공학석사 학위논문

Design of Mobile Personal Workout Assistant Using Deep Learning

딥러닝 기법을 이용한 운동 도우미 애플리케이션 디자인

2020년 08월

서울대학교 대학원

컴퓨터공학과

박관모

Abstract

Design of Mobile Personal Workout Assistant Using Deep Learning

Gwanmo Park
Department of Computer Science and Engineering
Seoul National University

Workout exercises are a crucial component of physical fitness, and they are an integral part of many people's lives. Maintaining the correct posture during the exercise is extremely vital as incorrect postures can lead to ineffective sessions or even injuries. However, it is not easy for non-experts to check on their postures. Not only is it hard to have an objective look at their own postures, but they also lack the proper knowledge. In this paper, we design a mobile personal workout assistant, which helps users evaluate their squat postures using only simple mobile devices such as smartphone. We designed a pipeline in which the keypoint data are first extracted from the RGB videos using a pose estimation algorithm, then analyzed using a deep neural network model inspired by those used in action recognition tasks. We collected a dataset of nearly 20,000 squat exercises to train the model from scratch, and successfully created a classification model with the test accuracy of 85%, suitable to create a prototype mobile application which the users can utilize to check their postures.

A user study aimed to evaluate the effectiveness of the application is planned for future research.

Keywords: posture correction, neural network, mobile device, pose estimation, deep learning

Student Number: 2018-23020

<Table of Contents>

<Table of Tables>

<Table of Pictures>

1. Introduction

Workout exercises such as squat, push-up, and shoulder press, are a crucial component of physical fitness and health. As such, many people regularly perform these exercises, making them an integral part of their lives. To reap meaningful benefits from workouts, however, maintaining proper posture is vital. Incorrect postures can activate wrong muscles or apply unwanted pressure on the body, leading to ineffective sessions or even injuries. Unfortunately, avoiding incorrect postures is not an easy task for most non-experts. It is hard to view their own bodies from objective perspective, and performing the exercise takes too much effort that they do not have the leisure of checking their postures. Widely accepted solution for this is to have someone else watch and give feedback. However, non-experts often do not have the proper knowledge of correct postures and can not give helpful feedback. Experts, on the other hand, can give valuable feedback and increase the quality of workout sessions, but they are in short supply and often require considerable economic costs.

There has been a number of research on posture correction, but most required special devices or used heuristic methods. In this paper we aim to use deep neural networks to design a personal workout assistant capable of giving feedback on squat postures using only simple mobile devices such as smartphones. We use a combination of pose estimation and video classification to analyze the workout postures. The skeletal pose data is first extracted from RGB video using an open-source pose estimation algorithm, which is then used as the input to a classification network. We utilized network architectures for action recognition tasks, modified to fit our needs. In order to train a quality model, we created a dataset of about 20,000

squat data performed by over 100 experts and novices, annotated by experts. As the dataset contains more labels than what we used in this paper, we expect that it can be in other interesting purposes.

2. Related Work

In this section, we organize the related work in three different areas: posture correction, action recognition, and pose estimation.

2.1. Posture Correction

There has been various attempts to create a computer-aided posture correction system. For example, Chen et al. [1] built a system which analyzed workout posture using pose estimation, similar to our own. However, they used geometric methods and dynamic time warping using very small amount of data. Nike+ Kinect Training was a fitness game for the Xbox 360 that used Microsoft Kinect, a depth-sensing camera device [2]. Han et al. [3] proposed use of deep neural network to analyze the skeleton data extracted using Microsoft Kinect. Both [2] and [3] used a special device, and [3] did not actually show the implementation.

2.2. Video Classification

For the classification model architecture, we adopted ideas from those developed for video classification tasks. Specifically, we closely looked into action recognition task, the identification of different actions from video clips. It is similar to our posture classification task in that both deal with a sequential data of a single action. Out task could be considered simpler as we extract human skeleton data instead of using the original video, significantly decreasing the size of input

data. At the same time, the posture classification task is somewhat more nuanced since it tries to spot a subtle difference between the same action.

