A STUDY OF SKELETAL-BASED IMAGE PROCESSING TECHNIQUE FOR CNN-BASED IMAGE CLASSIFICATION

By

Tsega Tsahai

A thesis submitted in partial fulfillment

of the requirements for the degree of

MASTER OF SCIENCE

in

Computer Science

Middle Tennessee State University

December 2022

ACKNOWLEDGEMENTS

I would like to thank Dr. Cen Li for providing excellent guidance through writing this paper. I would also like to thank my committee members, Dr. Suk Seo and Dr. Yi Gu, for providing me with valuable support.

ABSTRACT

In the twenty-first century, significant advancements in the field of computer vision facilitated a surge in the application of image classification in different industries. This work proposes an image classification technique that utilizes a Convolutional Neural Network (CNN) to simplify training by transforming raw images into reduced representations. This proposed technique is used in developing two CNN models. The first model is applied in a human-robot interactive game of *Simon Says*. In contrast, the second is applied in a fall detection system classifying human subjects' actions as sitting, falling, or on-feet. An accuracy of 92.55% was achieved for the human-robot interactive game, while the fall detection algorithm yielded an accuracy of 90.79%. We hope this work will be a great addition to the research community as it can further be expanded to incorporate different areas of computer vision, such as human gesture recognition for autonomous vehicles.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF SYMBOLS AND ABBREVIATIONS

CNN – Convolutional Neural Network

ANN – Artificial Neural Network

RGB – Red, Green Blue

AI – Artificial Intelligence

CHAPTER I

INTRODUCTION

Since its inception, machine learning has seen substantial growth in its real-world applications within different technology realms. These realms include speech recognition, medical diagnosis, statistical arbitrage, predictive analysis, and image classification [1]. Deep learning is one of the subsets of machine learning that uses layers to structure algorithms and create neural networks [2]. Among the areas where deep learning can be applied is image classification. Image classification is defined as the ability of a computer/machine to analyze and group images in certain predefined classes [3]. This can be accomplished using artificial neural networks (ANN), the most widely used one being the convolutional neural network (CNN).

Various CNN-based systems have been utilized for image classification, such as cancer cell recognition and classification [4], blood cell image classification [5], bacteria classification [6], and human pose detection [7]. Human pose detection algorithms have been implemented in many ways and for different purposes like determining a human subject's activity, relative size, or joint locations.

This work proposes an image classification technique that utilizes CNN to simplify training by transforming raw images into reduced representations. Most CNN image classification algorithms employ raw or augmented images to perform training and classification. However, in this work, the images are transformed into skeletal binary sequences before undergoing training. This project aims to investigate whether images produced from the proposed image reduction algorithm can be harnessed to perform

image classification and yield comparable results to traditional CNN image classification models.

The proposed technique is utilized to create a CNN model used for a human-robot interactive game. Nowadays, AI applications involving robots are rapidly increasing their involvement in the day-to-day lives of human beings. These robots communicate with human beings to aid us in tasks such as delivering items, assembling products, picking orders in warehouses, and cleaning our houses without needing our presence. In these applications, gesture recognition is vital as these robots must assess their environment before and during their tasks. In this work, we experiment with CNN gesture/pose recognition by implementing the proposed image reduction technique and applying it to a human-robot interactive game. Aside from CNN image classification, this game utilizes a humanoid robot to develop an interactive game between a human and a robot. The robot used in this system was the NAO robot which was equipped with sensors supporting sound, vision, speech, and movements. This interactive game was based on a popular children's game called *Simon Says*. In this game, the robot has the role of Simon and instructs the human player(s) to make pose(s) and determines whether the postures are correct by snapping a picture of the player(s) and utilizing image classification to determine which pose the player(s) exhibited.

Similarly, the proposed image classification technique is used to create a human fall detection algorithm. Due to the increasing levels of life expectancy and decreasing levels of fertility, the population of senior citizens in the world is on the rise. Simultaneously, the number of caregivers for the elderly is stagnant [8]. According to the World Health Organization [9], accidental falling is one of the most common reasons for

declining elderly citizens' health. According to [10], up to 75% of senior citizens in nursing homes fall annually. Due to the increased substantial risk factors associated with falling, the need for surveillance systems is indisputable. Therefore, a fall detection algorithm that utilizes the proposed image classification technique, along with a CNN model, is presented. Instead of relying on a humanoid robot as a visual data source, this portion uses a regular RGB (Red, Green, Blue) camera. With the utilization of CNN image classification, this system categorizes the actions of a human subject as *Sitting*, *On-Feet*, or *Falling*.

The remainder of this paper is structured as follows: Section two presents the background; section three presents the methodologies used to achieve desired results; section four presents the results of the methods and the discussion of the results; finally, section five presents the conclusion of the paper.

CHAPTER II

BACKGROUND

Computer Vision

Computer vision is a field of artificial intelligence that allows computers and systems to gather meaningful information from visual data such as images and videos and use that information to take next steps [11]. Computer vision is applied in industries such as retail, healthcare, security, video gaming, and automotive.

Object/Pose Detection using CNN

Object detection is a subset of computer vision defined as the collection of tasks leading to the identification of multiple objects in an image. These objects can be humans, animals, or inanimate objects. Using CNN, object detection can be implemented to calculate relative distances between objects and count the number of objects in an image. In [12], CNN object detection is utilized to calculate distances between human subjects. This tool was developed to halt the spread of the Coronavirus by detecting if the distance between two people adheres to social distancing guidelines.

