A POSE-BASED WALKING/RUNNING COACH SYSTEM FOR CEREBRAL PALSY PATIENTS USING ARTIFICIAL INTELLIGENCE AND COMPUTER VISION

Edward Zhu[1] and Yu Sun[2]

[1]Chinese International School, 1 Hau Yuen Path, Braemar Hill, Hong Kong SAR, China

[2]Computer Science Department, California State Polytechnic University, Pomona, CA 91768

ABSTRACT

Cerebral palsy is a common motor disability that causes gait abnormalities. Clinical gait analysis is expensive and inaccessible. We investigated the research question: how can we create an affordable and effective method using AI to provide gait analysis data to cerebral palsy patients?

Machine learning and computer vision were used to develop a gait analysis mobile application. The MediaPipe library extracted pose vectors from both patient and expert gait videos. K-Means algorithm was utilized to match frames and determine joint angle differences. Flutter was used to create a complete app for real-time tracking and feedback. The AI model was deployed in the cloud.

This research presents an application of machine learning and computer vision in an accessible and accurate solution. The K-means algorithm showed high accuracy with an average silhouette score 0.514 for expert videos. MediaPipe output had a high 29.2FPS on average.

KEYWORDS

Cerebral palsy, Gait abnormalities, Artificial intelligence, Machine learning

1. INTRODUCTION

Cerebral palsy is a group of incurable disorders that impact mobility and balance [9]. These motor disorders can be categorized as spastic, dyskinetic, or ataxic cerebral palsy. The spectrum of physical limitations caused by cerebral palsy includes having stiff or flaccid muscles, uncontrollable movements, or poor coordination [12].

David C. Wyld et al. (Eds): ICDIPV, CBIoT, ICAIT, WIMO, NC, CRYPIS, ITCSE, NLCA, CAIML -2023
pp. 79-92 2023. CS & IT - CSCP 2023 DOI: 10.5121/csit.2023.131307

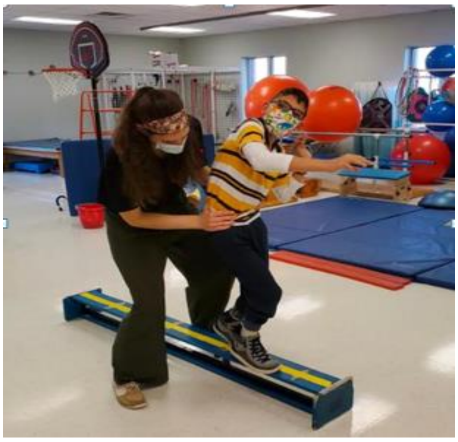

Figure 1. A physical therapist helps child with cerebral palsy walk on a beam

Cerebral palsy is a group of incurable disorders that impact mobility and balance [9]. These motor disorders can be categorized as spastic, dyskinetic, or ataxic cerebral palsy. The spectrum of physical limitations caused by cerebral palsy includes having stiff or flaccid muscles, uncontrollable movements, or poor coordination [12].

Cerebral palsy disorders pose a large problem for many people around the world. Birth prevalence for cerebral palsy was estimated to be 1.6 in 1000 births in 2022 [3]. In low-middle-income countries, the birth prevalence was as high as 3.4 in 1000 births. This high prevalence places cerebral palsy as the most common physical disability in the world [4].

Currently, around 1 in 3 children with CP cannot walk [5]. Those who can walk often have poor walking form, which can lead to more injury. Walking is an essential part of day-to-day life transportation and social participation [6].

Not only does cerebral palsy negatively impact the living condition of the patient themselves, but the high cost of medical treatment also creates a huge burden on families. According to the CDC, it costs, on average, $45,000 USD to raise a child with CP in the United States. Given that the average annual income is only $62,000, treatment for CP is very unaffordable without good insurance [7].