The state-of-the-art action recognition approaches could be broadly categorized into two groups: single stream network and two stream networks. LRCN[4] and C3D[5] uses single stack of layers to learn spatiotemporal information. TwoStreamFusion[6], TSN[7], ActionVlad[8], HiddenTwoStream[9], and I3D[10] all use separate stacks for spatial and temporal streams. Of these we mostly adopted the architectures that uses single stream network, because having separate stacks for spatial and temporal streams loses its meaning for us as we use the extracted keypoint data instead of images. Specifically, we implemented a model that uses both LSTM blocks (analogous to [4]) and temporal convolution layers (analogous to [5]). More details on them will be given in section 4.

2.3. Pose Estimation

The field of pose estimation has seen a rapid growth in the past decade. Early methods were based on depth images [11]. Nerual networks were first used in 2014 by Toshev et al. [12], and Cao et al. proposed a method for realtime multi-human 2D pose estimation in 2017 [13]. The method was further refined by Moon et al. by identifying and fixing common errors in pose estimation systems [14]. Multi-human 3D pose estimation developed by Moon et al. in 2019 [15]. In our system, we used OpenPose[13], an open-source project capable of detecting human body, hand, facial, and foot keypoints in realtime, for 2D pose estimation process (see figure 1). We chose to use OpenPose because it was well supported and the pre-trained models are readily available.

Figure 1. Example of OpenPose output. Image from www.github.com/CMU-Computing-Lab/openpose

	Correct	Incorrect
Training Set	8,824	8,827
Test Set	1,116	1,115

Table 1. Composition of the dataset.

Figure 2. Sample data point from a single squat exercise. 3D keypoint data (rightmost) are actually recorded in CSV formats.

3. Dataset

In order to train a quality model, we collected our own dataset of squat exercise data with the help of the experts from the Sports Science Department of Pusan National University. Total of 19,885 squats performed by 105 individuals were collected. Of those, 9,940 squats by 53 experts were classified as correct postures, and the remaining 9,942 squats performed by 52 novices were classified as incorrect postures. We specifically chose to collect incorrect postures from novices instead of experts, because we wanted the incorrect posture data to be representative of naturally arising postures, not the ones forcefully created by the experts. We selected 17 arbitrary individuals (from both experts and novices) to use their squat data as the test set. Test set was split based on individuals instead of data in order to ensure that the test set is comprised of entirely new data from the individuals that were not included in the training set. We expected that the squats performed by the same individual must

share some amount of characteristics. The exact composition of training and test sets are shown in Table 1.

We collected three different data from each squat: RGB video recorded from the front, RGB video recorded from a right-side diagonal direction, and the 3D skeleton data extracted from depth images[11] recorded using Microsoft Kinect. Sample data is shown in Figure 2. In addition to being classified into correct and incorrect postures, each data point has ten supplementary labels. Four of these labels are the metadata of the squat performer: age, gender, height, and weight. Remaining six labels identify the specific parts of the body that are considered incorrect: base of support, left knee, right knee, torso, hip joint, and neck. These five body parts were identified as the common causes of incorrect squat posture. All data were labeled by the experts from the Sports Science Department of Pusan National University. All of the individuals performed squats without holding anything on their hand, because the algorithm used to extract the 3D pose data often incorrectly identified the individual's arms or hands if they were holding something.

4. Method

4.1. Pipeline

In this section we describe the pipeline of our system. Frist, the videos recorded from the front are trimmed to accelerate the pose estimation process. Then we use OpenPose to extract 2D skeleton keypoints from the videos, but we only use certain portion of the output (see Figure 3) as some parts of the body, such as arms, head, and feet, do not have significant impact on the correctness of the squat posture.