This work uses a pretrained two-branch CNN model [13, 14] for pose detection. This two-branch pretrained model takes a color image as an input, and the first branch establishes a set of two-dimensional confidence maps. Meanwhile, the second branch establishes a set of two-dimensional part affinity maps. Greedy inference is then used to parse the confidence maps and the part affinity maps resulting in two-dimensional key points representing the joint locations of the human subject in the image. Using the Common Objects in Context (COCO) dataset as a training set, consisting of more than

300K labeled images [15], this model produces 18 key points, including the nose, neck, right/left elbows, and right/left shoulders.

Image Classification using CNN

One of the subsets of computer vision is image classification, which is the ability of a computer to analyze and group images in certain predefined classes [3]. This can be accomplished by utilizing neural networks. One of the most widely used neural network models for image classification is the convolutional neural network (CNN).

CNN is beneficial as it reduces the number of parameters required to set up a model, and features are spatially independent [16]. CNN has three main types of layers: convolution layers, pooling layers, and fully connected layers. The convolution layer applies a convolution operation to the input image and passes the result to the following layer [17]. This convolution operation takes a kernel/filter and passes it over the input image to transform it based on the filter values. The subsequent feature maps are computed with the formula presented below, where the kernel is denoted by *h*, the input image is represented by *f*, and the rows and columns indexes of the resulting matrix are denoted by *m* and *n* [18]:

$$G[m,n] = (f * h)[m,n] = \sum_j \sum_k h[j,k] f[m-j, n-k] \qquad (1)$$

The pooling layer follows the convolution layer and conducts down-sampling to decrease the spatial size of the input further. A prevalent pooling method is Max Pooling, which down-samples the input by taking the maximum value over the window size specified. Following that, the fully connected (Dense) layer assembles class scores to be utilized for classification. The rectified linear unit (ReLU) activation function is applied to feature maps produced from the previous layers to ensure the feature maps' contents

are zero and above. That is accomplished by the formula presented below. If this function receives a negative input, it will return 0. However, if a positive input is received, it will return that value.

$$f(x) = \max(0, x) \quad (2)$$

Sigmoid is another activation function common in CNN architectures where the output is the prediction probability as it exists between 0 and 1. Sigmoid is defined by the formula below, where Euler's number is denoted by *e*:

$$S(x) = \frac{1}{1+e^{-x}} \quad (3)$$

A frequent CNN architecture for image classification is structured where pooling layers follow the convolutional layers in a redundant fashion before feeding forward to the fully connected layers [19]. Figure 1 shows the structure of a typical CNN model. This model is used to classify an input image as *duck* or *not duck*. In this structure, two convolution layers with the ReLU activation function are used to apply a convolution operation on the input image. The max pooling layer is then used to down-sample the output of the convolution layers while the fully connected layers perform the classification of the input image.

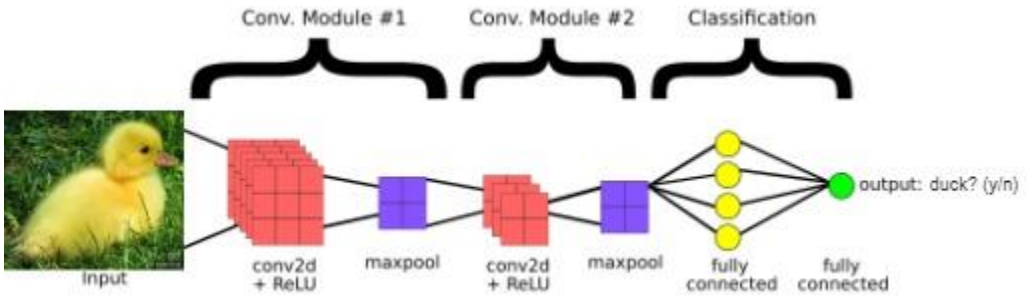

Figure 1. Common CNN structure with convolution layers, pooling layers, and fully connected layers

CNN image classification can be applied in many areas. In [20], it is involved in the classification of 102 flower species by using the DenseNet121 deep learning model. This is important as it helps in the management and protection of biodiversity. It is also applied in healthcare as it has been used to recognize deadly viruses, such as the Coronavirus, in computed tomography scans of lungs [21]. Also, CNN is useful in systems involving robots with the ability to perform tasks such as manufacturing, transportation, providing assistance, and interacting with human beings. In [22], a prototype of a socially assistive robot was created to interact verbally and recognize several objects, such as smartphones, bottles, clocks, and faces, using CNN. This robot is helpful as it can assist children in learning to interact socially by talking and recognizing objects.

Applications involving CNN image classification and object detection can help track the well-being of elderly citizens by monitoring several aspects of an individual, including the identification of falling. In [23], a fall detection system is developed by extracting a human silhouette from an image using mask R-CNN (mask regional CNN) and recognizing it as sitting, bending, standing, or lying. Similarly, in [24], a real-time skeleton-based fall detection algorithm is proposed where a human subject’s skeleton sequence is encoded onto an RGB image while maintaining spatial structure and time-dynamic information. In this paper, CNN encodes the images' reduced skeletal representations and classifies them into different classes using the Keras API.

CHAPTER III

METHODOLOGY

This methodology discusses an image classification algorithm applied to two different pose recognition processes. The first application is a human-robot interactive game. This algorithm is then expanded to be applicable in the medical field as it is utilized in a fall detection system. Figure 1 outlines the model framework of these two applications.

