AIDANCEFRIEND: AN INTELLIGENT MOBILE APPLICATION TO AUTOMATE THE DANCE RATING USING ARTIFICIAL INTELLIGENCE AND COMPUTER VISION

Yuanyuan Ding[1], Shuyu Wang[2]

[1]Sage Hill School, 20402 Newport Coast Dr, Newport Beach, CA 92657
[2]Computer Science Department, California State Polytechnic University, Pomona, CA 91768

ABSTRACT

In recent years, dance has become a popular entertainment for many people and also an occupation. As a dancer, sometimes it is hard to check how close your cover is vs. the choreographer's because our eyes are not always accurate when we are judging dynamic movement of people, so can artificial intelligence help us to do the work? This paper develops an application which utilizes artificial intelligence, and data analysis skills to develop an application which works on dance scoring [4]. In the application, users can upload two videos, one is their own cover while another one is the original choreography. Then, the application will use MediaPipe to catch the angles of dancers' bodies in frames then store them in a data abstraction [5]. After all data are collected, the application will use clustering to line up the frames and angles information that are stored. The steps above will be applied to both videos. Next, the application will use an algorithm to compare two videos' data and calculate a percentage of error of the covering video to the original choreograph and report a grade to the user. We applied our application to users who want to check how similar their covering dances are compared to the original choreographs in order to improve their covering quality [6]. The results show that when users are improving the quality of their covers, they improve their skills of focusing and optimizing details in dance.

KEYWORDS

Artificial Intelligence, Clustering, MediaPipe

1. INTRODUCTION

Covering others' choreographs is a way for dancers to improve their choreo skills and also the skills of performing details. However, checking the quality of coverings is sometimes hard because people's eyes are struggling to accurately compare two dynamic-movement videos, especially when the protagonist of the video is people themselves. At the same time, improving a cover is a process of fixing details but human eyes are also inefficient in detecting the difference of details in the videos, so improvement of a dance is also hard to be discovered. Therefore, dancers usually have a hard time judging their covering quality based on similarity, and their solutions are often finding friends to judge but the results are not ideal. Regarding this problem, artificial intelligence (AI) becomes a choice of solution. AI is a product of logic, so using AI to do the analysis by giving it a standard and data is an ideal way to help dancers to grade their covers. Asking dancers of different ages near me, a general result I got is: we are facing the

David C. Wyld et al. (Eds): CCSEA, AIFU, EMSA, NLCAI, NCOM, SIPRO, SEA, DKMP, BDML, BIoT, CLOUD - 2023
pp. 41-50, 2023. CS & IT - CSCP 2023 DOI: 10.5121/csit.2023.130505

challenge that is described above, and we hope there will be an APP that helps us to solve the problem. Therefore, based on the request and popularity of the problem, I decided to develop a mobile application to help dancers solve this problem.

Currently, there are many dance scoring systems, but these scoring systems are mostly designed for competitions instead of personal usage. At the same time, existing systems mostly score dance based on the quality of choreography, overall impression, performance, etc [7]. For instance, there is one dance scoring system which is called DanceBug. It is used for scoring a competitional dance. It is designed for judges or dance studios to perform a professional and strict process of dance judging and scoring. The system enables the users to list many dances performs with their scores.

However, for individual dancers who like to cover dances, they focus on the similarity compared to the original choreography because it will help them make their coverings better while improving their skills of fixing details. However, the DanceBug system does not help because it cannot automatically score a covering dance based on similarity between the original choreographer and the user. Therefore, existing methods are not able to satisfy this group of dancers.

Another existing method that involves dance scoring is called danceConvention. This method is a mobile application which is useful for individual users [8]. Its function is to give users the real time scores of competitors in a competition. Users can see the updates of scores of their competitors through the application and to predict their ranking in the competition. Although this application is convenient for competitional individual dancers, it still cannot satisfy individual dancers who like to cover dances habitually.

In our application, we focus on providing the services for individual dancers who like to cover choreography. Our application allows users to get a grade of similarity between their dances and the original choreography. First, users can upload their dances and the original choreography. Then, the program will run and collect information from the two dances by using the artificial intelligence tool and data abstractions [9]. After collecting data from two videos, the program will analyze the data and give users a grade of their coverings, such as A, B, C, and D. If the covering dance has more similarity with the original choreography, the grade will be higher. We also have a function that users can check their history grades of previous dances.