The keypoint data are normalized in the following way. A reference frame is selected for each video among the first few frames after a human is detected for the first time. We added the condition about human detection as in some videos the individuals were not standing within the camera frame at the start. The reference frame is selected from early portion to ensure the highest chance that the target is upright and standing straight in the reference frame. Once the reference frame is chosen, a transform matrix that would map the coordinate of the hip joint keypoint to (0, 0) and the length of the torso to 1 in the reference frame is found and uniformly applied to all the frames. This ensures the coordinates of the keypoints would fall within a similar bound of values for every data point, and the differences between the data points from individuals with differing heights are relieved. To mitigate the inevitable errors from the pose estimation we apply Gaussian filter with standard deviation 1 to each keypoint sequence. This has the effect of smoothing out the movement of each keypoint and suppressing any unexpected sudden jumps. Lastly, we calculated the per-frame changes in the normalized keypoint positions and use it as the input to the classification model.

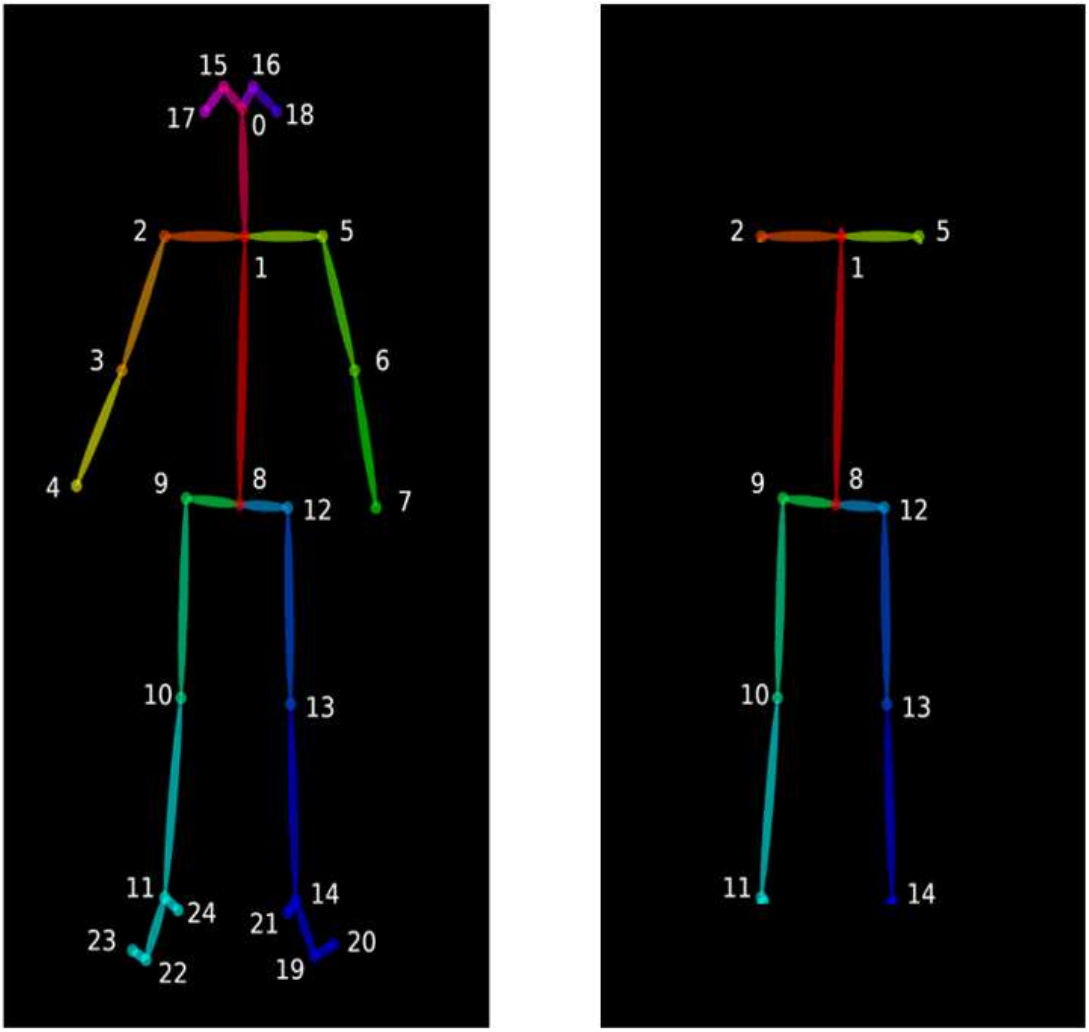

Figure 3. The full output of OpenPose (25 keypoints) on the right. The portion of the output we use (10 keypoints) on the left.

