INSTITUTO
UNIVERSITÁRIO
DE LISBOA

Application of Computer Vision Techniques for Tracking Judo Athletes

André Pereira de Almeida

Master in Computer Science

Supervisor:
PhD Tomás Gomes Silva Serpa Brandão, Assistant Professor,
ISCTE - Instituto Universitário de Lisboa

October, 2024

TECNOLOGIAS
E ARQUITETURA

Department of Information Science and Technology

Application of Computer Vision Techniques for Tracking Judo Athletes

André Pereira de Almeida

Master In Computer Science

Supervisor:
PhD Tomás Gomes Silva Serpa Brandão, Assistant Professor,
ISCTE - Instituto Universitário de Lisboa

October, 2024

Dedico este trabalho aos meus pais e ao meu irmão,
pela ajuda indispensável para iniciar os meus estudos
e por todo o apoio ao longo desta jornada.
Agradeço profundamente por estarem
sempre ao meu lado.
Obrigado

Acknowledgments

Completing this project feels like reaching the peak of a long, challenging journey, one made possible by the support of many remarkable people.

First, I owe a heartfelt thanks to my supervisor, whose dedication and readiness to help was a constant source of guidance and reassurance. His commitment to my success and his genuine willingness to provide clarity made me realize just how fortunate I was, especially when I saw how differently others' journeys could be.

My deepest gratitude goes to my family, my parents and my brother, for their unshakeable support and encouragement. They provided a foundation of love, wisdom, and resilience that kept me grounded and driven. Through every high and low, they believed in me and pushed me to go further, even when I doubted myself. I am truly grateful for their constant presence in my life.

To my friends, thank you for the laughs, for always being there to listen, and for pulling me out of my own head whenever I needed it. Your support brought me balance and made even the hardest days brighter.

A special thanks to my colleagues at ISCTE-IUL, whose shared experiences, camaraderie, and friendship added so much meaning to this journey. Working together with such talented and determined people has been a privilege.

Finally, I want to thank the judo community for being my sanctuary. Practicing and spending time with all of you brought joy, connection, and a healthy escape when I needed it most. The friendships and experiences I found there helped keep me centered and sane through it all.

To each and every one of you who offered support, guidance, or just a moment of kindness: thank you.

Resumo

A tecnologia de visão por computador tem revolucionado vários desportos, mas a sua aplicação no judo continua relativamente inexplorada. Esta dissertação investiga o desenvolvimento de um sistema automatizado para classificar projeções de judo, utilizando técnicas de estimação de pose e aprendizagem automática. A pesquisa aborda duas questões principais: identificar os modelos mais eficazes para a deteção da pose dos atletas e determinar a viabilidade de uma classificação precisa das projeções a partir de dados de vídeo.

O estudo avalia vários modelos de deteção de objetos e estimação de pose, destacando-se o YOLOv8 e o YOLO-NAS como as soluções mais promissoras para acompanhar os movimentos dos judocas. Foi desenvolvida uma abordagem de classificação baseada em redes LSTM duplas para processar a sequência temporal dos dados de pose, alcançando uma taxa de acertos de 79,17% no conjunto de testes na distinção entre diferentes técnicas de projeção e ações que não envolvem quedas.

Embora demonstre a viabilidade da classificação automatizada de projeções, a pesquisa identificou também desafios importantes, especialmente na manutenção de uma estimação de pose precisa durante sequências complexas de projeções e em situações de oclusão. Os resultados sugerem que as limitações atuais podem ser superadas através da expansão do conjunto de dados, melhorias arquiteturais e da evolução natural da tecnologia de estimação de pose.

Esta pesquisa estabelece uma base sólida para a análise automatizada de técnicas de judo, destacando o seu potencial como um marco fundamental para futuros avanços na análise desportiva, particularmente no judo. À medida que a tecnologia de estimação de pose evolui e os conjuntos de dados se tornam mais completos, a metodologia aqui apresentada abre caminho para ferramentas de análise de desempenho mais sofisticadas e inovações. Estes sistemas estão destinados a alcançar maior precisão e fiabilidade, conduzindo a aplicações práticas que podem beneficiar significativamente toda a comunidade do judo.

Palavras-chave: Judo; Classificação de Projeções; Visão por Computador; Machine Learning; Estimação da Pose

Abstract

Computer vision technology has revolutionized various sports, yet its application in judo remains relatively unexplored. This dissertation investigates the development of an automated system for classifying judo throws using pose estimation and machine learning techniques. The research addresses two primary questions: identifying the most effective models for athlete pose detection and determining the feasibility of accurate throw classification from video data.

The study evaluates multiple object detection and pose estimation models, with YOLOv8 and YOLO-NAS emerging as the most promising solutions for tracking judoka movements. A classification approach utilizing dual LSTM networks was developed to process the temporal sequence of pose data, achieving an accuracy of 79.17% on the test dataset in distinguishing between different throwing techniques and non-throw actions.

While demonstrating the viability of automated throw classification, the research also identified key challenges, particularly in maintaining accurate pose estimation during complex throwing sequences and occlusions. The findings suggest that current limitations could be addressed through dataset expansion, architectural improvements, and the natural evolution of pose estimation technology.

This research establishes a strong foundation for automated judo technique analysis, highlighting its potential as a pivotal building block for future advancements in sports analysis, particularly in judo. As pose estimation technology evolves and datasets become more comprehensive, the methodology introduced here paves the way for more sophisticated performance analysis tools and innovations. These systems are poised to achieve greater accuracy and reliability, ultimately leading to practical applications that can significantly benefit the entire judo community.

Keywords: Judo; Throw Classification; Computer Vision; Machine Learning; Pose Estimation

Index

List of Figures

List of Tables

1. Introduction

1.1 Context

In the constantly changing landscape of sports, the integration of technology has emerged as a critical factor for advancing not only athletic performance but also enhancing the overall experience for athletes, coaches, and spectators. Technological innovations play a key role in optimizing training processes, refining techniques, improving safety standards, and making sports more engaging for fans. Among the various technological advancements, computer vision has gained prominence for its potential to transform sports by offering sophisticated methods for motion analysis, automated performance evaluation, and the development of innovative training tools.

One of such sports is judo, a traditional martial art with a commitment to evolve. Each year, judo demonstrates its adaptability by revising its ruleset, aiming to enhance the sport's entertainment value, dynamism, and safety. However, it's worth noting that computer vision-based tools are not yet sufficiently explored in judo, especially compared to other more popular sports such as football, basketball, and cricket [1], [2]. Despite this, there are numerous opportunities to leverage computer vision in judo to enhance its practice and competition.

Several areas in judo could benefit from the application of computer vision. For example, object identification techniques could be used to track and identify athletes during matches, providing real-time data on position, movement, and key actions. Additionally, pose estimation algorithms could facilitate the automated recognition of judo throws, enabling the development of systems for maintaining detailed athlete statistics, assisting referees by validating scoring actions, and enhancing match analysis. Furthermore, for viewer experience, the use of action recognition techniques during a judo match could generate highlight reels that showcase the most exciting aspects of the sport.

This dissertation focuses on the classification of judo throws using computer vision techniques, developing a system capable of recognizing and accurately categorizing different judo throws based on video data by leveraging pose estimation algorithms. By demonstrating the feasibility of identifying specific judo throws like *uchi-mata* and *ippon-seio-nage*, this research provides a foundational building block for more sophisticated sports analysis tools. The model will demonstrate how integrating computer vision can enhance both the practice and competitive aspects of this traditional martial art, laying groundwork for future innovations such as automated match annotation systems and advanced performance analysis.

1.2 Problem

In contrast to certain sports, the application of computer vision techniques in judo presents a few challenges. Firstly, the problem lies in the limited exploration of computer vision in judo compared to other more popular sports. This lack of exploration is evident in the absence of datasets with annotations specific to judo, unlike other more popular sports, meaning that new datasets must be created.

Additionally, judo's close-contact nature leads to the occurrence of occlusions, obstructing the visibility of some areas resulting in loss of information, one of computer vision's weaknesses [3]. Making it more difficult to keep track of objects and movement, complicating the analysis process. Therefore, addressing these challenges is crucial for advancing the application of computer vision in judo and unlocking its potential benefits for the sport.

1.3 Research Questions and Objectives

This research seeks to develop a robust classification model for judo throws using computer vision techniques. The study is guided by the following research questions:

- What are the best models to detect the athletes' pose?
- How accurately can you classify judo throws from video data?

To address these research questions, the project will be guided by the following objectives, which outline the necessary steps to achieve the study's aims:

- **Create a dataset of judo throws:** Compile a diverse and comprehensive dataset comprising video recordings of different types of judo throws performed by skilled judokas. Alongside dataset creation, the videos will be annotated with detailed labels, including the type of throw and key frames that highlight important movements and poses. These annotations will serve as the ground truth for training and evaluating the learning models.

- **Evaluate object detection and pose estimation models:** The second objective is to assess various object detection and pose estimation models to determine their effectiveness in detecting athletes and their movements during judo throws. This evaluation will involve implementing multiple models and comparing their performance using metrics such as precision, recall, and intersection over union (IoU).

- **Train the selected model to optimize classification performance:** After identifying the best-performing model, further training and fine-tuning will be conducted using the annotated dataset to maximize accuracy.