Figure 2. The model framework for the human-robot interactive game and the fall detection algorithm

Image Processing

Data Collection

The data/images utilized for the human-robot interactive game were collected from team members who modeled these poses. While capturing these pictures, each team member did a slight variation of a particular pose by standing at different locations to mimic players standing at various locations during the game. In total, 429 labeled images belonging to eight pose classes were collected. Table 1 lists the eight poses along with their descriptions and counts.

Table 1. The eight different pose classes for the human-robot interactive game, along with their description and count

Pose Name	Description	Count
Pose1	Player(s) put both arms out by their sides	38
Pose2	Player(s) put both hands on their waist	63
Pose3	Player(s) put both hands on their head	58
Pose4	Player(s) raise both arms like a football goal post	72
Pose5	Player(s) raise right arm straight out to their side	72
Pose6	Player(s) raise left arm straight out to their side	67
Pose7	Player(s) lie on the ground on their side	10
Pose8	Player(s) raise right arm above head	49

The dataset used to train and test the fall detection algorithm is the *Multiple Cameras Fall Dataset* [25]. This dataset contains videos consisting of 24 scenarios recorded with eight video cameras. For all the scenarios, the possible positions or movements are sitting, lying on sofa, moving up, falling, walking/standing up, etc. Because still images were needed to train our fall detection algorithm, these videos were converted into images using Python's *OpenCV* library.

After the videos were converted into images, they were manually labeled into three categories. Table 2 shows the three categories, along with their description and count. Some images were flipped horizontally to achieve a balanced dataset and address the lack of images in the sitting category. Finally, we had a total of 564 training images.

Table 2. The three different pose classes for the fall detection algorithm, along with their description and count

Action Name	Description	Count
Fall	Human subject is lying on the ground, horizontally	187
Sit	Human subject is sitting	187
On-Feet	Human subject is on their feet, either walking or standing	190

Image Processing Approach for Data Preparation

The next step in our approach is to prepare the labeled images to achieve suitability for training. Gupta's multi-person pose estimation package, *OpenPose* [14], was adopted to process these images to make them best suited for training. Using the COCO dataset and the *OpenPose* package, we obtained a collection of 18-pixel coordinate points representing important skeletal points of the body. Some of these essential skeletal points of the body are the nose, neck, right/left elbows, right/left shoulders, and so on. This model takes a color image as input and returns an array consisting of matrices with the confidence maps of key points and part affinity heatmaps of each joint pair. These joint locations are computed as actual pixel coordinates of the image.

Following the creation of the key points, our method creates a binary image of line segments by connecting the pairwise coordinate points generated previously. Using OpenCV, a blank image was created, and each key point coordinate generated was located and drawn in this image. Line segments were then formed between valid joint pairs to create a binary skeletal representation of the image. We then compute the height

and width of the stick figure-like image and crop it to occupy the entire image (no extra blank space). Figure 2 illustrates this image processing technique for one of the poses associated with the human-robot interactive game. Additionally, Figure 3 represents the image processing technique for the three classes related to the fall detection system.

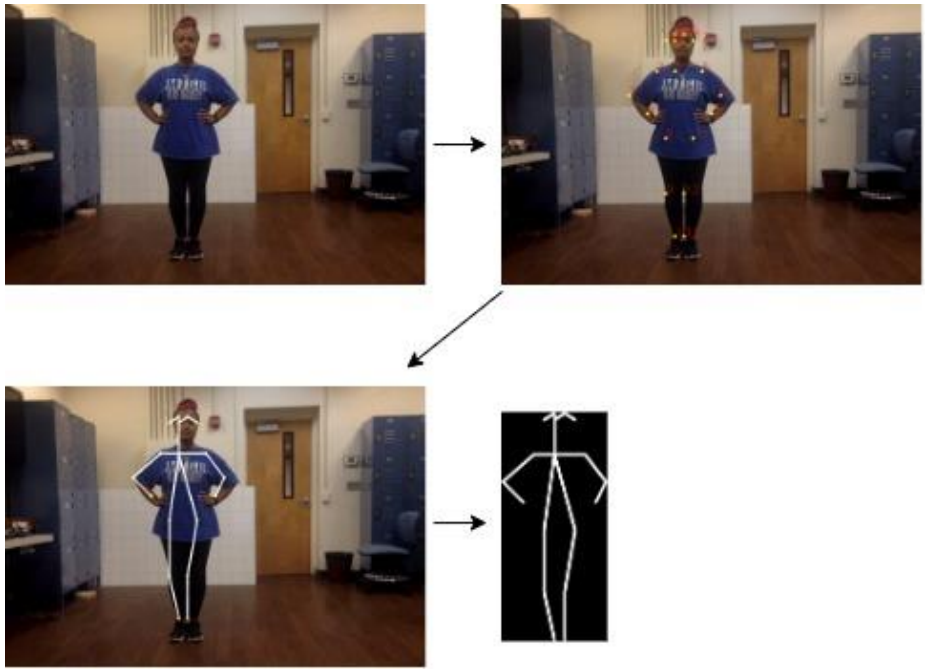

Figure 3. Image processing steps to create suitable images for training

Figure 4. Image processing steps representing three different classes

From Two-Player to Multi-Player Game

Initially, the human-robot interactive game allowed for one player at a time. That was upgraded to a two-player game, allowing the two players to play side-by-side. The two-player game was made possible by cropping the raw image into two halves and processing each half to generate each player's joint location. However, this method proved inefficient as situations where the players were overlapping or standing too close produced incorrect coordinates. An alternate image cropping approach was developed to

solve this issue and upgrade the algorithm to handle multiple players efficiently. In this new approach, instead of dividing the raw image into halves, the picture was cropped based on each player's location. The steps of this approach are as follows:

1. Use Gupta's *OpenPose*[14] to identify the joint locations of multiple players
2. Use the image processing method mentioned in the previous subsection to obtain the binary skeletal images of the players
3. In the binary image, locate the players' overall maximum and minimum height coordinates. This was accomplished using the joint coordinates of the ankles and the eyes. This step was added to achieve the most efficient image extraction, as it will give us the ability to dispose of the blank spaces found at the top and bottom of the image
4. Locate the minimum and maximum joint location coordinates of each player. The minimum joint location of a player is the joint closest to the left edge of the image, while the maximum location is the furthest from the image's left edge. For example, in Figure 4, the minimum and maximum joint locations for Player 1 are the coordinates located on the left wrist and right elbow. Similarly, for Player 2, the minimum and maximum joint locations are the coordinates located on the left and right wrists.
5. By utilizing the minimum/maximum height coordinates and the minimum/maximum joint locations, extract each player from the binary image by cropping (See Figure 4).

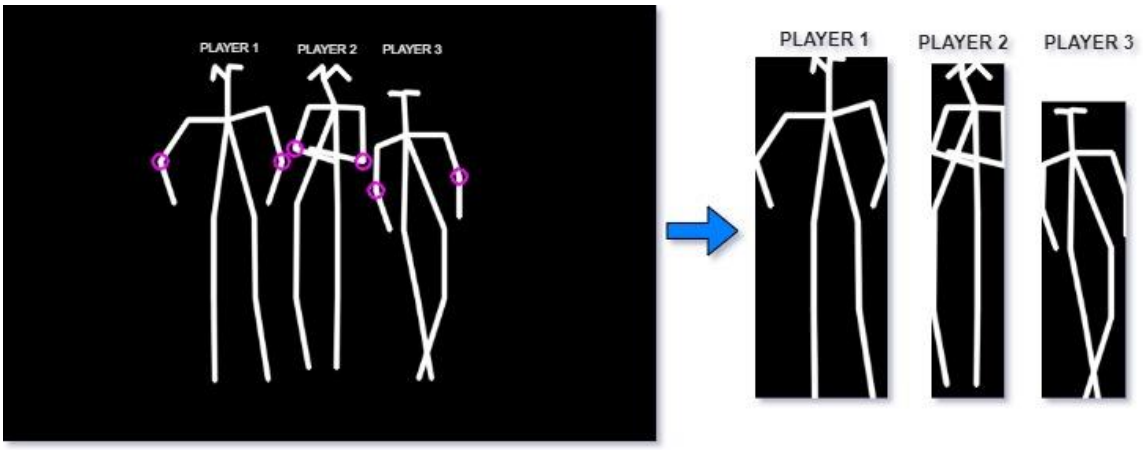

Figure 5. Image processing steps to support multi-player functionality

After these steps, each cropped image is fed into the CNN classification model and classified into one of the eight poses. By using this approach, tracking each player during the game is simplified. However, if the players were to switch places during the course of the game, this method does not have the ability to differentiate the players.

Pose Recognition

The Convolutional Neural Network Model

Besides some hyperparameter differences, the CNN models used for the robot-human interactive game and the fall detection algorithm have identical structures. Based on the skeletal representations produced from our image processing technique, CNN is utilized to classify these images into different classes. These binary images were resized to 150x150 pixels to be input into the CNN model. This CNN model (see Figure 4) has three convolutional layers, each followed by a max pooling layer. The convolution layers consist of kernel sizes of 3x3, while the max pooling layers consist of pool sizes of 2x2.

A flattening layer then converts the output into a one-dimensional array to be input to the next layer. A fully connected layer follows that with the ReLU activation function. We then add a dropout layer to regularize the model and avoid overfitting. Finally, this CNN model has another fully-connected layer with the Sigmoid activation function that performs the classification of the classes. The categorical cross entropy loss function and the Adam optimizer were used in the compilation of this model.