If a person was detected in k frames of the video, for example, the input tensor to the model would be of the size k-1 X 20. If the person did not move at all throughout the video, the resulting input to the classification model would be a tensor of all zeros.

The preprocessed keypoints data are then used as the input of the deep neural network which classifies them as either correct or incorrect postures.

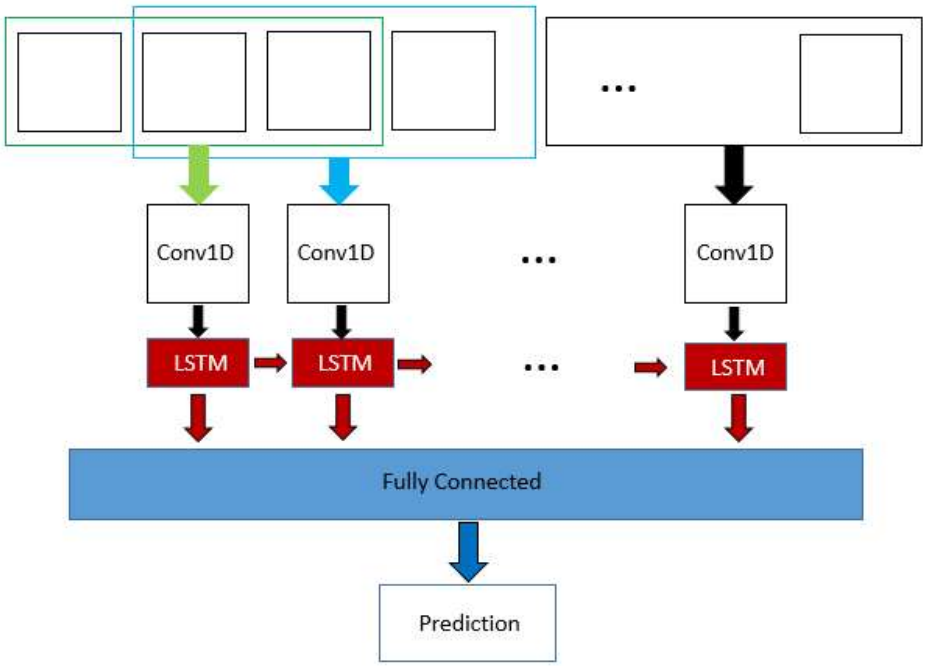

Figure 4. Simplified illustration of the model architecture. Only one convolution layer is illustrated.

4.2. Model Architecture

We viewed out task (classification of squat postures) as a narrow branch of the action recognition task, in which videos are classified according to the actions being performed by humans in them. It is better to have an understanding of the whole action instead of per-frame information, so the temporal aspect of the data is crucial for the task. As such, we borrowed some ideas regarding temporal information from previous studies. Specifically, we combined the concepts from LRCN[4] and C3D[5] to design a model with both temporal convolution layers and LSTM blocks. Our architecture consists of three 1-dimensional convolution layers, one bidirectional LSTM layer, and one fully connected layer followed by a softmax layer. The convolution layers have temporal kernel depths of 5, 3, and

3, respectively. That is, on the first convolution layer, five consecutive frames are convoluted to create new features, and on the second convolution layer, three consecutive convolution outputs are convoluted together, and so on. Single filter on the convolution layer effectively extracts one feature from the consecutive frames. The numbers of filters on the convolution layers are 32, 64, and 64, respectively. The bidirectional LSTM layer has 64 outputs while the fully connected layer has 32 outputs. The model was trained with the Adam optimizer with learning rate 0.003 and dropout rate 0.3. These parameters were found using a grid search which resulted in the highest validation accuracy of 0.9866. Figure 4 shows a simplified illustration of our model architecture. The hyperparameters, including the number of layers and the number of filters, were found using a grid search.