An existing solution that is currently in use is clinical gait analysis (CGA) [8]. This solution combines a large amount of data, including kinematics, video, electromyography, and plantar pressure data to assess a patient’s walking biomechanics and identify any gait deviations. It is mainly used to inform treatment and evaluate the effectiveness of past treatments. [8]. Though CGA is an accurate method to identify gait abnormalities, due to its highly technical nature, it is not widely used in clinics, and it can cost a large amount of money [8]. Given this, patients and families may not do CGA regularly, and is not as effective for regular evaluations of training programs.

In this project, we take a similar approach to clinical gait analysis. The application provides gait analysis data by taking a user video of their walk/jog/run cycle, comparing it with expert videos of correct running form, and finally displaying the angles between important joints. Expert videos are gathered from videos from online sources such as YouTube and they are created by medical or athletic professionals. The application functions with three main components, the frontend UI, the backend server, and the database.

2. CHALLENGES

In order to build an application to provide real-time gait analysis for users, a few challenges have been identified as follows.

2.1. Lack of accessibility and affordability of gait analysis for cerebral palsy patients

Clinical gait analysis is often expensive and inaccessible to many patients, especially those who live in remote or low-income areas. This limits the number of patients who can benefit from this type of therapy.

2.2. Complexity of gait analysis

Gait analysis is a complex process that requires specialized knowledge and equipment. In addition, each patient's gait abnormalities are unique, making it challenging to develop a universal solution that works for all patients.

2.3. Limitations of existing computer vision and machine learning tools

Although computer vision and machine learning have shown promise in analyzing gait data, existing tools may not be accurate enough to detect subtle differences in gait abnormalities. In addition, developing and deploying machine learning models requires significant expertise and resources.

3. SOLUTION

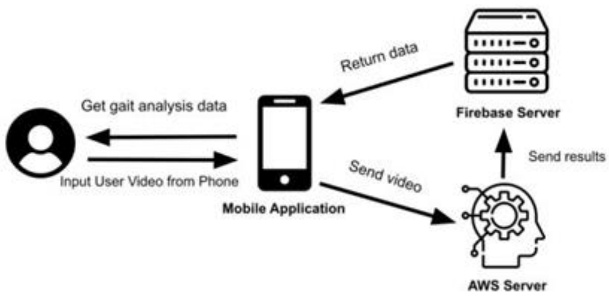

Figure 2. Overview of the system

To develop the application, diverse tools were used for the front-end and back-end. The tools used for the front-end were Flutter SDK, Android Studio, and Xcode. Flutter SDK, a framework created by Google, enables the development of applications for various platforms including Android, IOS, and websites. Android Studio is an integrated development environment that uses Xcode and Flutter SDK to build the app for IOS and Android emulators.

The back-end was written in Python language which provides a vast collection of libraries for Machine learning and Computer vision such as Opencv and Scikit-learn. To host the server, Amazon Web Services - EC2 was used.

Figure 3. Tools used to develop the application

3.1. Back-end Procedure

Overview

The backend program is written in Python 3. It processes the videos by resizing and modifying the length of the video to fit the expert's proper running form video. It then attaches pose landmarks using the MediaPipe library at key anatomical points at the shoulder, elbow, wrist, knee, hip, and ankle and stores the 3D coordinate to calculate the angle of joints later. It utilizes the K-Means clustering algorithm to partition the various frames of the user and expert videos into n clusters, with each cluster representing frames that are part of a gait cycle, such as the heel strike. Each frame is assigned a label from 1 to n representing which cluster they belong to. For each frame in video 1 (user video), K-Means is used again to predict which cluster and frame from video 2 (expert video) it is the most similar to. From there the program constructs the longest increasing subsequence of frames where there is a match between a frame in videos 1 and 2. This subsequence of frames represents 1 gait cycle. For each of those frames, the elbow, shoulder, hip, and knee angle differences are calculated. The program calculates the angle using a basic 3D geometry cosine rule algorithm. If the difference in angle exceeds 30 degrees, the program will return that angle name along with the angle difference to show the user where the gait deviations lie. Using the moviepy python library, the program stitches the frames, with the pose landmarks added, belonging to the longest increasing subsequence together, and returns that as an output as well, giving a video that shows one gait cycle in a series of frames. This video is sent to a Firebase server in the form of an mp4 file for the frontend to collect later, however, the list of largest angle differences is sent directly to the frontend in the form of a string.