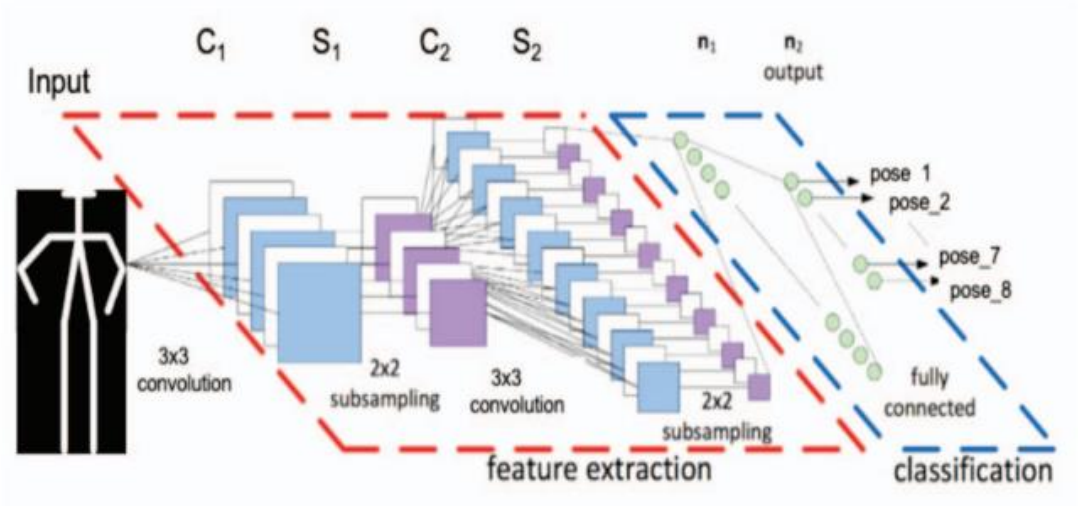

Figure 6. CNN architecture used to perform image classification

CHAPTER IV

RESULTS AND DISCUSSION

Hyperparameter Tuning

The proposed model was trained and tested on *Google Colab* with the *NVIDIA-SMI* GPU. It was evaluated using k-fold cross-validation with *k=10*. To ensure the most optimal hyperparameters were being utilized, different hyperparameter values, such as batch size, epoch number, learning rate, optimizer function, and dropout rate, were experimented with. Figure 6 shows some experimented hyperparameters for the fall detection and human-robot interactive game algorithm. Table 3 shows the optimal hyperparameters obtained from our experiment and thus used in the CNN model.

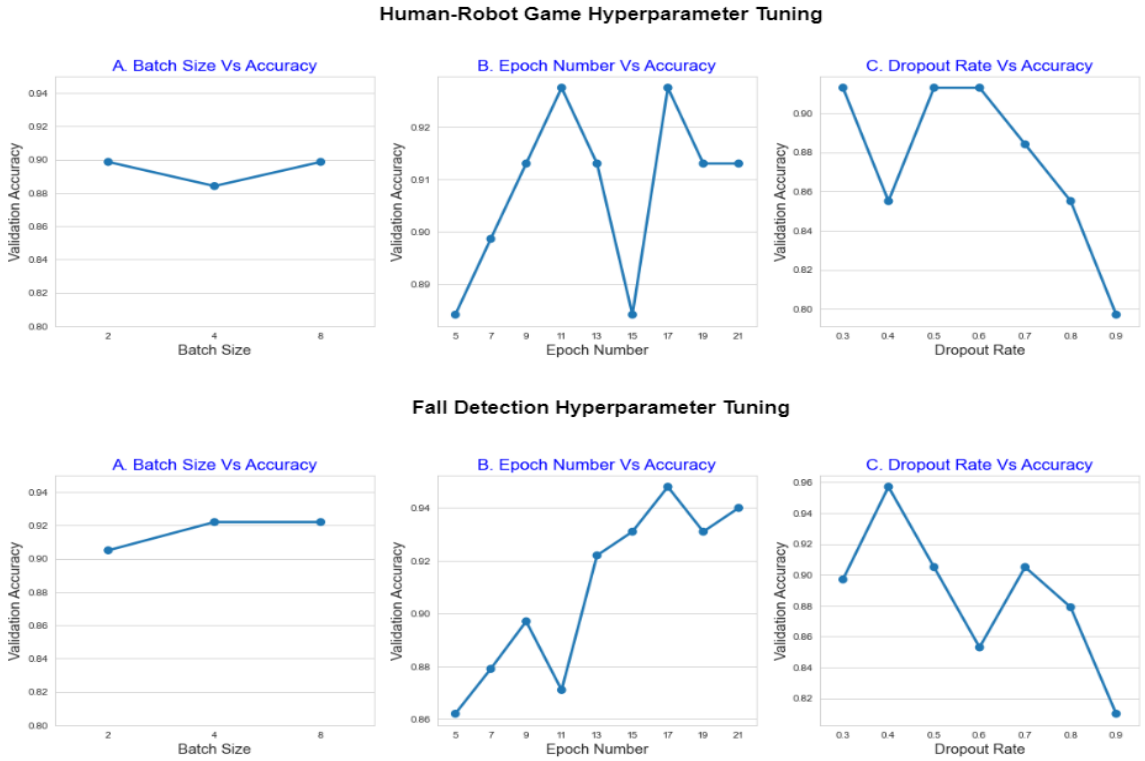

Figure 7. Results of hyperparameter (batch size, epoch number, and dropout rate) tuning for the human-robot game model and the fall detection model

Table 3. Optimal hyperparameters selected for the human-robot game model and the fall detection model

Hyperparameter	Human-Robot Interactive Game	Fall Detection Algorithm
Epoch Number	11	15
Batch Size	8	4
Dropout Rate	0.5	0.4

Human-Robot Interactive Game

After implementing the abovementioned methodologies and selecting the optimal hyperparameters, the CNN model associated with the human-robot interactive game was evaluated using a 10-fold cross-validation. This evaluation provided an average validation accuracy of 92.55% (See Figure 7).