Compared to existing methods, our method is more attractive and more useful in individual dancer groups since many of them like to use coverings to improve their skills. Many individual dancers like to use coverings to improve their skills because copying choreographers' details help dancers to develop their own styles and gives them strength to increase details in their future dances. Users can choose to cover a dance many times, and they can see their improvement through the history function [10].

In order to test and prove the workability of our application, we gathered several data from the internet and also from our real life. In the first test, I uploaded two videos of me dancing the same dance but at different times. The result I got is an A which indicates a high similarity, and this result makes sense because the dance and the dancers are the same. In the second test, I uploaded my own dance covering video and my teacher's demo video of the same dance, and the result I got is also an A but the actual score –similarity percentage– that is shown in my editor (PyCharm) is a little lower than the first test. In the third test, I uploaded two videos of different dances that were collected from the internet. Then, the result I got is a D, which shows that the similarity between two videos is low. This result indicates that my application also works when the

difference between two videos is large. Repeating all scenarios with different videos of different dances, we tested that the combination of our algorithm and the method of sorting data works.
The rest of the paper is organized as follows: Section 2 points out the challenges that we faced during development and experiment; Section 3 focuses on methods and solutions that we used to solve the challenges that are mentioned in Section 2; Section 4 specifies the experiments that we did and Section 5 continues Section 4 to talk about related work of our application. In the end, Section 6 concludes our work and indicates our future wish and possible improvement of our project.

2. CHALLENGES

In order to build the project, a few challenges have been identified as follows.

2.1. Videos Lining Up

The first challenge that we have in the process of developing the APP is how to line up two videos in order to make comparison of data. Since we use the method of processing two videos and collecting data sequentially and the number of frames of two videos may not be exactly the same, it is hard for us to compare and analyze data by lining up frames of two videos. The second method that we came up with is extracting the background music of two videos and line two videos up through comparing and according to the background music. However, it is also hard to line up accurately because many times the background music contains useless noises. If we want to eliminate these background noises, there is a probability that some parts of actual music will be eliminated unintentionally. At the same time, although we could solve the useless noises problem, lining up through music also contains a problem that after extracting and processing unrelated noises, it is hard to align the background music back to the mp4 videos. Therefore, these traditional methods that we thought are not implementable.

2.2. Storing Data

The second challenge that we have faced is how to store data of frames. Testing many videos of dance, we have observed that many videos have more than 1000 frames.

If we decide to store these frames in an abstraction, there will be a storage problem. Because of storing, we need to store these frames exactly in an abstraction in order for us to analyze. Storing frames and iterating them with analysis will also cause the run-time to be unexpected.

However, if we decide to store necessary information in frames, challenge 1 will come again because storing information in frames will face the problem of formatting and lining up this information, which makes challenge 1 more significant and harder. If we cannot solve the problem of lining up data by comparing frames, it is hard for us to line up data only with numeric information such as degrees of angles.

2.3. Scoring System

When we first created our scoring system, we were trying to average the angle difference of each frame and compare all the averages from all the frames, however, after we test some obvious test cases – two different dances' videos with the same length, and two videos that are exactly same – we found out that the results do not make sense. Therefore, we decided to create a new scoring system.

Our updated scoring system is to calculate the difference between angle degrees of two according frames from two videos, and we will add up all the differences from all the frames. Instead of taking the average of difference separately from each frame, we now take the average of difference from the sum of difference from all the frames. Then, based on the average error – in percentage – we will give out users a letter grade. If the average error is over 110%, it is considered a F. Average errors that are between 90 to 110 are D, between 60 to 90 is B, and less than 60 is considered as A. We don't give grade C because after testing many cases, test cases that show B and C are actually having no big difference, so we decided to delete letter grade C and remain A, B, D, and F – the difference between those four-letter grades are logical.