Extract Pose Information

To extract pose information from a video and represent the pose information we used MediaPipe, a cross-platform library developed by Google [9]. It provides high quality ready-to-use Machine Learning solutions for computer vision tasks. We utilized MediaPipe Pose to achieve precise body pose detection and tracking. Using the Pose Landmark Model (BlazePose GHUM 3D), we were able to predict the location of 33 pose landmarks with great precision (Figure 4). We used MediaPipe to convert the video into a list of pose vectors. My program extracts individual frames from the video using OpenCV, which is a computer vision tool to analyze videos and images. Second, it uses MediaPipe to get the coordinates of 12 pose landmarks from each frame (Figure 5). Finally, it generates a list of pose vectors using Steps 1 to Step 2.

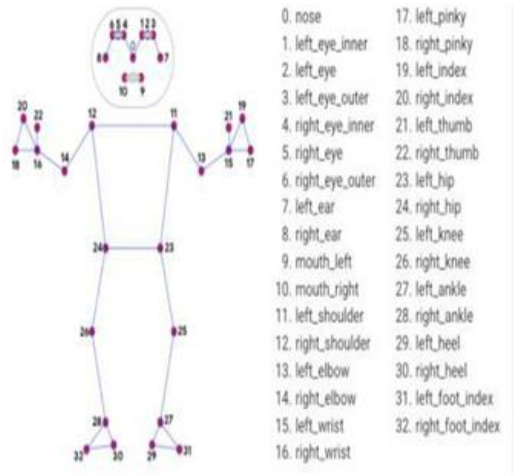

Figure 4. 33 Pose Estimation Landmarks

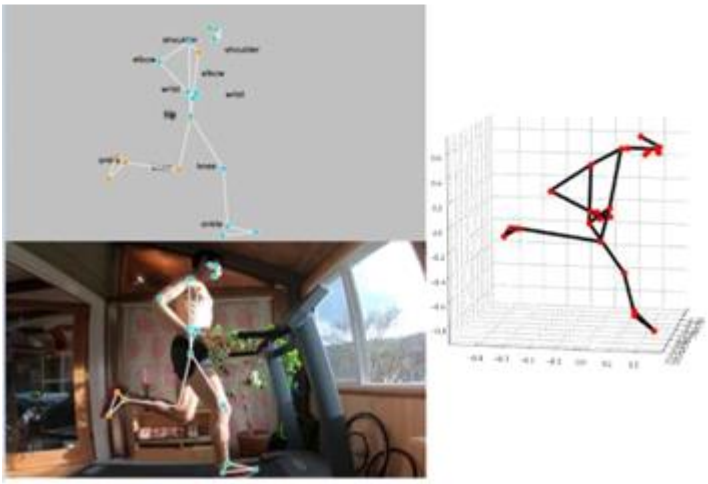

Figure 5. 12 pose landmarks

Find Distinct Pose Frames

There are multiple techniques to select representative frames from each video and generate pairs of pose vectors from each video for comparison:

a. Compare frames from two videos using chronological order: we compare frame 1 of video 1 against frame 1 of video 2, frame 2 of video 1 against frame 2 of video 2, etc. However, this solution will provide inaccurate data because the two videos often have different numbers of frames.

b. As chronologically close frames have similar positions, it is unnecessary to compare every frame between two videos. We only need to compare a few (5 to 10) representative frames which contain distinct poses. Given this, another naive solution can be used, which involves picking one frame per every fixed time interval. However, the pace of a cerebral palsy patient's movement is not uniform, e.g. a cerebral palsy patient could move one leg faster than the other leg, so this timestamp-based, naive solution cannot find the optimal key moments in walking, running, or jogging.