1.4 Methodology and dissertation structure

To complete the established objectives and since one of the products of this dissertation will be an artifact, a system capable of identifying what type of throw is executed in a video. The Design Science Research (DSR) Process [3] was chosen as the methodology, which as illustrated in Figure 1, divides the whole project into the following steps:

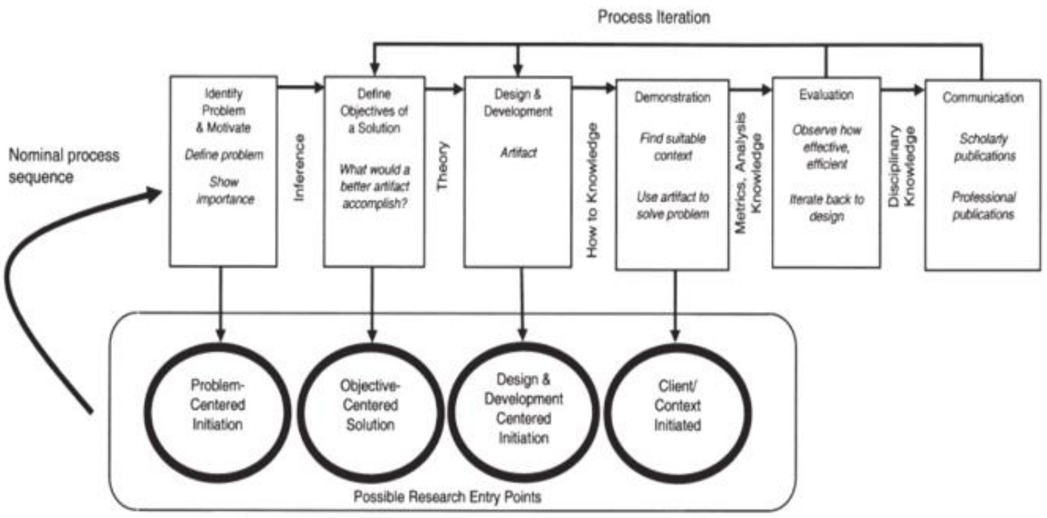

Process Iteration
Nominal process sequence
Identify Problem & Motivate
Define problem
Show importance
Inference
Define Objectives of a Solution
What would a better artifact accomplish?
Theory
Design & Development
Artifact
How to Knowledge
Demonstration
Find suitable context
Use artifact to solve problem
Metrics, Analysis Knowledge
Evaluation
Observe how effective, efficient
Iterate back to design
Disciplinary Knowledge
Communication
Scholarly publications
Professional publications
Problem-Centered Initiation
Objective-Centered Solution
Design & Development Centered Initiation
Client/ Context Initiated
Possible Research Entry Points

Figure 1 DSR Methodology Process Model [4]

Problem identification and motivation: This is the step for defining the problem and justify the value of its solution. Here it was identified that judo is a martial art that is not taking advantage of what computer vision systems can provide by doing so there is a significant opportunity for improvement within the sport.

Define the objectives for a solution: Based on the problem identified, the objectives for a solution are described. These objectives should be viable and realistic. At this stage, a system for identifying different types of judo throws was defined as the main objective.

Design and development: Here, the design and functionality of the artifact are determined before starting the creation process of the artifact. This entails the construction of a dataset with annotations for training and evaluation. Additionally, the optimal model for pose estimation is carefully selected, considering the compatibility with judo-specific movements. Subsequently, the model for identifying various types of judo throws is developed.

Demonstration: In this activity, the artifact is put to the test, by trying to solve the problem or parts of it. This may involve practical testing, simulations, or other forms of validation. Tests with a small portion of the dataset reserved for validation will be used.

Evaluation: Following the demonstration, measure the effectiveness of the artifact as a solution to the problem. At the end of this step, depending on the results, the design of the artifact may have to be iterated to try to improve its results. In this part, the tests done in the previous point will be used to assess the model's performance.

Communication: In the last step, make the problem and the artifact accessible through an appropriate mode of communication to the audience. This will involve the submission of this dissertation and presentation of its findings.

Following the DSR steps, the dissertation is structured into four main chapters. Chapter 2 presents a brief literature review, combining background concepts and related work providing a foundation on computer vision in sports, particularly focusing on pose estimation and action recognition, while identifying gaps in judo-specific research. Chapter 3 covers the development process, detailing the creation of a comprehensive dataset of annotated judo throws, model selection, training, and evaluation using metrics like precision and recall to assess the model's performance. The final chapter concludes with a summary of the research findings, a discussion of limitations, and recommendations for future work to further advance judo throw classification using computer vision techniques.

2. Literature Review

This chapter focuses on exploring information related to the topic of the dissertation, starting with researching key concepts associated with computer vision. Subsequently, it presents an overview of existing research done in this field, providing insights into the current state of computer vision research and its diverse applications.

2.1 Background Concepts

2.1.1. Computer vision

Computer vision is a field that aims to understand and replicating the visual perception of humans by using mathematical models and computational techniques. Despite humans effortlessly perceiving the three-dimensional structure of their surroundings and interpreting visual cues like shapes, textures, and objects, replicating this capacity in computers presents formidable challenges. While considerable progress has been made in recent decades, particularly in areas such as radiometry, optics, sensor design, and computer graphics, achieving a level of image understanding comparable to human perception remains an ongoing pursuit [5].

2.1.2. Convolutional Networks (CNN)

Traditional fully connected networks pose a few challenges when processing images, due to the high dimensionality of visual data, leading to substantial requirements for training data, memory, and computational resources. This poses obvious practical problems in terms of the required training data, memory, and computation resources.

Additionally, nearby pixels of an image are statistically related by their values. However, fully connected networks cannot take advantage of this fact, they need to learn the pattern of pixels at every position, making it very inefficient.

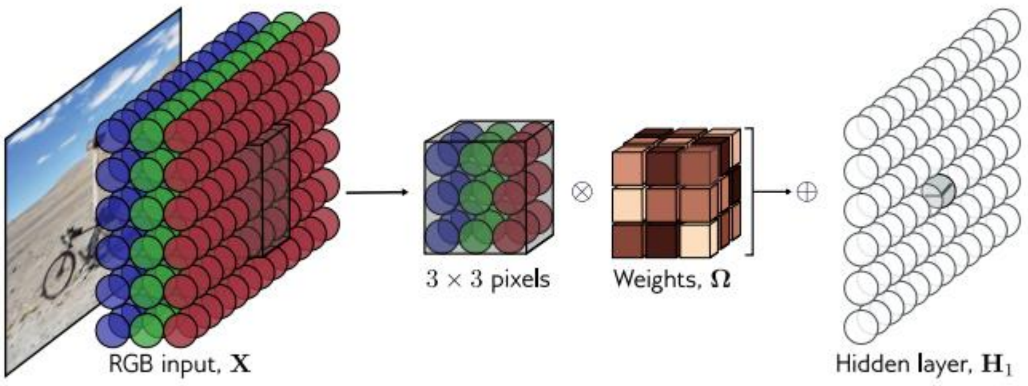


Figure 2 Convolution applied to an image [6]

This is where the convolutional network comes in, CNNs are comprised of convolutional, pooling and fully connected layers. The convolutional layer applies filters to the image, aiming to capture the most relevant features. These layers increase in complexity, the earlier ones detecting more basic features like textures and shapes, as illustrated in Figure 2, while the last ones produce higher level features. Pooling layers are typically placed between convolutional layers to reduce the dimensionality of the feature maps, thereby decreasing the computational load and helping to retain essential features while discarding less important information. Finally, fully connected layers are used to flatten the pooled feature maps and generate outputs for classification, as shown in Figure 3.

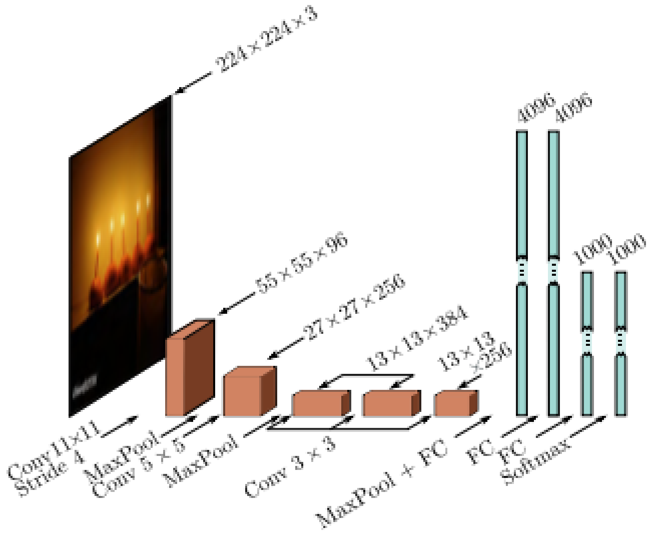


Figure 3 AlexNet, example of a CNN architecture [6]

By processing an image independently of its region and taking advantage of the relationship of nearby pixels, CNNs offer a more efficient approach to processing images compared to traditional fully connected networks [5], [6].

2.1.3. Object Detection

Object detection is a computer vision task that involves identifying and locating objects in an image or video, generally with the representation of a bounding box. This an area of computer vision that improved greatly with the introduction of CNNs.

Object detectors can be categorized into single-stage, two-stage, and the more recent transformer-based detectors. Single-stage detectors classify and localize semantic objects in a single shot using dense sampling, while two-stage detectors have a separate module to generate region proposals in the first stage and then classify and localize them in the second stage. Transformer-based detectors, inspired by its use natural language processing, utilize a direct set prediction model that uses a transformer encoder-decoder architecture to predict all objects at once.