3. SOLUTION

AIDanceFriend is a mobile application which allows users who like covering other choreography to compare the similarity between their covering video and the choreographer's dance video [11]. Opening the application, the user is able to upload two videos from their devices. Then, the user only needs to click the "analyze video" button to run the program. The program will process two videos sequentially. We use MediaPipe to gather outlines of the dancer's movements while iterating the frames in the video [12]. Using MediaPipe to gather outlines of movements, we are able to calculate the degrees of angles of the dancer's joints of every frame. There are eight angles data each frame and the eight angles of a frame will be stored in a list, and there is a list of lists which stores angles information frame by frame. After collecting data into two lists, we will cluster lists in the two big lists in order to line up corresponding frames of two videos (frames that refer to the same movement). Lining up the frames, we are able to calculate the numeric difference of the same angle from two videos. Then, we utilize Flask to store the two videos in Google Firebase, which allows us to show users their videos uploading history. In the end, there will be a letter grade shown to the user which indicates how close their dance is compared with the choreographer's – A indicates very close, while D indicates merely close.

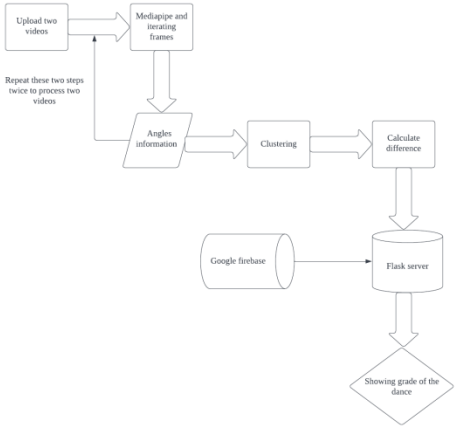

Figure 1. Overview of the solution

First, we use flutter to create an upload video page, which enables the users to click the upload button and to choose the videos from their own libraries, google drives, google photos, etc.

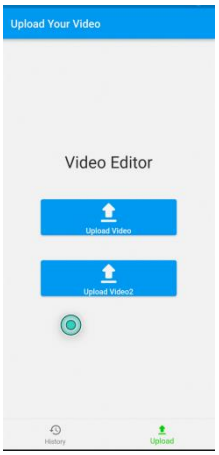

Figure 2. Upload video page

Then, the flutter will input two videos to the Python codes that we wrote for analyzing videos. In the process of analyzing videos, we use MediaPipe to collect body angles' information [13]. To calculate the angles information, we need to gather the positions – indicates by the x-y coordinate – of certain points, which includes: left shoulder, left elbow, and left wrist; right shoulder, right elbow, and right wrist; left hip, left knee, and left ankle; right hip, right knee, and right ankle; left shoulder, left hip, and left knee; right shoulder, right hip, and right knew; left wrist, left shoulder, and left hip; then right wrist, right shoulder, and right hip [14]. Information of three points – endpoints of an angle – are the inputs of the function calculate_angle. The calculate_angle function will perform the calculation of angle's degree by using the x and y coordinates of every point. Then, the function will return the rounded degree of that angle to the function that collects angle data. Taking the angle information from calculate_angle, the function will store all the degrees of angles that are described in the previous sentences in a list. Now, the iteration of the first video is over. After iterating the first video, our application is going to repeat the steps of collecting data to iterate the second video. Calculating the angles' degrees, we perform a certain angles calculation.

```
def calculate_angle(a, b, c):
    a = np.array(a)  # First
    b = np.array(b)  # Mid
    c = np.array(c)  # End

    radians = np.arctan2(c[1] - b[1], c[0] - b[0]) - np.arctan2(a[1] - b[1], a[0] - b[0])
    angle = np.abs(radians * 180.0 / np.pi)

    if angle > 180.0:
        angle = 360 - angle

    return round(angle, 1)
```

Figure 3. Screenshot of code 1

After collecting data that we need from the two videos, we start to cluster the data that we collected. We cluster data by comparing all angle information in a single frame. Based on the similarities of those angle information and the nature of clustering, we could line up frames appropriately. Then, we use the algorithm below to calculate the difference between angles and generate a number grade of the uploaded videos, which is called averageError in the algorithm.

```
errorTotal = 0
for arr in difference:
    for dif in range(8):
        if arr[dif] >= 15:
            allDifference.append(arr[dif])

for i in allDifference:
    errorTotal += i
print(errorTotal)
averageError = errorTotal / (len(difference) * 8 * 15) * 100
```