Training Accuracy	Validation Accuracy	Test Accuracy	Test Precision	Test Recall
0.9929	0.9866	0.8498	0.8713	0.8394

Table 2. The performance of our model.

	Training Accuracy	Validation Accuracy	Test Accuracy
LSTM + Convolution	0.9929	0.9866	0.8498
LSTM Only	0.8712	0.8532	0.7613
Convolution Only	0.9546	0.9441	0.8203
Auto-ML	-	-	0.688

Table 3. Table of classification accuracies using various methods.

5. Evaluation

We evaluated our system on the test set described in Section 3. The results are reported in Table 1. Our model achieved accuracy of 0.8498, precision of 0.8713, and recall of 0.8394. In addition to the model described in Section 4, we also trained and evaluated models 1) without the convolution layers and 2) without LSTM blocks. We did not have a suitable baseline performance as we could not find any prior work dealing with squat postures specifically. As such, we used the results of Auto-ML Video Intelligence service of Google Cloud as the baseline even though it was not designed for this particular task, but it is still capable of classifying videos. The results are reported in Table 3. The combination of LSTM blocks and temporal convolution layers had the highest performance with test accuracy of 85%. Convolution-only method performed better (82% accuracy) than LSTM only method (78% accuracy). This implied that the benefits of temporal convolution layer is larger than those of LSTM blocks. Auto-ML Video Intelligence service understandably had low accuracy of 69%. The test accuracies are significantly lower than training or validation accuracies for all three versions of our model. This could be due to the overfitting on all three occasions, or it could mean that the large number of the test set data is very different from the data in the training set.

6. Mobile Device Prototype

Lastly, we implemented a prototype workout assistant application for mobile devices. It consists of two components: the server and the mobile application client. The server receives a single squat workout video each time, analyzes the posture through the pipeline, and responds with the classification score (the output of softmax layer) along with the received video but with the extracted keypoints rendered on top (see figure 6). The client allows users to record their squat posture and send the recording over to the server to have it analyzed. When the user touches the 'start' button (see figure 5), the recording starts 3 seconds afterwards to grand the user some time to position themselves correctly. The recording ends automatically after some amount time configured by the user, and the recorded video is sent to the server. Visual and auditory cues are given to signify both the start and the end of the recording session. When the client receives a response from the server, the recorded video is shown to the user along with the extracted keypoints and whether the posture was 'good' or 'bad' (figure 8). The user can configure the classification score threshold to adjust the 'difficulty' of the workout session. The application is best used if the device is placed at a height of around 130cm (or just below chest height) and far enough from the user's workout spot to be able to capture the user's entire body. The user obviously needs to be facing the camera just like the videos used in training. The server was implemented with Python and TensorFlow, while the client was implemented for android devices using Android Studio.

Figure 5. The UI design for the mobile application. Users can configure recording time and score threshold using the dropdown menu on the upper right corner.

Figure 6. The dropdown menu.

Figure 7. The menu to change the recoding time.