Each of these different types of detectors have their own characteristics and advantages. Single-stage detectors are more suited for real-time applications due to its execution time and simpler design, while two-stage detectors generally take longer to generate proposals but can handle an arbitrary number of objects. Transformer-based detectors, although requiring comparatively more data to train, have achieved state-of-the-art performance in tasks that prioritize improved accuracy and managing complex interactions between objects, compared to other sequential architectures [7].

YOLO (You Only Look Once) network is an example of a single-stage detector where the input image or video frame starts by being reshaped and divided it into a grid of cells, as illustrated in Figure 4. Within each of them, the most likely class is calculated based on the features extracted. Using these predicted class probabilities, the network draws bounding boxes around the detected objects, each with an associated confidence score indicating the likelihood of the box accurately classified. At the end, bounding boxes below a certain threshold are discarded, only showing the boxes with higher confidence [6].

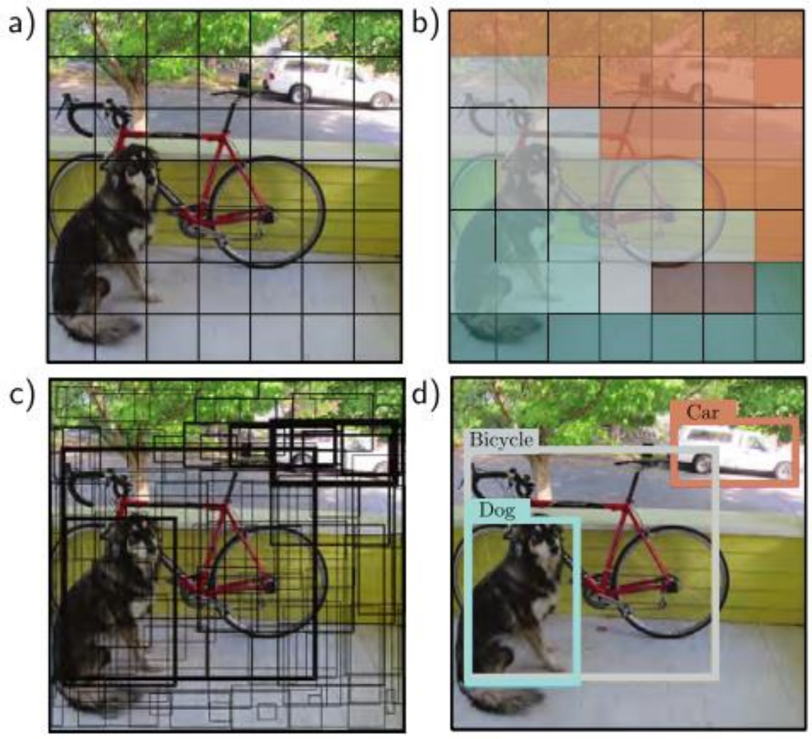


Figure 4 YOLO object detection [6]

An instance of a two-stage detector is the Mask R-CNN, as represented in Figure 5, during the initial stage of image processing starts by going through convolutional layers to extract its features, forming features maps, which are then processed by a Region Proposal Network (RPN), identifying areas of interest in the image. In the second stage, object identification and the corresponding mask is done in each area of interest. The resulting processed image has bounding boxes and segmentation masks delineating each object.

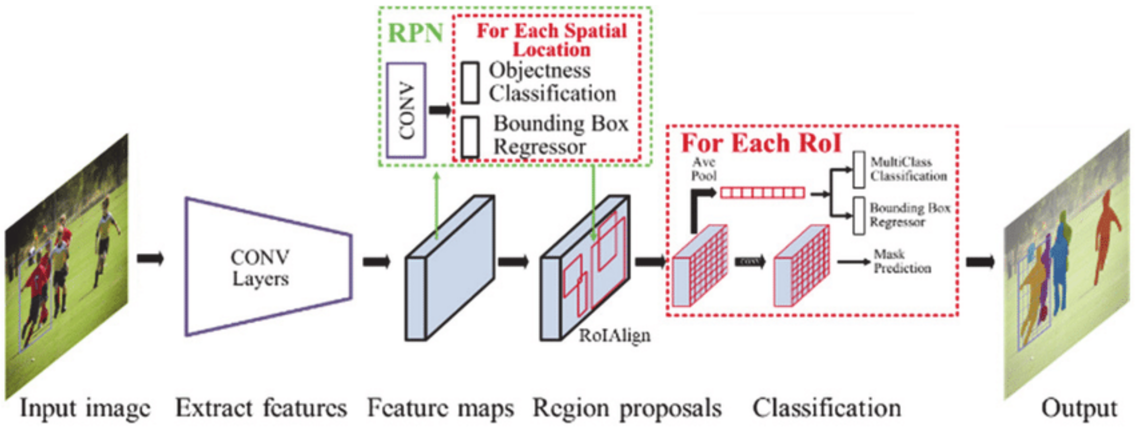


Figure 5 Diagram of Mask RCNN framework [8]

Finally, detection transformer (DETR) is formed by three components, illustrated in Figure 6: a CNN backbone for extracting features, a transformer encoder-decoder and feed forward network (FFN) for the detection prediction. The backbone analyzes the input image and produces an activation map. The transformer encoder diminishes the channel dimension while employing multi-head self-attention and feed-forward networks. Meanwhile, the transformer decoder engages in parallel decoding of N object embeddings and predicts box coordinates and class labels via object queries. DETR processes all objects collectively by leveraging pairwise relations, benefiting from the context of the entire image [9].

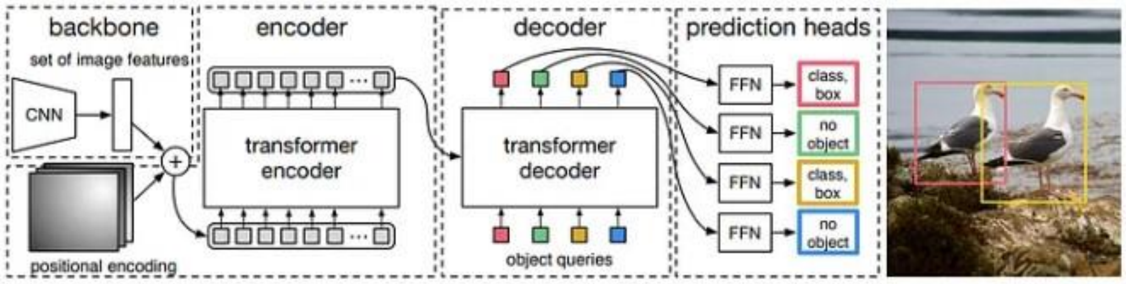


Figure 6 Diagram of DETR framework [9]

2.1.4. Pose Estimation

Pose Estimation is a computer vision technique aimed at identifying and locating various parts of the human body (key points) within an image or video frame, and subsequently connecting them to form a skeletal representation of a person. This process enables machines to understand and interpret human poses, facilitating applications such as action recognition and gesture analysis.

Pose estimation can be implemented in two forms: 2D, where key points are tracked on a flat plane, or 3D, where depth information is added for more spatial accurate predictions. There are also two main approaches to make these estimations, the bottom-up method, which starts by detecting each key point and then groups them to form the pose of an individual, the other method being the top-down approach, which first identifies a person within a bounding box and then estimates the body joints within said box [5].

2.2 Related Work

In the study of judo as a sport, various analytical approaches have been employed, even before the advent of computer vision technology. A notable example is the work of Kato & Yamagiwa [10], who analyzed throws performed at World Judo Championships and their respective effectiveness. Their study provided valuable insights about the relationships between categories of throwing techniques, represented in Figure 7, while aligning with the principles outlined by Jigoro Kano [11], highlighting the importance of understanding the dynamics of judo techniques and their practical application in competitive settings.

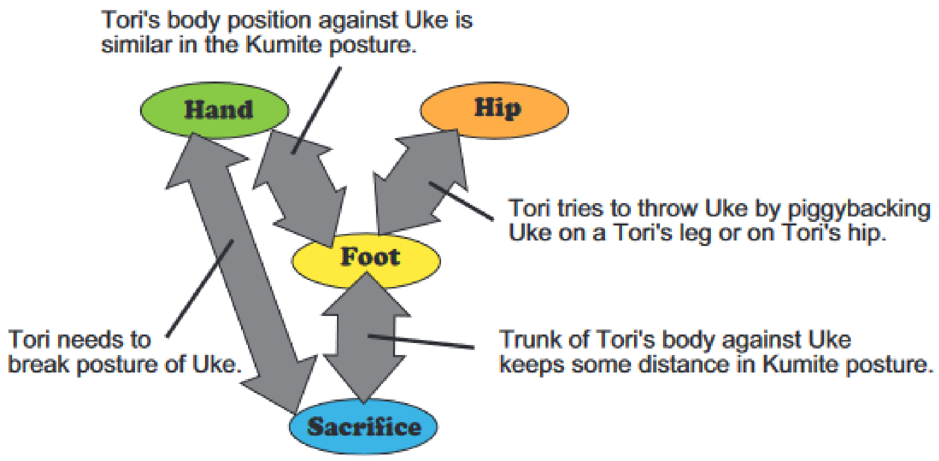

Tori's body position against Uke is similar in the Kumite posture.
Hand
Hip
Tori tries to throw Uke by piggybacking Uke on a Tori's leg or on Tori's hip.
Foot
Tori needs to break posture of Uke.
Trunk of Tori's body against Uke keeps some distance in Kumite posture.
Sacrifice

Figure 7 The relationships among the categories of the throwing techniques [10]

The application of computer vision in combat sports presents unique challenges, as demonstrated by research in related fields. In rugby, a sport that shares judo's challenge of frequent player occlusions, Nonaka [12] developed a system to detect high-risk tackles from video footage, testing the performance of multiple models. Despite the challenging context, their system successfully identified 50% of dangerous situations. However, significant issues emerged, including slow processing speed, difficulties in pose estimations during player occlusions, and the occurrence of false positives. These challenges parallel those faced in judo analysis, particularly during dynamic throwing situations.