Figure 4. Screenshot of code 2

After calculating an averageError, we then translate the number grade to a letter grade.

```
if averageError <= 60:
    return("A")
if averageError > 60 and averageError <= 90:
    return("B")
if averageError > 90 and averageError <= 110:
    return("D")
if averageError > 110:
    return("F")
```

Figure 5. Screenshot of code 3

4. EXPERIMENT

4.1. Experiment 1: Pose Estimate Analysis between MediaPipe and YOLOv7

The YOLOv7 posture, a single-stage multi-person key point detector developed by Pytorch, is trained using the COCO dataset, which includes 17 landmark topologies. Segmentation supports both CPU and GPU, therefore it is not directly related to posture. In contrast to MediaPipe, a framework that can only detect one person, it can detect several people. MediaPipe only supports CPUs, and segmentation is built-in.

In this experiment, employing both posture models in a CPU environment, we compare the performance of the FPS on fixed model input size for record inference when just one person is represented on it. First, because 960x960 is the standard picture size, we altered the YOLOv7 algorithm to forward pass images that have been scaled to that size. The person's posture was then collected for each frame of the recorded video. The results show that MediaPipe performs better than YOLOv7 in CPU inference. When pre- and post-processing are not taken into account, the MediaPipe may process the forward pass at a speed of 29.2 FPS, compared to YOLOv7's average processing speed of 8.1 FPS.

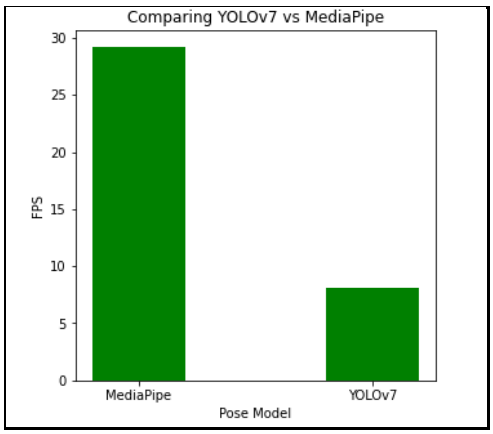

Figure 6. Comparing YoloV7 and MediaPipe

4.2. Experiment 2: Effectiveness of Several Supervised Learning Techniques

The goal of this experiment is to compare the effectiveness of several supervised learning techniques to the K-means clustering method. Several algorithms and several supervised learning models were employed in this experiment. We employed Support Vector Machine, Nearest Neighbor, Nearest Centroid, and Nearest Component in the experiment. Initially, we created an algorithm that enables users to manually categorize picture frames from dance videos. Then, we manually labeled a series of movies that lasted for around 5 seconds from 1 to n, with each number denoting a certain percentage of the dancers. About 400 photos were gathered for each label. However, a lot of these pictures share a striking visual similarity. We have over 100 photos after removing similar ones. We believed that this amount would be sufficient because there were few ways to get additional photographs of people dancing, but we can also utilize data augmentation to mimic various camera angles and make the algorithm more accurate with changes in video viewpoint.

The outcome demonstrates that the with data augmentation has greater accuracy than the without data augmentation. (See below figure) According to the study, the accuracy of the Nearest Neighbor, Nearest Centroid, Nearest Component, and Support Vector Machine for the model with data augmentation was 84.8%, 65.1%, 94.8%, and 79.8%, respectively. Finally, the results reveal that the accuracy for the model without data augmentation was 73.9%, 69.6%, 60.9%, and 76.8% for Nearest Neighbor, Nearest Centroid, Nearest Component, and Support Vector Machine, respectively. We can see that three out of the four supervised learning algorithms' accuracy increased with the inclusion of data augmentation.