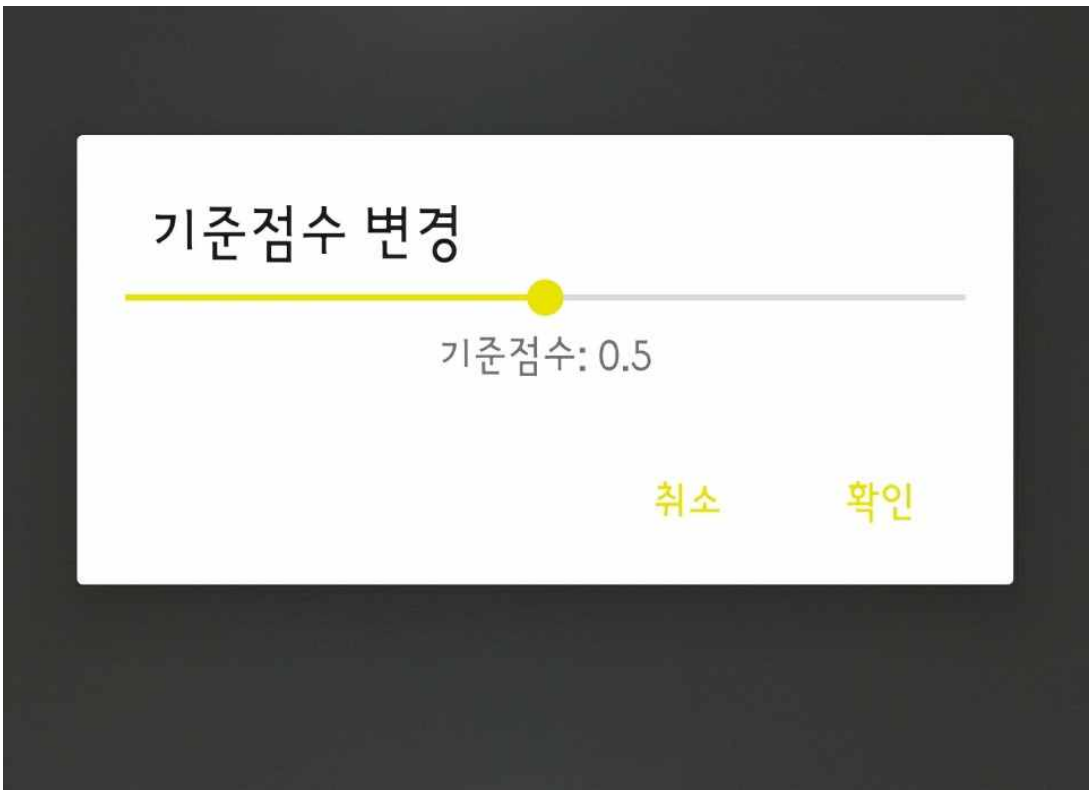

Figure 9. The menu to change the threshold score.