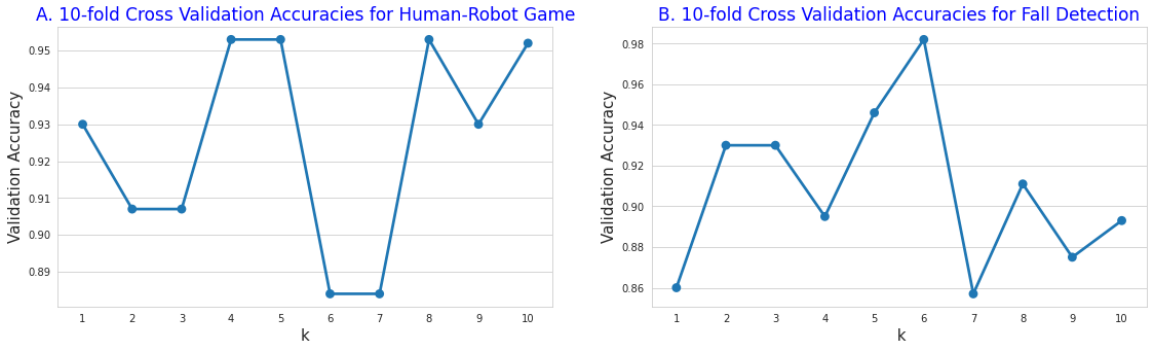

Figure 8. 10-fold cross-validation results for the human-robot game and the fall detection algorithm

Fall Detection Algorithm

The CNN model associated with the fall detection algorithm underwent evaluation using 10-fold cross-validation and achieved an average validation accuracy of 90.79% (See Figure 7). The same CNN fall detection model was run with raw unprocessed images as input. These raw images contained identical class distribution as the processed images. This model performed better than the model with the processed images and achieved an average 10-fold cross-validation accuracy of 94.66%. This model performed better as there is more image information in the raw images than in the processed binary images. However, this model's training and validation time was more than four times that of the processed binary images model. The faster rate at which the processed binary image model trains is beneficial in scenarios where adaptive learning is implemented.

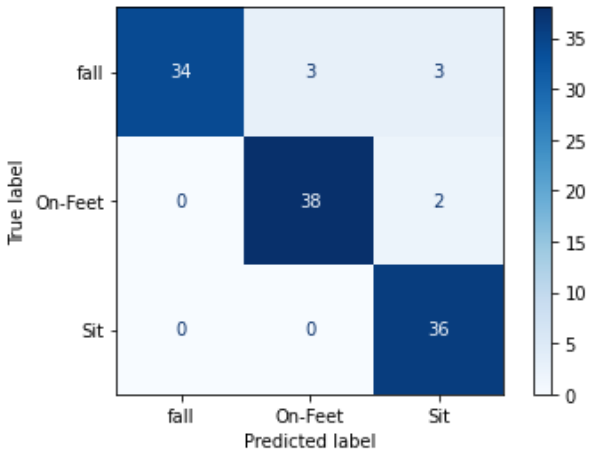

Figure 9. Confusion matrix for the fall detection model

As can be seen from the confusion matrix in Figure 8, our fall detection CNN model exhibits difficulty classifying the fall-labeled images, as 15% (6 out of 40) of these

images are misclassified. Out of the 15% of the misclassified images, 50% are misclassified as *Sitting*, while the other half are misclassified as *On-Feet*.