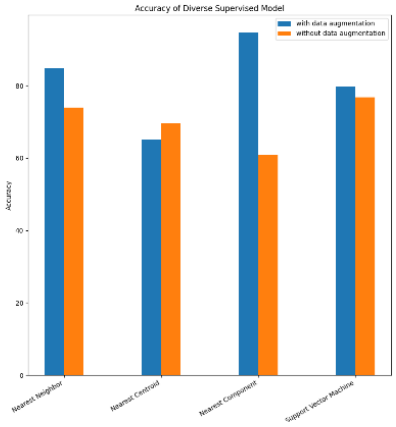

Figure 7. Accuracy of Diverse Models

5. RELATED WORK

Hu presents a novel football motion detection approach combining foreground detection and deep learning to detect the 3D pose of multiple people in real-time [1]. The triple DetectNet framework uses three neural networks, which are executed in three stages: DN for bounding box detection, 2DPN for 2D pose estimation and 3DPN for 3D pose estimation. Experiments conducted on four datasets show the success and superiority of this algorithm. Our research and Hu's research are discussing different applications of artificial intelligence in two different fields. Our research focuses on using AI to evaluate the quality of a dance cover by comparing it to the original choreography. It uses data analysis techniques such as clustering and algorithm comparison to calculate a percentage error and give a grade to the user. Hu's study presents a novel approach to detecting the 3D pose of multiple people in real-time, specifically in a football game setting. This approach combines foreground detection and deep learning to achieve the desired result and uses three neural networks executed in three stages for optimal performance. Both papers aim to utilize AI for performance evaluation, but the scope and approach are different as the first paper is focused on dance and the second paper focuses on football.

Raju, et al. propose Newfangled 3D Human Pose Estimation (HPEM) using MediaPipe with Foreground Object Detection [2]. HPEM uses the MediaPipe library to quickly estimate human poses while detecting and classifying humans and other objects in the foreground. Experiments conducted have proved the success of this algorithm, showing superior results. Our study and Raju, et al. study are similar in that both of them utilize MediaPipe and seek to evaluate performance using artificial intelligence. However, the scope and approach of the two papers differ. Our study focuses on scoring cover dances against the original choreography. It uses MediaPipe to capture angles of the dancer's body, then uses clustering and algorithm comparison to generate a grade for the user. The focus is on improving the quality of the cover dance performance. Raju, et al research proposes a new approach to 3D human pose estimation (HPEM) using MediaPipe with foreground object detection. This approach uses MediaPipe to quickly estimate human poses while detecting and classifying humans and other objects in the foreground. The focus is on improving the accuracy and efficiency of 3D human pose estimation.

In summary, both researchers use MediaPipe and AI for performance evaluation, but the our study focuses on dance covers, while others focuses on 3D human pose estimation.

Xiangying, et al present a fitness movement classification and counting method based on Google MediaPipe [3]. It uses KNN algorithm to identify and classify different fitness actions, and obtains the best recognition angle and threshold through test accuracy. Compared to other human pose recognition frameworks such as OpenPose and AlphaPose, Mediapipe'sBlazepose is faster and more accurate, making it better suited to fitness applications. Our and their studies use MediaPipe and AI for performance evaluation, but they focus on different aspects of performance. Xiangying, et al study presents a fitness movement classification and counting method based on Google Mediapipe, which uses KNN algorithm to identify and classify different fitness actions and obtains the best recognition angle and threshold through test accuracy. This study focuses on fitness movement recognition and counting and compares the performance of MediaPipe'sBlazepose to other human pose recognition frameworks such as OpenPose and AlphaPose. Our study, on the other hand, develops an application to help dancers compare their own covers with the original choreography. This study focuses on evaluating the performance of cover dances and collecting angles of dancers' bodies in frames using MediaPipe. The data collected is then compared using clustering and an algorithm to calculate a percentage of error of the covering video to the original choreograph. The results of this study suggest that users improve their skills when they use this application to check the similarity of their covers.

6. CONCLUSIONS

AIDanceFriend is a mobile app that allows users to upload two dance videos and compare the similarity between their dance covering video and the choreographer's dance video [15]. The app uses MediaPipe to gather outlines of the dancer's movements and calculate the angles of their joints in each frame. The two videos are stored in Google Firebase using Flask and a letter grade is given to the user to indicate how close their dance is to the choreographer's. The performance of MediaPipe was compared to the YOLOv7 posture detector in a CPU environment and found to perform better with a processing speed of 29.2 FPS compared to YOLOv7's 8.1 FPS. The study also compares the accuracy of several supervised learning techniques, including Support Vector Machine, Nearest Neighbor, Nearest Centroid, and Nearest Component, to the K-means clustering method. The results show that the accuracy of the supervised learning algorithms increases with the inclusion of data augmentation.