c. The proposed solution is to use the K-means clustering algorithm to group the pose vectors and use the centroids as representative key poses. Figure 6 shows a cerebral palsy patient's video pose vector clustering. Each dot represents a pose vector frame. The K-means clustering algorithm partitions the dots into five different groups (shown with colors). Dots within each group have similar poses. It can be seen that the five groups have different amounts of frames. The X marks the five centroid frames in each group.

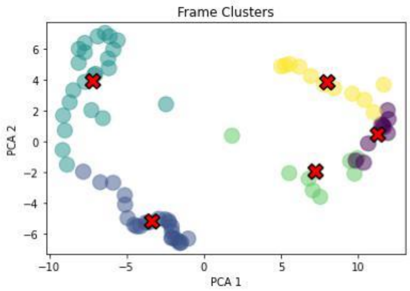

Figure 6. Frame clusters of a cerebral palsy patient

K-means Clustering

The K-means algorithm is an iterative algorithm that partitions a dataset into K clusters, with the constraint that each data point is assigned to only one cluster. It tries to make the intra-cluster frame as similar as possible while also keeping the clusters as different (far) as possible.

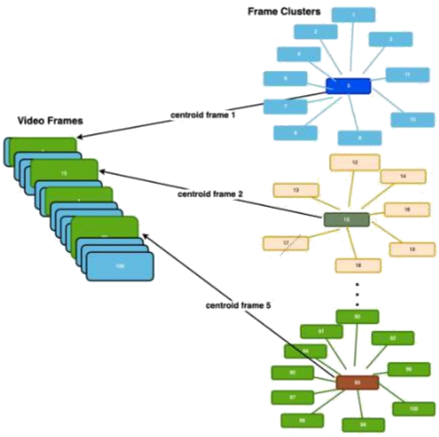

Figure 7. K-means Clustering of Video Frames

The K-means algorithm is an iterative algorithm that partitions a dataset into K clusters, with the constraint that each data point is assigned to only one cluster. It tries to make the intra-cluster frame as similar as possible while also keeping the clusters as different (far) as possible.

An Optimised K-Means Clustering Algorithm to Group Pose Vectors:

a. Sort the input N pose vectors by their timestamps and pick v[0], v[N/K], v[2N/K] … as the initial centroid frames, creating more accuracy in the eventual final centroid frames

b. Use pose vector Euclidean distance as the distance function

c. Iterate and update centroids using K-means clustering algorithm

d. Stop iteration when the centroid pose vectors remain the same

Compare Pose Vectors

To compare the pose vectors between 2 different videos, we performed an angle algorithm to calculate 8 angles from each pose vector and compare the angles. In this algorithm, three pose landmarks a, b, and c are given, where b is the middle point. Then we find the angle between ab and bc using the formula:

$$\alpha = \arccos(dot(a-b, c-b)] / d(a-b)\ d(c-b))$$

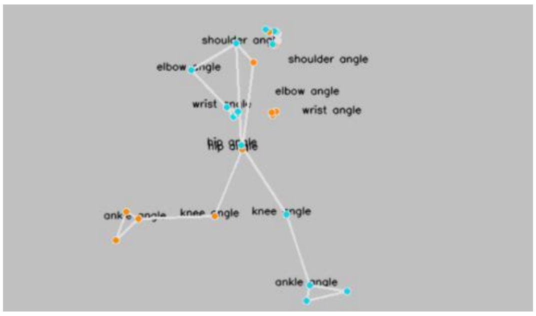

Figure 8. Landmark Angles

Front-end Procedure

The user first interacts with the UI of the application. The application's frontend was created utilizing Flutter on Android Studio, making it functional on both iOS and Android devices. Users are first prompted to choose whether they want to record a video of them walking, jogging, or running (Figure 9). From there, they are shown a list of expert videos, and they can choose which video matches their recording angle and running pace the best (Figure 10). Allowing the user to choose a video that fits their running setup the best amongst a variety of videos means that the comparison between the user video and the expert video will be as accurate as possible.