Further Investigation on Misclassification of Images

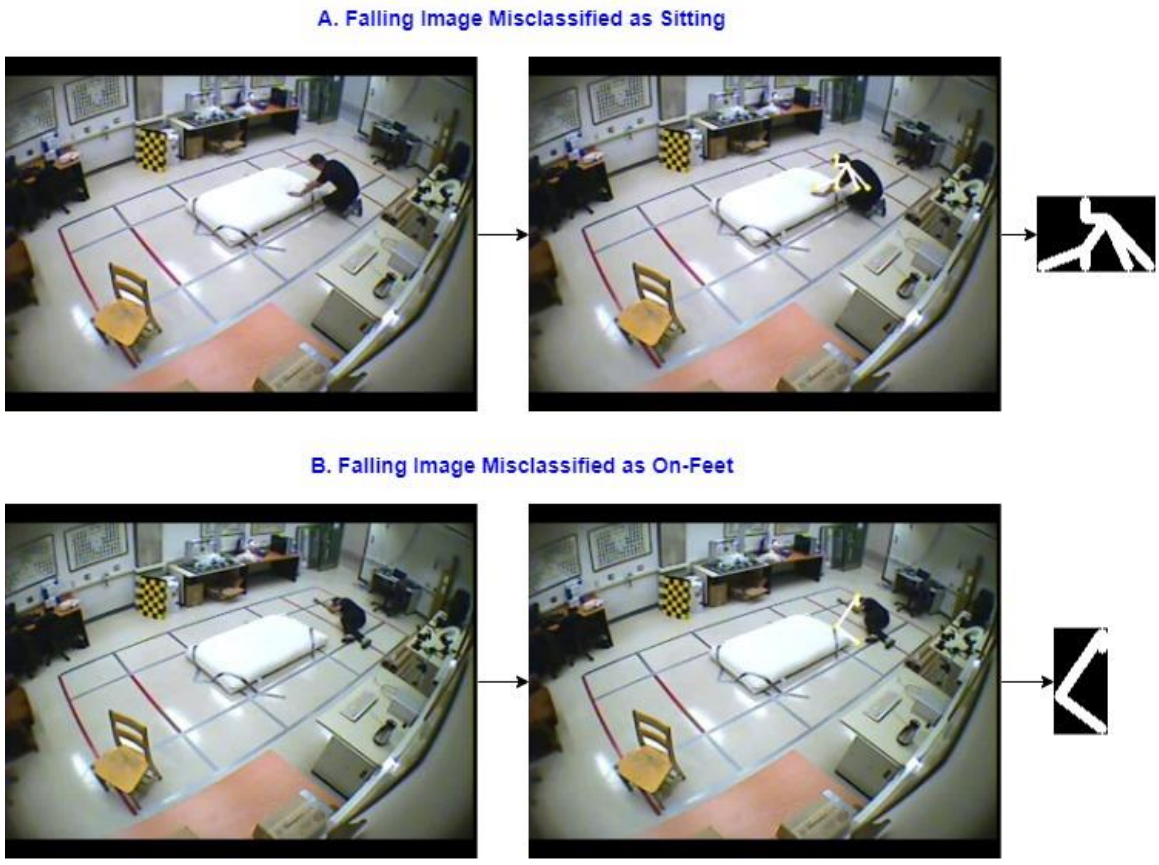

Figure 10. Fall-labeled images and their misclassification

We ran the same binary skeletal image CNN model three times but with different input data. The first model was trained with only *Fall* and *On-Feet* images, the second one with *Fall* and *Sit* images, and the third one with *On-Feet* and *Sit* images. As can be seen from Table 4, these three models all yield better validation accuracies than that of the three-class fall detection model. Figure 10 displays the three confusion matrices for

these three models. It can be observed that the binary images of a Fall-labeled image and an On-Feet-labeled image have similarities that the model does not classify very well, as 12.5% (5 out of 40) of the Fall-labeled images were misclassified as On-Feet. By observing Figure 9, it is reasonable to conclude that the main factor for the misclassification of the falling images is the lack of accuracy in creating the skeletal representations of the human subjects in the images. The skeletal representations of the images are constructed incorrectly, as the key joint locations of the humans are not identified correctly.

Table 4. Accuracies of the three models trained with binary data

Model Classes	**Accuracy**
Fall and On-Feet	0.93
Fall and Sit	0.93
On-Feet and Sit	0.96

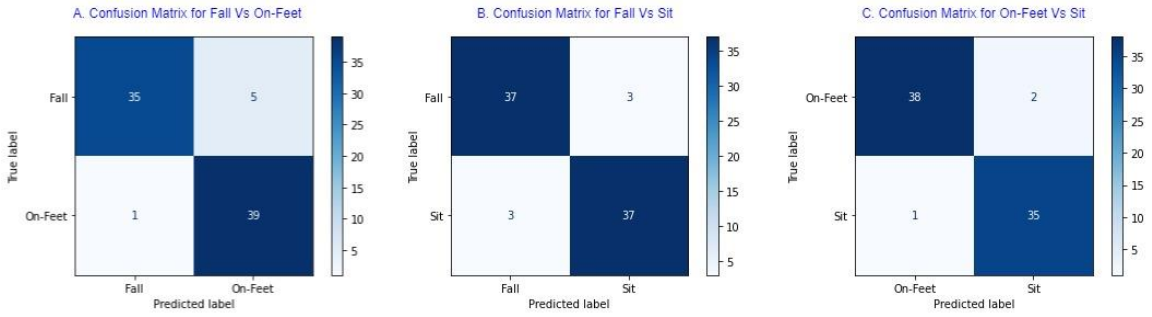

Figure 11. Confusion matrices of the three models trained with binary data

CHAPTER V

CONCLUSION

The image processing technique proposed in this work utilizes machine learning, specifically image classification, to create a human-robot interactive game and a fall detection system. These systems were based on the binary human skeletal representation of raw RGB images. A three-layer CNN model was built on these binary images to detect different human poses. These human poses include the eight poses used for the human-robot interactive game (see Table 1) and the three poses used for the fall detection system (see Table 3). The experimental results show that an accuracy of 92.55% is obtained for the human-robot interactive game, while an accuracy of 90.79% is obtained for the fall detection system. The image processing method proposed in this work is beneficial as the information in images is not affected by background noise or lighting conditions, and the training and validation times of the applied models decrease. For future work, we would like to expand the fall detection algorithm to detect multiple falls in one scene/image. Our fall detection system currently only processes images with one human subject. This issue can be addressed by using a similar process used to give the human-robot interactive game multi-player functionality.

BIBLIOGRAPHY

[1] L. Guerrouj, "Machine Learning: 6 Real-World Examples," Salesforce EMEA Blog, Sep. 09, 2021. https://www.salesforce.com/eu/blog/2020/06/real-world-examples-of-machine-learning.html

[2] "Deep learning vs machine learning," Zendesk. https://www.zendesk.com/blog/machine-learning-and-deep-learning

[3] "Image Recognition with Deep Neural Networks and its Use Cases," AltexSoft. https://www.altexsoft.com/blog/image-recognition-neural-networks-use-cases/

[4] Y. Yu, E. Favour, and P. Mazumder, "Convolutional Neural Network Design for Breast Cancer Medical Image Classification," 2020 IEEE 20th International Conference on Communication Technology (ICCT), pp. 1325–1332, Oct. 2020, doi: 10.1109/icct50939.2020.9295909.

[5] G. Liang, H. Hong, W. Xie, and L. Zheng, "Combining Convolutional Neural Network With Recursive Neural Network for Blood Cell Image Classification," IEEE Access, vol. 6, pp. 36188–36197, 2018, doi: 10.1109/access.2018.2846685.