Computer vision applications in martial arts have shown promising results, particularly in less contact-intensive disciplines. Nurahmadan, Jayanta and Pradnyana [13] employed pose estimation using OpenPose to classify four Taekwondo movements based on body keypoint values. While successful, their system was limited to scenarios where the athlete's front side faced the camera, avoiding the complexities present in contact-heavy martial arts like judo. Similarly, Quinn and Corcoran [14] explored the feasibility of generating statistics for combat sports athletes in boxing and MMA using object detection and pose estimation with YOLOv5. Their system tracked metrics such as the number of performed kicks or punches along with corresponding hits and misses, though it too struggled with evaluating more contact-heavy moments within matches.

Addressing the challenges of analyzing complex martial arts interactions, recent research by Hudovernik and Skocaj [15] focused on detecting combat positions and implementing automatic scoring in Jiu-jitsu. Their study achieved impressive results in identifying various match situations, even in highly occluded positions. The framework they presented not only successfully recognized different grappling positions but also implemented a scoring system that closely matched expert judges' assessments. Additionally, they proposed an innovative approach for identifying judges' decisions through gesture recognition. These advances in analyzing ground-based grappling techniques demonstrate the potential for similar applications in judo, while acknowledging the unique challenges posed by the dynamic nature of throwing techniques and the need for specialized approaches in judo-specific applications.

Through this review of existing research, two significant gaps become apparent. First, there is a notable scarcity of computer vision applications specifically designed for judo. While traditional video analysis methods have contributed to understanding judo techniques, the potential of modern computer vision approaches remains largely unexplored in this domain. Second, existing studies in martial arts movement recognition have primarily focused on techniques with less complexity and fewer occlusions. These studies, while valuable, deal with movements that are more easily captured and analyzed compared to judo throws, which involve intricate interactions between two athletes in close contact, rapid movements, and frequent occlusions. This research aims to address these gaps by developing a specialized system for judo throw classification that can handle the unique challenges posed by judo techniques.

3. Development

To achieve the objectives proposed the whole process was envisioned beforehand, as depicted in Figure 8.

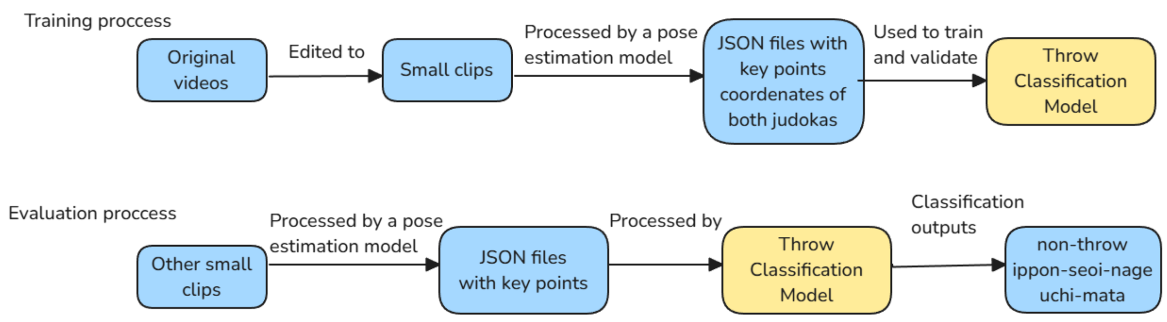

Training proccess
Original videos
Edited to
Small clips
Processed by a pose estimation model
JSON files with key points coordenates of both judokas
Used to train and validate
Throw Classification Model
Evaluation proccess
Other small clips
Processed by a pose estimation model
JSON files with key points
Processed by
Throw Classification Model
Classification outputs
non-throw
ippon-seoi-nage
uchi-mata

Figure 8 Diagram of the developed system

The development process begins with collecting videos of judokas performing throws. These videos serve as the foundation for the dataset and are subsequently edited into shorter clips, each focusing on a single throw. The aim is to capture the key moments of the throws while excluding irrelevant content.

After editing, the clips are processed using a pose estimation model, which generates JSON files containing the coordinates of various key points on the judokas' bodies for each frame. These key points represent significant joints, such as the shoulders, elbows, knees, and ankles, allowing for a detailed analysis of the athletes' movements over time. The pose estimation data provides a frame-by-frame record of how these key point coordinates change throughout each clip, effectively capturing the dynamic nature of the throws.

The resulting JSON files serve as the input data for training and validating the classification model. Each file is associated with a label specifying the type of action depicted in the clip: either a specific judo throw (e.g., *ippon-seoi-nage*, *uchi-mata*) or a non-throw action. The classification model is designed to learn patterns in the key point data that distinguish different types of movements, enabling it to differentiate between the throws and non-throws based on the unique movement characteristics.

The development process involves several key stages: first, collecting and preparing the video data; next, evaluating multiple models to determine the most promising one for this task; then, processing the clips with the pose estimation model to extract key point coordinates and finally, fine-tuning the selected model for optimal accuracy. Each of these steps is crucial for ensuring the final model's ability to accurately classify judo actions. The design, implementation, and evaluation of this system will be discussed in detail in the following chapters.

3.1 Understanding and creating the dataset

In machine learning applications, the quality and suitability of the training data are crucial factors that determine the system's effectiveness. In the context of judo throw classification, the existing databases are insufficient, as they typically focus on competition results or distinguishing between general physical activities rather than differentiating specific judo techniques. Therefore, a custom dataset was created to address this gap, with a focus on two specific throws: *ippon-seoi-nage* and *uchi-mata*.

To establish a basis for dataset creation, it is essential to understand the characteristics and mechanics of the two selected judo throws. Both *ippon-seoi-nage* and *uchi-mata* are standing throws, which adds complexity to the classification task. Unlike sacrifice throws, where the thrower deliberately falls to the ground, making visual differentiation easier, standing throws involve movements that the model needs to learn to distinguish.

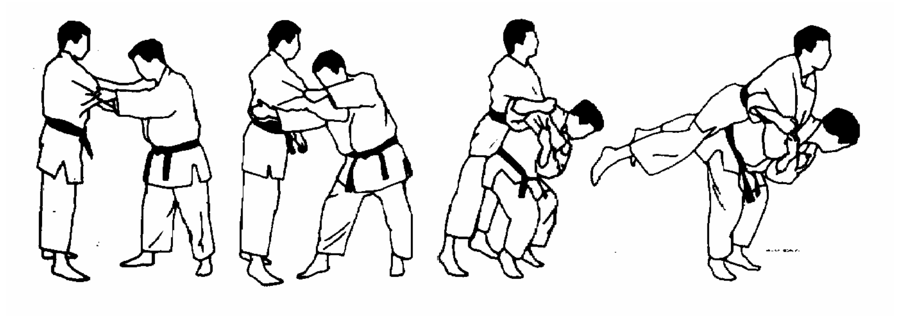


Figure 9 Diagram for ippon-seoi-nage

As illustrated in Figure 9, *ippon-seoi-nage* is performed by the thrower (*tori*) floating the opponent's (*uke*) balance forward. The thrower pivots to the left while slipping their right arm underneath the opponent's chest and up under their right armpit, securing a grip on the opponent's right sleeve or shoulder. With the opponent pulled onto the thrower's back, the thrower drops their center of gravity and, using precise hip rotation, throws the opponent over their right shoulder.

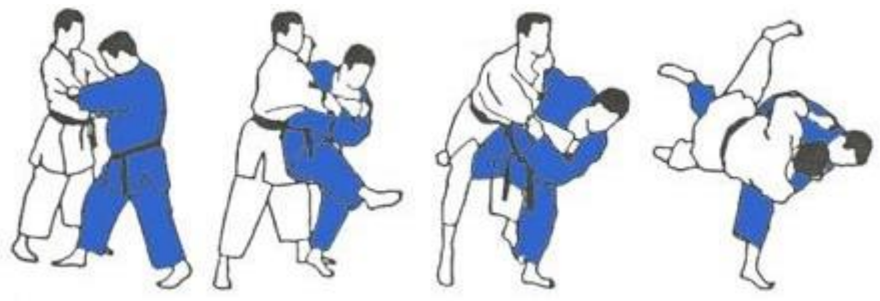


Figure 10 Diagram for uchi-mata

As for the *uchi-mata* throw, represented in Figure 10, it involves the thrower destabilizing the opponent by opening their stance slightly and breaking their balance to the front or right front corner, causing the opponent to bend forward. The thrower then sweeps the inside of the opponent 's left thigh deeply with the back of their own right thigh, all while twisting their body to the left. The sweeping motion combined with the rotational force drives the opponent over the thrower's hip.

These throws were chosen not only because of their technical relevance in judo but also because they present a classification challenge due to their similar starting positions and movement dynamics. Additionally, non-throw clips were included to provide a baseline for the model to differentiate between throw and non-throw actions, enhancing the robustness of the classification system.