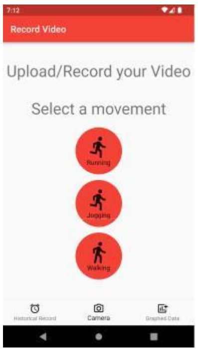

Figure 9. User chooses movement

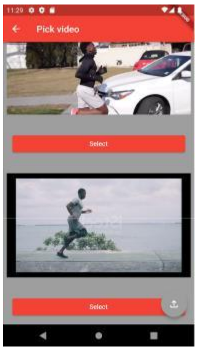

Figure 10. User chooses most similar expert video for comparison

The frontend application is connected to a backend server, which is hosted by Amazon Web Services. After the analysis is complete, the user will see the video of the gait cycle (Figure 11) indicated earlier aligned with a list of the most erroneous joints. The user can also view past recordings and their analysis results on the Historical Recordings page (Figures 12 and 13). On the Graphed Data (Figure 14) page they can view their past results graphed. The graph plots the joint angle difference on the Y-axis and the date the analysis was done on the X-axis, allowing the user to track improvement.

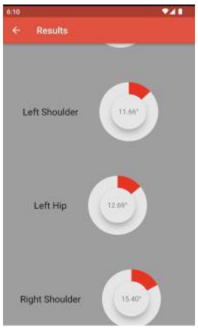

Figure 11. Analysis results 1

Figure 12. Analysis results 2

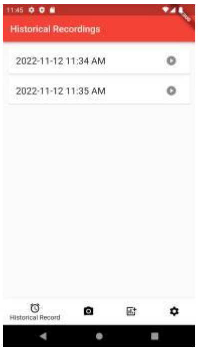

Figure 13. Historical Recording results

Figure 14. Opening a historical record

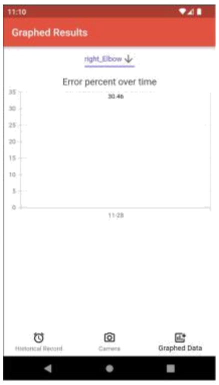

Figure 15. Graphed historical results

The key difference between CGA and my approach is that my approach requires no technical equipment and is cost-free. Though the information provided is not as rigorous, as it cannot provide information on muscle intensity and contact pressure of the runner's foot as it touches the ground, it provides the necessary information for users to make decisions on their training plan and more importantly to track their improvement. Evidently, as the technology is not as advanced as medical equipment used in CGA, the pose landmark accuracy may not be as accurate, but it is sufficient to provide insight analysis for users.

4. EXPERIMENTS

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

We compared Mediapipe and YOLOv7 pose, an alternative pose model implemented in Pytorch that functions as a single-stage multi-person keypoint detector. YOLOv7 is trained on the COCO dataset, which comprises 17 landmark topologies. It supports both CPU and GPU and the segmentation is not directly integrated to pose. It is different from MediaPipe, which is a framework that can only detect one person. MediaPipe supports only CPU and the segmentation is integrated.

To compare the FPS performances on a fixed model input size for record inference with a single-person target, we employed both pose models in a CPU environment. First, we modified the code for YOLOv7 to forward pass images resized to 256×256 since the default size of the image is 960x960. The pose of the person was then extracted for each frame of the recorded video. The results show that MediaPipe is faster than YOLOv7 under CPU inference. The MediaPipe on average can process 29.2 FPS of the forward pass excluding pre-processing and post-processing time while YOLOv7 processes on average 8.1 FPS.