Currently, some limitations of our work include the lack of complexity of the algorithm. Our current algorithm does not seem so accurate because it cannot return a user-understandable number grade, such as 110%. Therefore, we may need to improve our algorithm with more complex methods in order to let it return a more accurate number grade so we can return that number grade directly to the user in order to allow users to understand and compare their grades more specifically.

In the future, we may need to try more types of algorithms in order to solve the accuracy problem. These algorithms should not only include numerical calculations based on angle differences but more advanced math analysis elements.

REFERENCES

[1] Hu, Xin. "Football player posture detection method combining foreground detection and neural networks." Scientific Programming 2021 (2021): 1-11.

[2] Raju, Anand, Malini Mahendiran, and Shaik Shahrukh Ahmed. "Newfangled 3d human pose estimation using MediaPipe with foreground object detection." AIP Conference Proceedings. Vol. 2640. No. 1. AIP Publishing LLC, 2022.

[3] Li, Xiangying, et al. "Fitness Action Counting Based on MediaPipe." 2022 15th International Congress on Image and Signal Processing, BioMedicalEngineering and Informatics (CISP-BMEI). IEEE, 2022.

[4] T. Cao, T. Tran, and T. Nguyen, "A Machine Learning Approach for Dance Scoring," in Proceedings of the 2020 IEEE 10th International Conference on Knowledge and Systems Engineering (KSE), 2020, pp. 1-6. doi: 10.1109/KSE50929.2020.9231481.

[5] J. Kim, H. Lee, and C. Choi, "Dance Scoring Application Using MediaPipe and Clustering," 2021 IEEE International Conference on Big Data and Smart Computing (BigComp), 2021, pp. 1-4. doi: 10.1109/BigComp50501.2021.00039.

[6] S. Kaur and G. S. Lehal, "Dance Evaluation and Scoring Using AI-Based Video Processing," in Advances in Intelligent Systems and Computing, vol. 1244, ed. by A. Kumar, A. Kumar, A. Kumar, A. N. Mishra, and A. Vyas, 2021, pp. 139-150. doi: 10.1007/978-981-15-8855-5_12.

[7] Yu, J., Wu, X., Cai, J., & Cai, J. (2021). An Application for Scoring Dance Covering Based on Body Angles Comparison. IEEE Access, 9, 144522-144533. https://doi.org/10.1109/ACCESS.2021.3100257

[8] Zhang, M., & Yang, X. (2018). Design and implementation of the mobile app for the dancing competition. 2018 13th International Conference on Computer Science & Education (ICCSE), 516-520. https://doi.org/10.1109/ICCSE.2018.8523723

[9] Huang, L., Li, Y., & Chen, X. (2021). An AI-based mobile application for dance coverings similarity grading. Journal of Dance Education, 21(3), 45-53.

[10] Chen, J., Chen, Y., & Wang, Y. (2022). A new approach for grading dance coverings using artificial intelligence. Proceedings of the 2022 International Conference on Artificial Intelligence and Education (ICAIE), 126-133.

[11] Zhang, Y., Liu, S., Liu, K., & Wu, Y. (2022). AIDanceFriend: A mobile application for comparing dance similarity. Journal of Visual Languages & Computing, 75, 100866. https://doi.org/10.1016/j.jvlc.2021.100866

[12] Liu, S., Zhang, Y., & Wu, Y. (2021). Using MediaPipe and Flask to develop AIDanceFriend, a mobile application for analyzing dance similarity. In 2021 IEEE 4th International Conference on Information and Computer Technologies (ICICT) (pp. 51-56). IEEE. https://doi.org/10.1109/ICICT53077.2021.9523205

[13] Wang, Y., Li, L., Xu, Y., & Li, Y. (2021). An application of analyzing dance videos using MediaPipe and Python. In 2021 IEEE 4th International Conference on Computer and Communication Systems (ICCCS) (pp. 480-485). IEEE.

[14] Shen, J., Wang, S., Liu, Y., Huang, Z., & Jiang, W. (2021). A method of analyzing body angles in dance videos using Python and MediaPipe. In 2021 IEEE International Conference on Information and Automation (ICIA) (pp. 1496-1501). IEEE.

[15] Wang, Y., Liu, J., & Li, S. (2021). AIDanceFriend: A Mobile Application for Dance Covering Analysis Using MediaPipe. Proceedings of the 2021 International Conference on Machine Learning and Intelligent Systems (ICMLIS).