An additional challenge in classifying judo throws lies in the wide variety of body positions and frequent occlusions that occur during execution. Unlike many martial arts movements that remain relatively consistent in form, judo throws involve close-contact maneuvers, often resulting in one athlete partially or completely obscuring the other. For instance, the moment a throw reaches its peak force, the thrower's body may block the view of the opponent, creating gaps in the visual data available for accurate classification. Uncommon positions, such as low stances or mid-air rotations, add further complexity, as these varied postures are challenging for a model to learn from limited examples alone. Figure 11 highlights these difficulties, showing instances of throws where occlusions or atypical angles present significant obstacles for accurate detection and classification.

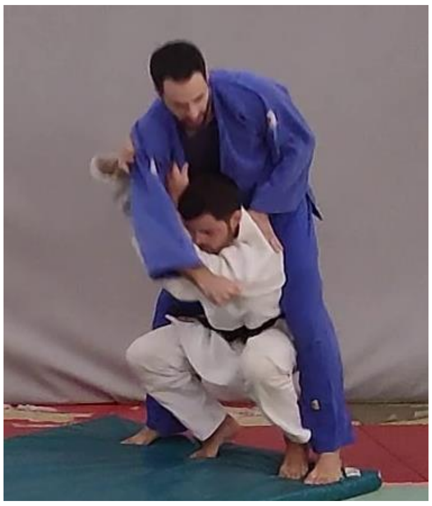
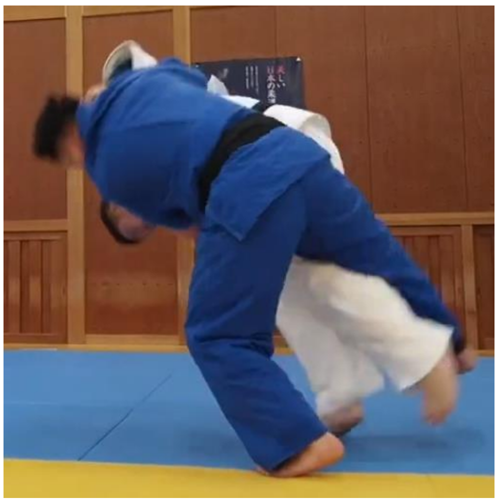



Figure 11 Examples of occlusions and uncommon positions

The dataset was constructed from 26 video recordings, 8 of which were newly recorded specifically for this study to increase the dataset's size and variability. These videos were selected based on criteria that ensured only two people were present in the frame to simplify the detection process. The videos were then segmented into shorter clips, each lasting between 1 to 3 seconds, to focus on the crucial moments of the throws. This approach resulted in a total of 219 clips.

To facilitate the training of models, the clips were annotated not only with labels indicating the type of action (*ippon-seoi-nage*, *uchi-mata*, or non-throw) but also with ground truth bounding boxes drawn around the athletes. These annotations were essential for evaluating different models during the search for the most suitable approach for detecting and classifying judo throws.

3.2 Model Evaluation for Object Detection and Pose Estimation

To discover the best model for throw classification, the newly created dataset was put to the test. The evaluation used metrics like precision, recall, and mean intersection over union (IoU) to measure each model's ability to accurately detect judokas in their varied and often unusual positions during throws. To make this assessment, specific frames from the videos were selected and with its ground truth bounding boxes manually drawn for both athletes, which allowed for a direct comparison with the bounding boxes predicted by the different models.

$$IOU = \frac{Area\ of\ overlap}{Area\ of\ union} \qquad (1)$$

One crucial aspect of the evaluation is calculating the IoU between two bounding boxes. This calculation is performed by a dedicated function that receives two parameters: the ground truth bounding boxes and the predicted bounding boxes. The function computes the area of intersection, the areas of the individual bounding boxes, and the union of the two boxes, excluding the intersection area. With the final value being calculated with equation (1).

$$Precision = \frac{True\ Positives}{True\ Positives + False\ Positives} \qquad (2)$$

$$Recall = \frac{True\ Positives}{True\ Positives + False\ Negatives} \qquad (3)$$

In addition to IoU, another function calculates the evaluation metrics—precision, recall, and mean IoU. This function takes three parameters: the ground truth bounding boxes, the predicted bounding boxes, and the IoU threshold. It initializes variables to accumulate metrics and uses loops to determine the maximum IoU for each combination of ground truth and predicted bounding boxes. Matches are established based on whether the IoU exceeds the predefined threshold, allowing the function to classify true positives, false positives, and false negatives. Ultimately, the metrics for the mean IoU are computed, precision using equation (2) and recall with equation (3).

For the task of object identification, specifically detecting people, I used the models MobileNet SSD, RetinaNet, YOLOv5, YOLOv8, and YOLO-NAS.

Models	Precision	Recall	Avg IoU
MobileNet SSD	0.9392	0.5699	0.6723
RetinaNet	0.9611	0.5547	0.7631
YOLOv5	0.9416	0.5183	0.7179
YOLOv8	0.98	0.5982	0.7698
YOLO-NAS	0.9915	0.6788	0.8091

Table 1 Athlete identification results

Table 1 allows for a comparison of the performance of different models based on the average of three distinct metrics. YOLO-NAS stands out as the best model, achieving the highest values across all metrics, with a precision of 0.9915, recall of 0.6788, and an avg IoU of 0.8091. YOLOv8 follows closely, also showing strong results, especially in precision (0.98) and avg IoU (0.7698), but with a slightly lower recall (0.5982).

Both RetinaNet and YOLOv5 performed well in precision (0.9611 and 0.9416, respectively) and avg IoU (0.7631 and 0.7179), but encountered challenges with recall, achieving 0.5547 and 0.5183. MobileNet SSD, while maintaining a respectable precision of 0.9392, demonstrated lower performance in recall (0.5699) and avg IoU (0.6723), making it less effective overall compared to the other models.

For pose estimation, four models were evaluated: MoveNet, MediaPipe, YOLOv8 Pose, and YOLO-NAS Pose. This evaluation was not conducted in a qualitative manner, as there is no straightforward way to precisely measure the performance of these models for this specific task. Instead, the analysis was visual, checking whether the judokas' skeletons was detected in the frames and assessed and how closely the predicted key points matched actual body positions.

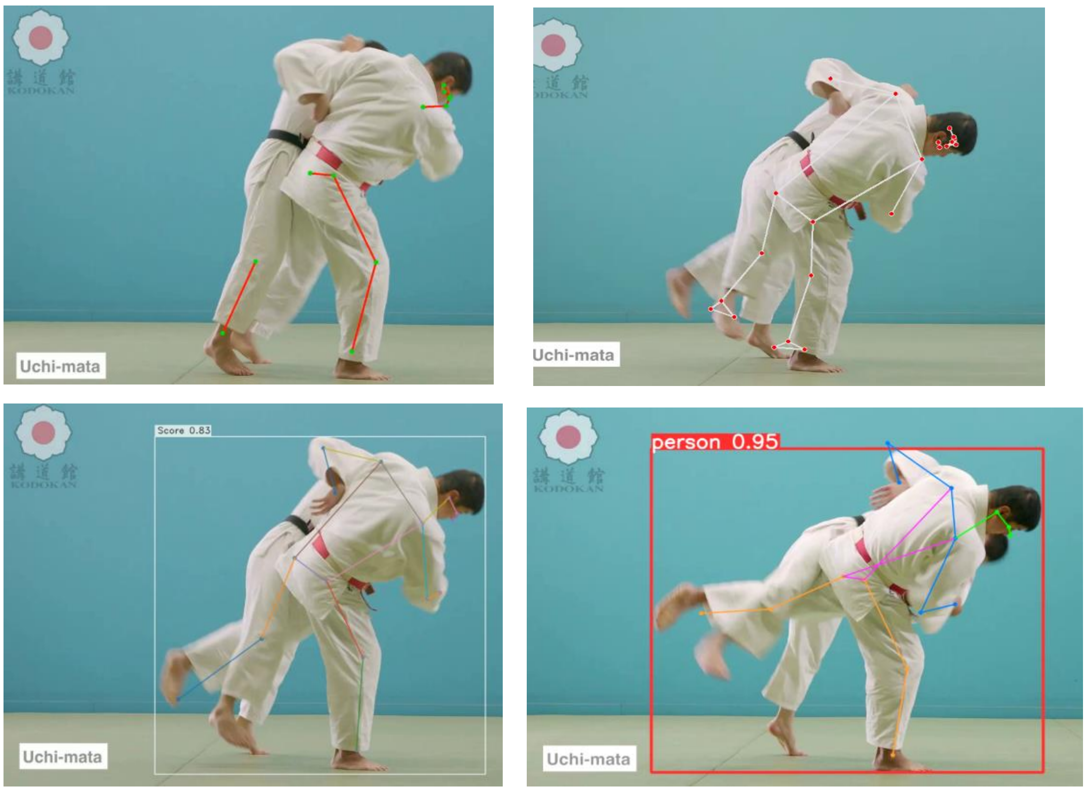


Figure 12 Some examples of pose estimation performed by the different models

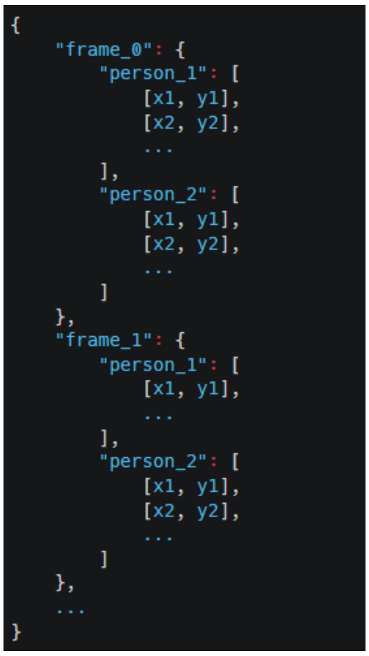
(from the top left to the right, Movenet, MediaPipe, Yolo-NAS Pose, Yolov8 Pose)

Figure 12 shows examples of pose estimation by each model, had the least accurate performance, struggling to produce a reliable skeleton. Mediapipe detected key points with reasonable accuracy but misidentified the thrower’s leg as belonging to the opponent YOLO-NAS Pose provided the most accurate estimation overall, while YOLOv8 Pose also performed well, though it misinterpreted the right leg as making the sweep instead of the left.