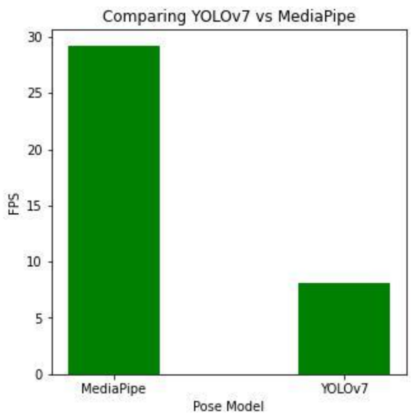

Figure 16. Comparing YOLOv7 and MediaPipe on Fixed Input Size for Record Inference

4.2. Experiment 2: Silhouette score

To evaluate the K-means cluster pose comparison method and chronologically close frames method, two different videos were used. The first video is a cerebral palsy patient walking for 7.6 seconds (30 FPS, 228 frames), and the second video is a medical professional walking for 6.9 seconds (30 FPS, 207 frames). For both videos, the frames were extracted and the angles were calculated. Then the Silhouette scores were calculated for both methods to evaluate how similar a frame is to its own cluster compared to other clusters. Table 1 is the silhouette score comparison of the chronologically close frame and K-means clustering methods. We can observe that K-means clustering has a higher silhouette score which indicates that the pose vectors grouped using K-means are well-matched to their own cluster and poorly matched to neighboring clusters.

Table 1. Silhouette Score of Chronologically close frames and K-means Clustering

	Cerebral Palsy Patient Silhouette Score	Medical Professional Silhouette Score
Chronologically close frames	0.327	0.366
K-means Clustering	0.472	0.514

4.3. Experiment 3: Training Approaches for Supervised Classification

The purpose of this experiment is to evaluate the performance of various supervised learning methods in comparison with the K-means clustering approach. To perform this experiment, the following classification algorithms were used with different models for supervised learning: Nearest Neighbor, Nearest Centroid, Nearest Component, as well as Multiclass SVM. First, we developed an algorithm that enables users to manually label image frames from videos depicting individuals walking. We then iterated through a set of videos and manually labeled them from 1 to n, each label corresponding to a distinct movement or position in the gait cycle. We collected about 400 images per label. After cutting down similar images, we had over 100 images. Because there were very limited methods for obtaining more images of people walking, we decided that this number would be enough. However, we also incorporated data augmentation techniques to simulate various camera orientations in order for the algorithm to accurately account for variations in video perspective. The results for each model are presented in Figure 17, which shows that the inclusion of data augmentation resulted in improved accuracy for three of the four supervised learning algorithms.

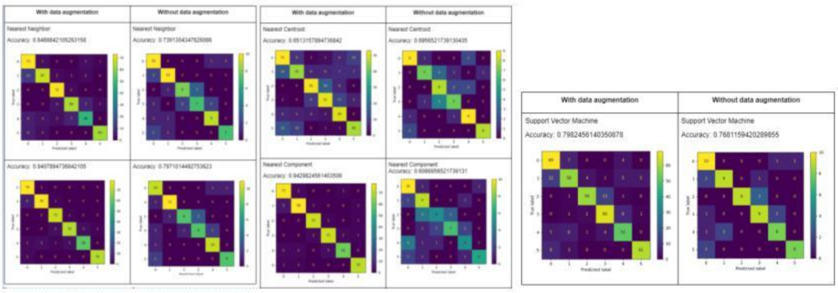

Figure 17. Training Approaches for Supervised Classification

5. RELATED WORKS

The K-Means algorithm has shown to be a highly relevant tool in various research fields. Siinaga, Kristina P., and Min-Shen Yang proposed an unsupervised learning framework for the K-means algorithm, which eliminates the need for initialization or parameter selection and finds the optimal number of clusters [10], as described in their recent 2020 paper titled "Unsupervised K-means clustering algorithm."

