z. Dong, W. K. Kuan, Y. F. Huang. A Framework for Fall Detection Based on OpenPose Skeleton and LSTM/GRU Models. In *Applied Science.* 2020.

[3] Z. Cao, G. Hidalgo, T. Simon, S. E. Wei, Y. Sheikh. OpenPose: Realtime Multi-Person 2D Pose Estimation Using Part Affinity Fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol.43, No.1, pp. 172-186, Jan. 1 2021.*

[4] C. Sawant. Human activity recognition with openpose and Long Short-Term Memory on real time images. *IEEE 5th International Conference for Convergence in Technology (I2CT).* 2020.