Both YOLOv8 Pose and YOLO-NAS Pose produced the best results, demonstrating the higher level of accuracy in detecting the judokas' skeletons. YOLOv8 was selected for the remainder of the project, as it not only delivered the better results but also has a wider range of resources and documentation available due to its more extensive use in other projects.

3.3 Data Preparation

As specified before, the data of the coordinates of the different key points per frame are saved in a JSON file, with each clip corresponding to a single file. The pose estimation model detects 17 key points for each person, representing various body joints. These key points are recorded as x, y coordinates for every frame in the sequence.



```
{
    "frame_0": {
        "person_1": [
            [x1, y1],
            [x2, y2],
            ...
        ],
        "person_2": [
            [x1, y1],
            [x2, y2],
            ...
        ]
    },
    "frame_1": {
        "person_1": [
            [x1, y1],
            ...
        ],
        "person_2": [
            [x1, y1],
            [x2, y2],
            ...
        ]
    },
    ...
}
```

Figure 13 JSON schema for saving key points coordinates of two judokas

Based on these requirements, the schema used for the JSON file is represented in Figure 13, where each entry corresponds to a frame in the video, containing the coordinates for all 17 key points of both 'person_1' and 'person_2'.

Upon examination of the generated files, it becomes evident that when YOLO fails to detect a key point, the corresponding coordinates are left blank. To prevent any issues during subsequent processing, these blank entries are systematically filled with [0.0, 0.0], thereby standardizing the data format. Various strategies for handling missing key points are considered, such as interpolating key point values between the previous and subsequent frames. However, after implementing and testing these approaches, it is concluded that they do not improve the overall results.

To ensure the accuracy of the key point data saved in the JSON files, a validation script is developed for future debugging. This script reads the "key points" data and overlays color-coded skeletons for each individual onto the original video, enabling a visual comparison between the predicted poses and the actual movements captured in the clip.

Through this visualization, it becomes evident that the tracking system occasionally swaps the identities of the two individuals between frames which can be observed in figure 14. This error occurs because the pose estimation algorithm consistently labels the detection with higher confidence as 'Person 1'.

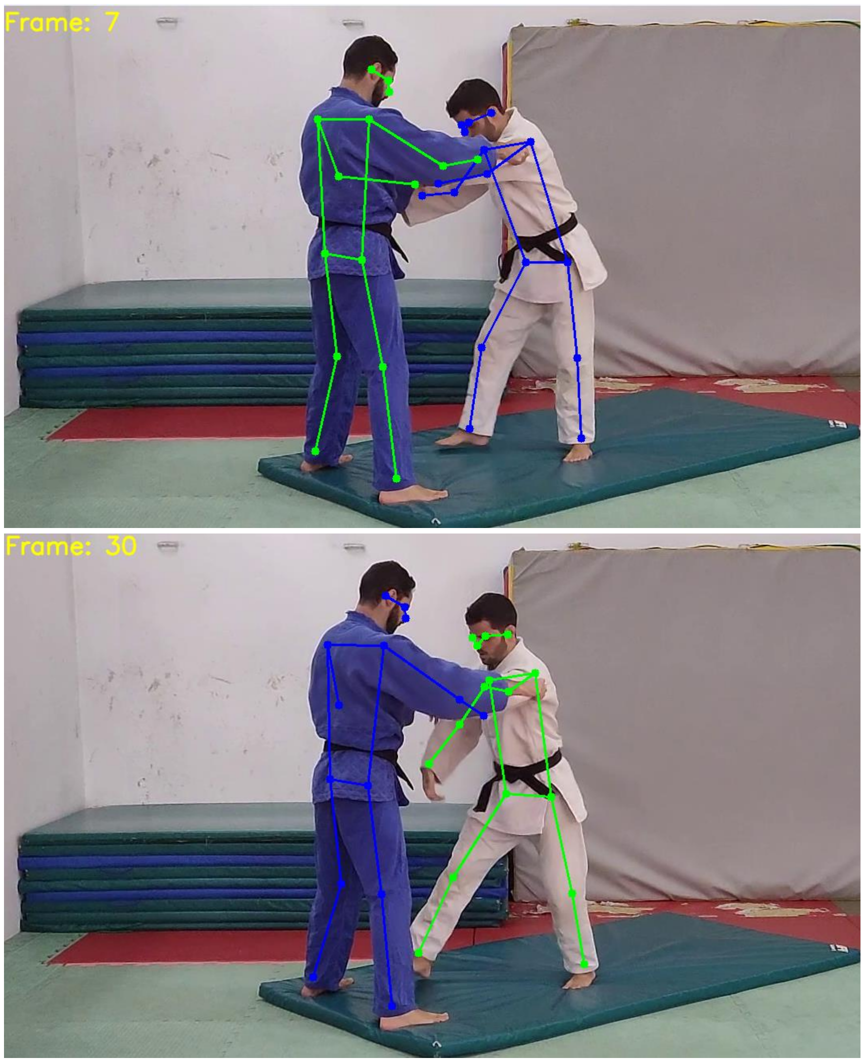


Figure 14 Example of a mismatch of IDs

To address the issue of occasional ID (identification) swaps between tracked individuals, a correction algorithm was implemented. This algorithm operates on the principle that the movement of an individual between consecutive frames should be relatively small compared to the distance between two different individuals.

The algorithm compares the key points of each person in the current frame with those of both people in the previous frame. It calculates the total Euclidean distance between corresponding key points, ignoring any undetected or invalid key points. An ID swap is detected if the following condition is met:

- The distance between Person 1 in the previous frame and Person 2 in the current frame is smaller than the distance between Person 1 in both frames.
- The distance between Person 2 in the previous frame and Person 1 in the current frame is smaller than the distance between Person 2 in both frames.

When this condition is true, it suggests that the identities of the two individuals have been swapped. The algorithm then corrects this by exchanging the key point data between Person 1 and Person 2 in the current frame.

This method is applied sequentially to all frames in the video, comparing each frame with its predecessor. The corrected data is then saved, ensuring consistent tracking of individuals throughout the video sequence. The same frames of Figure 14 can be seen in Figure 15 after being reprocessed.

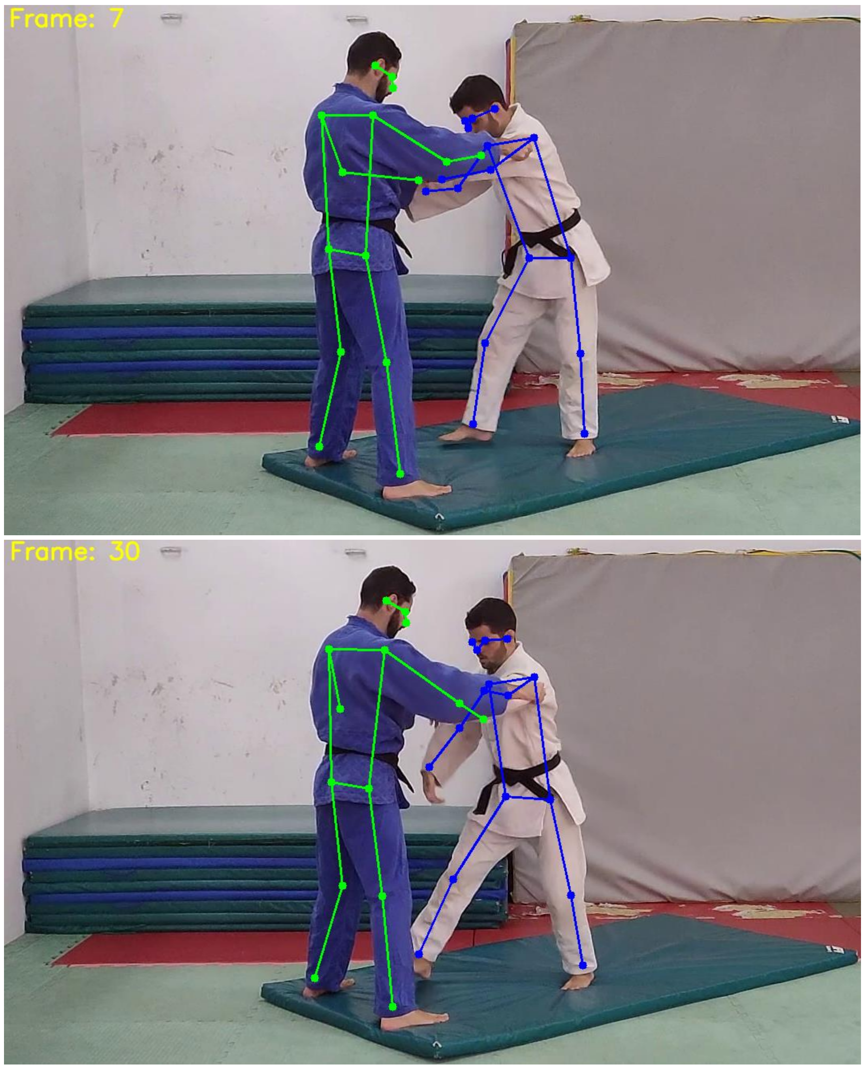


Figure 15 Example of the ID Mismatch Correction

3.4 Model Training

3.4.1. Direct Classification System

This section outlines the initial version of the model training process for classifying judo throws based on pose estimation data. To effectively evaluate the model's performance, the dataset was split into training and validation sets using an 80/20 ratio. This division ensures that the model is trained on a substantial portion of the data while retaining enough samples for validation to assess its generalization capabilities.