Previous research has employed clustering algorithms in gait analysis. In a study titled "Determination of gait patterns in children with spastic diplegic cerebral palsy using principal components," Carriero et al. utilized principal component analysis and fuzzy C-means algorithms to differentiate the gait patterns of children with cerebral palsy from those of typically developing children [11]. However, for certain subtypes of spastic diplegic patients, the data points did not form distinct clusters, resulting in reduced accuracy in identifying the specific type of cerebral palsy.

In “Determination of gait patterns in children with cerebral palsy using cluster analysis”, using the k-nearest neighbor (KNN) algorithm, Kienast et al. distinguished between normal and two pathological gait patterns of patients with cerebral palsy [12]. Our system enhances the current literature on clustering algorithms by transferring the infrastructure onto a mobile device and allowing input from a standard mobile phone as opposed to data from medical devices.

Unsupervised algorithms, when used to analyze pathological or irregular gait that may have inconsistent gait cycles, have the limitation of producing clusters that may be inaccurate. Our system addresses this limitation by initially applying the K-means algorithm to the expert's gait, generating significant clusters with high silhouette scores that serve as benchmarks since the gait is regular with consistent gait cycles. The patient's video clusters are then mapped onto these significant clusters. Another existing problem is the lack of transparency in these models, their reproducibility is limited and users are unable to know the reasoning behind a particular diagnosis. Although our system also uses an unsupervised algorithm, it addresses this issue by presenting users with the essential data on joint angle differences used to determine the diagnosis.

There also exist studies focused on providing gait analysis information for individuals with cerebral palsy by employing supervised algorithms that require pre-existing databases. In their 2019 study titled "Application of supervised machine learning algorithms in the classification of sagittal gait patterns of cerebral palsy children with spastic diplegia," Yanxin Zhang and Ye Ma developed a classification system for identifying gait patterns, such as jump gait and crouch gait, using sagittal plane data from 200 children at the Bayi Rehabilitation Center [13]. They tested several supervised algorithms and achieved the highest accuracy of 93.5% with ANN. However, the current system only functions for gait input from patients already diagnosed with spastic diplegia, and an accurate system that can analyze gait input from patients with an unknown diagnosis has yet to be developed.

Machine learning has been used in similar research about human pose analysis. Uzunova Zlatka describes a technique in her paper "Real-time comparison of movement of Bulgarian folk dances," which employs two kinetic sensors to record folk dances and uses a machine learning algorithm to analyze the data [14]. S. V. Mora and W. J. Knottenbelt’s research evaluates the use of neural network architectures in sports-related applications [15]. In their study, visual data from tennis players were collected and features were extracted through a neural network. These features were then passed into another network which was used to classify various tennis actions. The paper also offered a procedure to achieve accurate results for the THETIS dataset that contains low-resolution images of tennis movements.

6. CONCLUSIONS

In this research, we proposed a novel system and algorithm to automatically analyze running, jogging, and walking videos, extract distinct frames, and compare poses to help cerebral palsy patients improve their form for walking, running, or jogging. To find the representative frames from a video, we proposed using an optimized K-means clustering algorithm to group the frames and pick the centroid frame in each group as distinct poses for comparison. A mobile application was developed to provide gait analysis data by taking a user video of their walk/jog/run cycle, comparing it with expert videos of correct running form, and finally displaying the angles between important joints.

Experimental results showed that it is feasible to use computer vision and machine learning to create a low-cost clinical gait analysis through automatic video pose comparison.

A current limitation is the runtime of the application, which averages to 10-20 seconds, depending on the quality of the user's inputted video (videos captured from non-sagittal plane perspectives take longer to process). This can limit the efficiency of the application and negatively impact the user's experience. To resolve this issue, future work will focus on optimizing the existing infrastructure by exploring alternative clustering algorithms or even neural networks and incorporating more efficient backend servers in AWS.