[6] T. Shaily and S. Kala, "Bacterial Image Classification Using Convolutional Neural Networks," 2020 IEEE 17th India Council International Conference (INDICON), pp. 1–6, Dec. 2020, doi: 10.1109/indicon49873.2020.9342356.

[7] N. D. Navghare and L. M. Gladence, "End to End Learning Human Pose Detection Using Convolutional Neural Networks," Machine Learning and Information Processing, pp. 135–142, 2021, doi: 10.1007/978-981-33-4859-2_13.

[8] "By 2030, U.S Demographic Milestone Begins to Lower Caregiver Ratio – West View Nursing and Rehabilitation Center," WEST VIEW NURSING AND REHABILITATION CENTER. http://westviewnursing.com/by-2030-u-s-demographic-milestone-begins-to-lower-caregiver-ratio (accessed Nov. 11, 2022).

[9] "Integrated care for older people: guidelines on community-level interventions to manage declines in intrinsic capacity," www.who.int. https://www.who.int/publications/i/item/9789241550109

[10] M. C. Author, "Fall Injuries," Fight Nursing Home Abuse. https://www.fightnursinghomeabuse.com/nursing-home-abuse-and-neglect/fall-injuries/

[11] IBM, “Computer Vision,” Ibm.com, 2019. https://www.ibm.com/topics/computer-vision

[12] S. K, B. S, and P. M. B, “Social Distance Identification Using Optimized Faster Region-Based Convolutional Neural Network,” 2021 5th International Conference on Computing Methodologies and Communication (ICCMC), pp. 753–760, Apr. 2021, doi: 10.1109/iccmc51019.2021.9418478.

[13] Z. Cao, G. Hidalgo Martinez, T. Simon, S.-E. Wei, and Y. A. Sheikh, “OpenPose: Realtime Multi-Person 2D Pose Estimation using Part Affinity Fields,” IEEE Transactions on Pattern Analysis and Machine Intelligence, pp. 172–186, 2019, doi: 10.1109/tpami.2019.2929257.

[14] V. Gupta, “Multi Person Pose Estimation in OpenCV using OpenPose,” LearnOpenCV, Sep. 11, 2018. https://learnopencv.com/multi-person-pose-estimation-in-opencv-using-openpose/ (accessed Nov. 11, 2022).

[15] T.-Y. Lin et al., “Microsoft COCO: Common Objects in Context,” arXiv:1405.0312 [cs], Feb. 2015, Accessed: Nov. 06, 2022. [Online]. Available: http://arxiv.org/abs/1405.0312

[16] S. Albawi, T. A. Mohammed, and S. Al-Zawi, “Understanding of a Convolutional Neural Network,” 2017 International Conference on Engineering and Technology (ICET), pp. 1–6, Aug. 2017, doi: 10.1109/icengtechnol.2017.8308186.

[17] “What is a Convolutional Layer?,” Databricks. https://www.databricks.com/glossary/convolutional-layer#:~:text=Convolutional%20layers%20apply%20a%20convolution

[18] Piotr Skalski, “Gentle Dive into Math Behind Convolutional Neural Networks,” Medium, Apr. 12, 2019. https://towardsdatascience.com/gentle-dive-into-math-behind-convolutional-neural-networks-79a07dd44cf9

[19] K. O'shea and R. Nash, “An Introduction to Convolutional Neural Networks,” 2015. [Online]. Available: https://arxiv.org/pdf/1511.08458.pdf

[20] N. Alipour, O. Tarkhaneh, M. Awrangjeb, and H. Tian, “Flower Image Classification Using Deep Convolutional Neural Network,” 2021 7th International Conference on Web Research (ICWR), pp. 1–4, May 2021, doi: 10.1109/icwr51868.2021.9443129.

[21] B. V. Krishna, P. N. R. Bodavarapu, P. Santhosh, and P. V. V. S. Srinivas, “Chest Computed Tomography Scan Images for Classification of Coronavirus by Enhanced Convolutional Neural Network and Gabor Filter,” 2021 5th International Conference on Intelligent Computing and Control Systems (ICICCS), pp. 825–831, May 2021, doi: 10.1109/iciccs51141.2021.9432345.

[22] D. H. Fuadi, D. Novita, and M. Taufik, “Socially Assistive Robot Interaction by Objects Detection and Face Recognition on Convolutional Neural Network for Parental Monitoring,” 2021 International Conference on Artificial Intelligence and Mechatronics Systems (AIMS), pp. 1–6, Apr. 2021, doi: 10.1109/aims52415.2021.9466091.

[23] R. Hasib, K. N. Khan, M. Yu, and M. S. Khan, “Vision-based Human Posture Classification and Fall Detection using Convolutional Neural Network,” 2021 International Conference on Artificial Intelligence (ICAI), pp. 74–79, Apr. 2021, doi: 10.1109/icai52203.2021.9445263.

[24] J. Wu, K. Wang, B. Cheng, R. Li, C. Chen, and T. Zhou, “Skeleton Based Fall Detection with Convolutional Neural Network,” 2019 Chinese Control And Decision Conference (CCDC), pp. 5266–5271, Jun. 2019, doi: 10.1109/ccdc.2019.8832565.

[25] E. Auvinet, C. Rougier, J. Meunier, A. St-Arnaud, and J. Rousseau, “Multiple Cameras Fall Data Set,” Technical report 1350, DIRO - Université de Montréal, Jul. 2010