Since not all clips are the same length, uniform sampling is implemented to extract 30 frames from each clip. This approach standardizes the input data, allowing the model to process sequences of uniform length during training.

For this first version, a Long Short-Term Memory (LSTM) network is employed due to its effectiveness in handling sequential data, such as the time-varying coordinates of key points. The model is implemented using PyTorch, a popular deep learning framework.

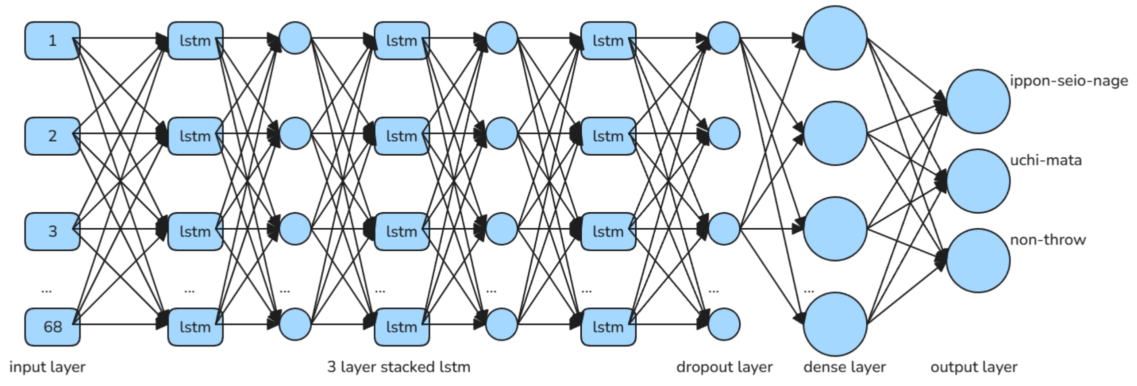


Figure 16 Diagram for the first version of Throw Classifier

The architecture of this LSTM model is illustrated in Figure 16. In this implementation, the input dimension corresponds to the number of key points being tracked, totaling 68 (17 key points × 2D coordinates × 2 people). The hidden dimension is set to 64, and the output dimension represents the three possible classes of judo throws (e.g., *ippon-seoi-nage*, *uchi-mata*, or non-throw). The LSTM comprises three layers, facilitating the extraction of complex temporal features from the input data.

The training process utilizes CrossEntropyLoss class as a criterion and an Adam optimizer with a learning rate of 0.001. The training loop runs for a predefined number of 50 epochs, employing early stopping to prevent overfitting. Early stopping monitors validation loss and halts training if no improvement is observed over five epochs.

After training the model, the performance is evaluated using accuracy and loss metrics for both the training and validation datasets. The training results depicted in Figure 17, over 50 epochs, reveal a model that demonstrates learning but also exhibits signs of overfitting. The training accuracy shows steady improvement, reaching approximately 80% by the final epoch, while the validation accuracy, though also increasing, maxes out around 65% with noticeable fluctuations.

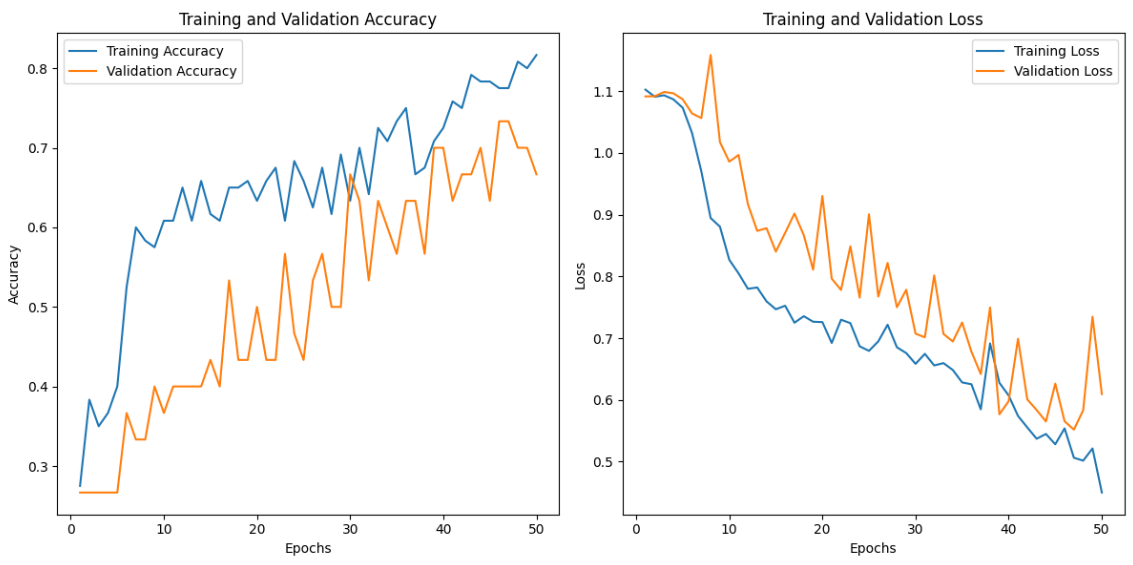


Figure 17 Graph of the evolution of metrics throughout epochs

The corresponding loss curves reveal a similar pattern: both training and validation losses decrease as the epochs progress. The training loss declines more smoothly and ends at a lower value, indicating effective fitting to the training data. However, the validation loss remains higher at above 0.6, which points to some overfitting, as the model struggles to generalize to the validation set. An early spike in validation loss around epochs 5-10 further suggests that the model initially faced challenges with generalization, though it stabilized in later epochs.

	Accuracy	Loss
Train	0.8167	0.4500
Validation	0.6667	0.6095

Table 2 Final performance metrics for train and validation

The model's final performance metrics, presented in Table 2, indicate a training accuracy of 0.8167 suggesting that the model can effectively learn and classify judo throws from the training data. However, the validation accuracy of 0.6667 suggests that there is still room for improvement in terms of generalizing new, unseen data.

The loss metrics provide further insight into the training dynamics: a training loss of 0.4500 shows that the model fit with the training data is acceptable, whereas a higher validation loss of 0.6095 implies that the model encounters more difficulties when making predictions on the validation set, suggesting some level of overfitting.

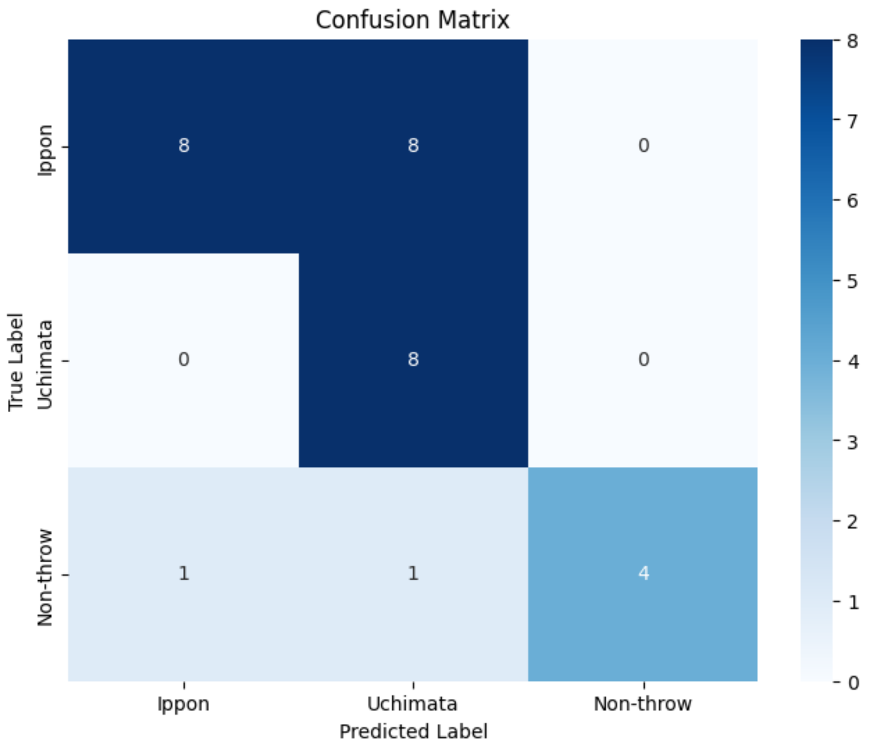


Figure 18 Confusion Matrix for the first version of Throw Classifier

The confusion matrix compares the true labels with the predicted labels for each class, providing insights into the model's classification performance. Correct classifications are located along the diagonal, while values outside the diagonal indicate misclassifications.