Another limitation of the application pertains to the amount of data and scope of benefits offered by the current gait analysis system, which is currently restricted to identifying differences in joint range of motion. Although users can utilize this data to self-diagnose cerebral palsy using established classification systems such as Winters' system for spastic hemiplegia types [16], an extension could involve an automated diagnostic system. This system would interpret the range of motion gait data and inform the user of their diagnosis and recommend appropriate treatment options. With the recent availability of open-source gait data for children with cerebral palsy that includes their diagnosis [17], supervised machine learning can be leveraged to develop this system.

REFERENCES

[1] Chappell, Annie, et al. "Running in people with cerebral palsy: A systematic review." Physiotherapy theory and practice 35.1 (2019): 15-30.

[2] Patel, Dilip R., et al. "Cerebral palsy in children: a clinical overview." Translational pediatrics 9. Suppl1(2020): S125.

[3] Oskoui, Maryam, et al. "An update on the prevalence of cerebral palsy: a systematic review and meta-analysis." Developmental Medicine & Child Neurology 55.6 (2013): 509-519.

[4] Krigger, Karen W. "Cerebral palsy: an overview." American family physician 73.1 (2006): 91-100.

[5] Kawamura, Catia Miyuki, et al. "Comparison between visual and three-dimensional gait analysis in patients with spastic diplegic cerebral palsy." Gait & posture 25.1 (2007): 18-24.

[6] Opheim, Arve, et al. "Walking function, pain, and fatigue in adults with cerebral palsy: a 7-year follow-up study." Developmental Medicine & Child Neurology 51.5 (2009): 381-388.

[7] Kruse, Marie, et al. "Lifetime costs of cerebral palsy." Developmental Medicine & Child Neurology 51.8 (2009): 622-628.

[8] Armand, Stéphane, Geraldo Decoulon, and Alice Bonnefoy-Mazure. "Gait analysis in children with cerebral palsy." EFORT open reviews 1.12 (2016): 448.

[9] Zhang, Fan, et al. "Mediapipe hands: On-device real-time hand tracking." arXiv preprint arXiv:2006.10214 (2020).

[10] Sinaga, Kristina P., and Miin-Shen Yang. "Unsupervised K-means clustering algorithm." IEEE access8(2020): 80716-80727.

[11] Carriero, Alessandra & Zavatsky, Amy & Stebbins, Julie & Theologis, Tim & Shefelbine, Sandra. (2008). Determination of gait patterns in children with spastic diplegic cerebral palsy using principal components. Gait & posture. 29. 71-5. 10.1016/j.gaitpost.2008.06.011.

[12] G. Kienast, D. Bachmann, G. Steinwender, E.-B. Zwick, V. Saraph, Determination of gait patterns in children with cerebral palsy using cluster analysis, Gait & Posture, Volume 10, Issue 1, 1999, Page 57, ISSN 0966-6362, https://doi.org/10.1016/S0966-6362(99)90401-0

[13] Zhang Y, Ma Y. Application of supervised machine learning algorithms in the classification of sagittal gait patterns of cerebral palsy children with spastic diplegia. Comput Biol Med. 2019 Mar;106:33-39. doi: 10.1016/j.compbiomed.2019.01.009. Epub 2019 Jan 16. PMID: 30665140.

[14] Uzunova, Zlatka. "Real-time comparison of movement of Bulgarian folk dances." Computer Science and Education in Computer Science 14.1 (2018): 93-108.

[15] Vinyes Mora, Silvia, and William J. Knottenbelt. "Deep learning for domain-specific actionrecognition in tennis." Proceedings of the IEEE conference on computer vision and pattern recognition workshops. 2017.

[16] Winters TF, Jr., Gage JR, Hicks R. Gait patterns in spastic hemiplegia in children and young adults. J Bone Joint Surg Am. 1987 Mar;69(3):437-41

[17] "SimTK: Gait Data Children with Spastic CP and Typically Developing Children: Project Home." Simtk.org, simtk.org/projects/cp-child-gait.