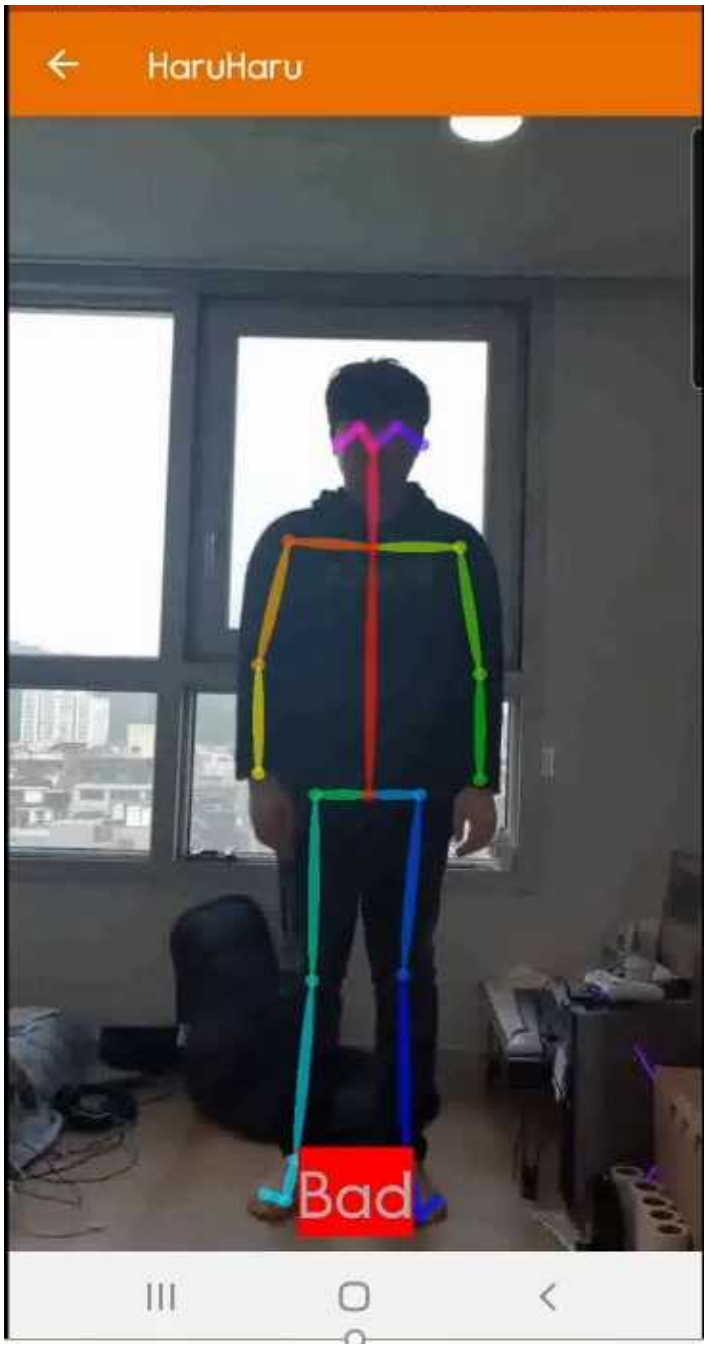

Figure 8. The result screen of the mobile application.

7. Discussion

Unfortunately, while our current system is able give feedback on the posture with some level of confidence, it suffers from few critical limitations. First, the model is not very robust. Placing the recording device too high or too low can make it hard for the system to correctly assess the posture, since the bent angles of the knees, for example, may be significantly different depending on the device's position. The model also completely fails if the user is not facing the camera for obvious reasons. This could possibly be remedied if we could accurately extract 3-dimensional keypoint data from simple RGB videos. While there were several studies on the topic, as far as we know, the state-of-the-art technique estimates 3-dimensional pose from 2-dimensional pose, and we were concerned that the subtle differences between correct the incorrect squat postures may be lost in the estimation process.

Second, the model is not able to give tips about fixing the incorrect postures. In the current form, it can only determine if the posture is correct or incorrect, but that may not be enough for beginners who lack the proper knowledge. In fact, our system cannot even tell why the posture is incorrect. We attempted a multi-label classification that not only classified the correctness of the whole posture but also the correctness of each body parts listed in Section 3. However, the results were very unreliable. For example, because majority of the incorrect postures (over 90%) were caused by incorrect knees, the model simply classified any incorrect postures as having incorrect knees. On the other hand, the base of support was so rarely incorrect that the model never classified it to be incorrect. Understanding why a deep neural network acts the way it does is a hard problem, and integrating XAI techniques to the system may help

with this issue.

Lastly, it is too hard to expand to other workout exercises. Expanding the system to pushups, for instance, would require collecting thousands of pushup data, which is extremely time and energy consuming. Data augmentation would help reducing the workload, but it may be trickier than it seems. Applying traditional image/video data augmentation techniques (such as cropping or changing color) may not yield meaningfully different 2-dimensional keypoint data. Naively augmenting the keypoint data may result in postures that do not happen naturally.

8. Conclusion

In this work we design a mobile personal workout assistant using deep neural network. We first collected a squat workout dataset with around 20,000 data points. Then we devised a pipeline with two stages: pose estimation and classification. We used OpenPose to extract 2-dimensional human body keypoint estimation, and designed a classification model inspired by models developed for the action recognition task, We evaluated the performance of our system on the test set. Lastly, using our system we implemented a mobile application capable of giving feedback on squat postures using only a simple mobile device.

References

[1] Chen, Steven and Yang, Richard. (2018). "Pose Trainer: Correcting Exercise Posture using Pose Estimation"

[2] 2012. Nike+ Kinect Training. https://www.akqa.com/work/nike/kinect/

[3] Han, Seung-Ho and Kim, Han-Gyu and Choi, Ho-Jin, "Rehabilitation posture correction using deep neural network," 2017 IEEE International Conference on Big Data and Smart Computing (BigComp), Jeju, 2017, pp. 400-402.

[4] Dnahue, Jeff and Hendricks, Lisa Anne and Rohrbach, Marcus and Venugopalan, Subhashini and Guadarrama, Sergio and Saenko, Kate and Darrell, Trevor. "Long-term Recurrent Convolutional Networks for Visual Recognition and Description". 2014. on arXiv.

[5] Tran, Du and Bourdev, Lubomir and Fergus, Rob and Torresani, Lorenzo and Paluri, Manohar. "Learning Spatiotemporal Features with 3D Convolutional Networks". 2014. on arXiv.

[6] Feichtenhofer, Christoph and Pinz, Axel and Zisserman, Andrew. "Convolutional Two-Stream Network Fusion for Video Action Recognition". 2016. on arXiv.

[7] Wang, Limin and Xiong, Yuanjun and Wang, Zhe and Qiao, Yu and Lin, Dahua and Tang, Xiaoou and Gool, Luc Van. "Temporal Segment Networks: Towards Good Practices for Deep Action Recognition". 2016. on arXiv.

[8] Girdhar, Rohit and Ramanan, Deva and Gupta, Abhinav and Sivic, Josef and Russell, Bryan. "ActionVLAD: Learning spatio-temporal aggregation for action classification". 2017. on arXiv

[9] Zhu, Yi and Lan, Zhenzhong and Newsam, Shawn and Hauptmann, Alexanger G.. "Hidden Two-Stream Convolution Networks for Action Recognition". 2017. on arXiv.

[10] Carreira, Joao and Zisserman, Andrew. "Quo Vadis, Action Recognition? A New Model and the Kinetics Dataset". 2017. on arXiv.

[11] J. Shotton et al. "Real-Time Human Pose Recognition in Parts from Single Depth Images," CVPR 2011, Providence, Rl, 2011, pp. 1297-1304.

[12] A. Toshev and C. Szegedy, "DeepPose: Human Pose Estimation via Deep Neural Networks," 2014 IEEE Conference on Computer Vision and Pattern Recognition, Columbus, OH, 2014, pp. 1653-1660.

[13] Z. Cao, T. Simon, S. Wei and Y. Sheikh, "Realtime Multi-person 2D Pose Estimation Using Part Affinity Fields," 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, 2017, pp. 1302-1310.

[14] Moon, Gyeongsik, Ju Yong Chang and Kyoung Mu Lee. "PoseFix: Model-Agnostic General Human Pose Refinement Network." 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (2019): 7765-7773.

[15] G. Moon, J. Y. Chang and K. M. Lee, "Camera Distance-Aware Top-Down Approach for 3D Multi-Person Pose Estimation From a Single RGB Image," 2019 IEEE/CVF International Conference on Computer Vision (ICCV), Seoul, Korea (South), 2019, pp. 10132-10141.

국문초록

스쿼트, 푸쉬업과 같은 운동은 건강을 유지하는 역할을 하며 많은 사람의 삶에 중요한 부분을 차지한다. 이 같은 운동을 할 시에는 올바른 자세를 유지하는 게 중요한데, 특히 근육운동의 경우 잘못된 근육을 사용하거나 몸에 필요 이상의 압박을 주어서 운동의 효과를 받지 못하거나 심지어는 부상이 생길 위험도 있다. 문제는 비전문가들의 경우 본인의 자세를 확인하기가 어려울 수 있다. 운동 중 본인의 몸을 객관적으로 보기도 어려울뿐더러 올바른 자세에 대한 지식이 없는 경우도 많다. 본 연구에서는 스마트폰과 같은 간단한 모바일 기기만으로 사용자의 스쿼트 자세에 대한 피드백을 줄 수 있는 모바일 운동 도우미 시스템을 고안한다. 본 시스템에서는 먼저 모바일 기계에서 촬영된 RGB 영상으로부터 포즈 추정 기술을 사용, 관절 위치를 계산한 뒤, 딥러닝 네트워크를 사용해 자세의 올바른 정도를 분석한다. 여기서 사용되는 딥러닝 네트워크는 기존의 행동인식 네트워크를 참고하여 디자인되었고 20,000개가량의 스쿼트 데이터로 만든 데이터셋을 사용해 학습되었다.

주요어: 자세교정, 인공 신경망, 모바일 기기, 포즈 추정, 딥러닝
학번: 2018-23020