Figure 18 shows that the model struggles to differentiate between the two types of judo throws, with 8 instances of "*uchi-mata*" being misclassified as "*ippon-seoi-nage*." This indicates a significant overlap in the learned features of these two classes. Additionally, there are 2 cases where "non-throw" actions are misclassified as throws. However, this issue is less pronounced compared to the confusion between the two types of throws.

Overall, the confusion matrix highlights that the primary classification challenge lies in distinguishing between the similar throw categories, rather than between throw and non-throw actions.

3.4.2 Two-Stage Classification System

Given the moderate accuracy, relatively high loss, and noticeable confusion in differentiating the two types of throws, a strategic change is considered to improve classification performance. Instead of using a single model to directly classify all three categories, a two-step approach is proposed:

1. **Is Throw Classification (Throw vs. Non-Throw)**: First, a model will classify the data as either "Throw" or "Non-Throw."
2. **Throw Type Classification (Ippon vs. Uchimata)**: If the first model predicts "Throw," a second model will classify the type of throw (*ippon-seoi-nage* or *uchi-mata*).

By breaking the classification task into two simpler stages, the models are expected to better capture the features distinguishing throws from non-throws and then focus more specifically on differentiating the two types of throws. This approach aims to reduce confusion in the final predictions and improve overall accuracy.

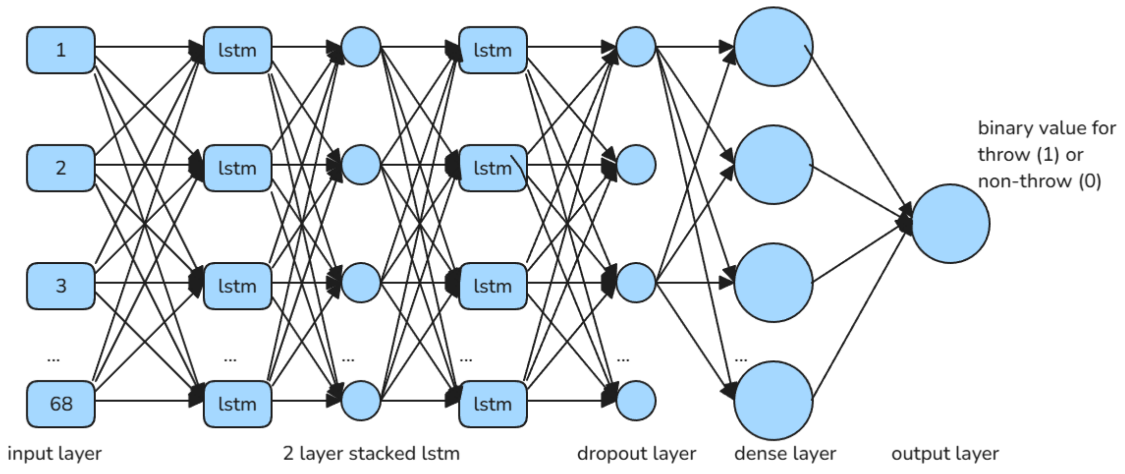


Figure 19 Diagram of isThrowClassifier model

For the first stage, an LSTM network is still utilized, but it has been adapted for binary classification, as presented in Figure 19. With the only changes being a single output neuron, either returning 1 for throws or 0 for non-throws. Reducing the number of layers from 3 to 2. And using binary cross entropy with the help of the BCEWithLogitsLoss (binary cross-entropy), instead of cross entropy as a criterion, since it is better suited for binary classification.

The adapted architecture features the following changes, the output layer now has a single neuron, outputting a value that indicates the probability of the input being a "Throw" (1) or "Non-Throw" (0). The number of LSTM layers has been reduced from three to two, aiming to simplify the model and prevent overfitting, given the less complex binary classification problem. The loss function has been switched to BCEWithLogitsLoss which is designed for binary classification tasks, making it more appropriate than the previously used cross-entropy loss.

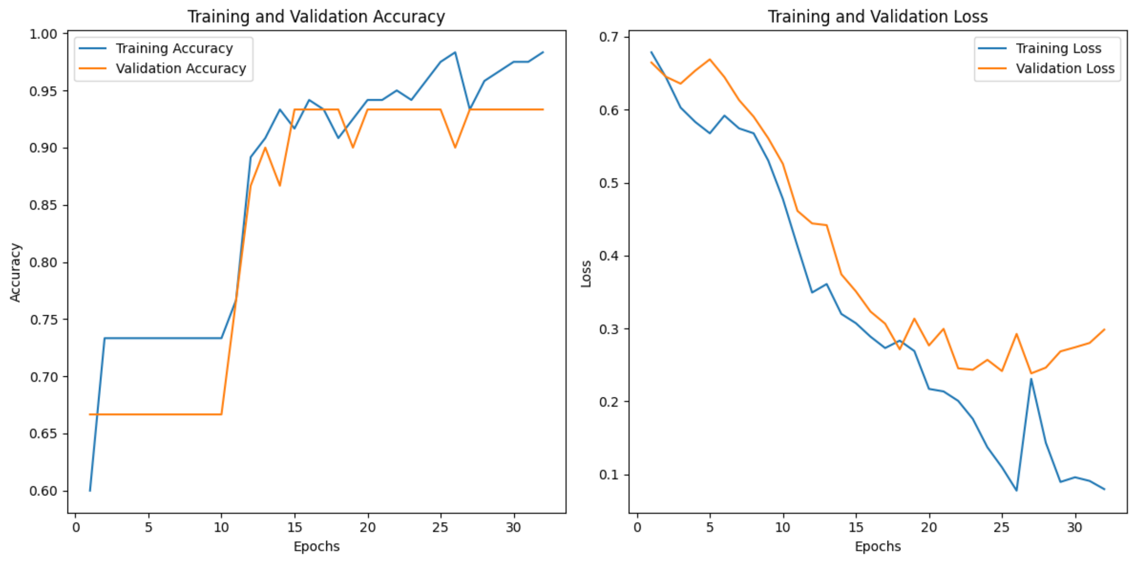


Figure 20 Graph of the evolution of metrics for isThrowClassifier model

Figure 20 illustrates the model training for a bit more than 30 epochs and shows a clear trend of increasing accuracy and decreasing loss. In terms of accuracy, there is a significant jump around the 10th epoch, where the training accuracy reaches approximately 90%. This improvement suggests that the model began to effectively learn the distinguishing features of the input data at this point.

After this initial increase, the training accuracy plateaus, indicating that the model has reached a relatively stable state of performance, hovering around 95% by the end of the training period. Similarly, the validation accuracy also stabilizes around 93%, reflecting that the model is generalizing reasonably well to unseen data.

The loss graphs further illustrate this trend, with the training loss consistently decreasing over the epochs, ultimately approaching less than 0.1. This low training loss indicates that the model is fitting well to the training data. The validation loss shows a similar behavior until epoch 18, where it stabilizes around 0.3.

	Accuracy	Loss
Train	0.9833	0.0796
Validation	0.9333	0.2983

Table 3 Final performance metrics for train and validation for isThrowClassifier model

The final performance metrics indicate that the training accuracy is 98.33%, showing effective classification of the training data. The training loss of 0.0796 reflects good fitting to the training set. The validation accuracy is 93.33%, while the validation loss is 0.2983, indicating the model generalizes reasonably well to unseen data, though some room for improvement remains.

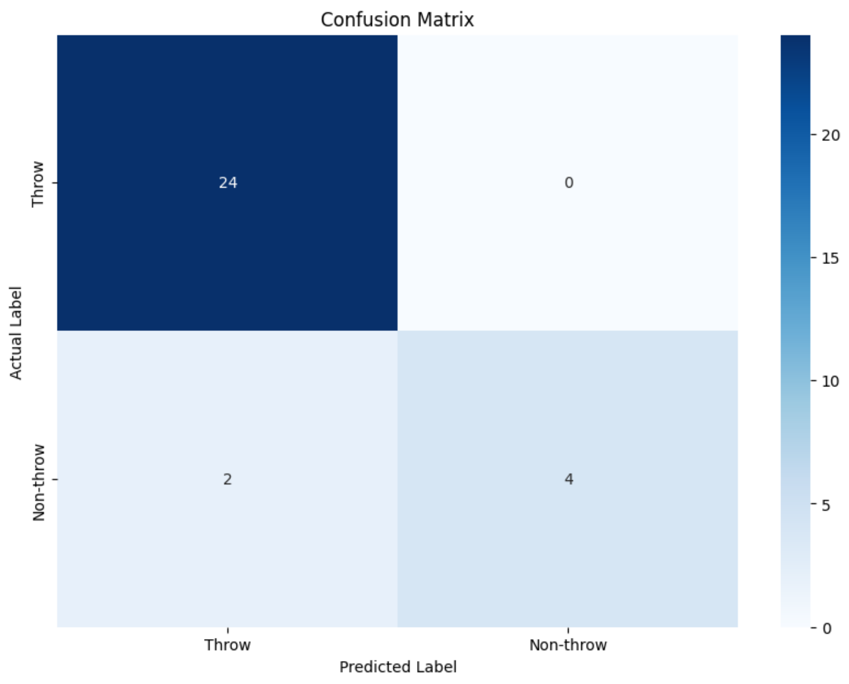


Figure 21 Confusion Matrix for isThrowClassifier model

Based on Figure 21, the model classifies 24 instances of throws correctly and misclassifies 2 instances of non-throws as throws. There are no misclassifications of throws as non-throws, indicating a strong ability to identify throw actions accurately.

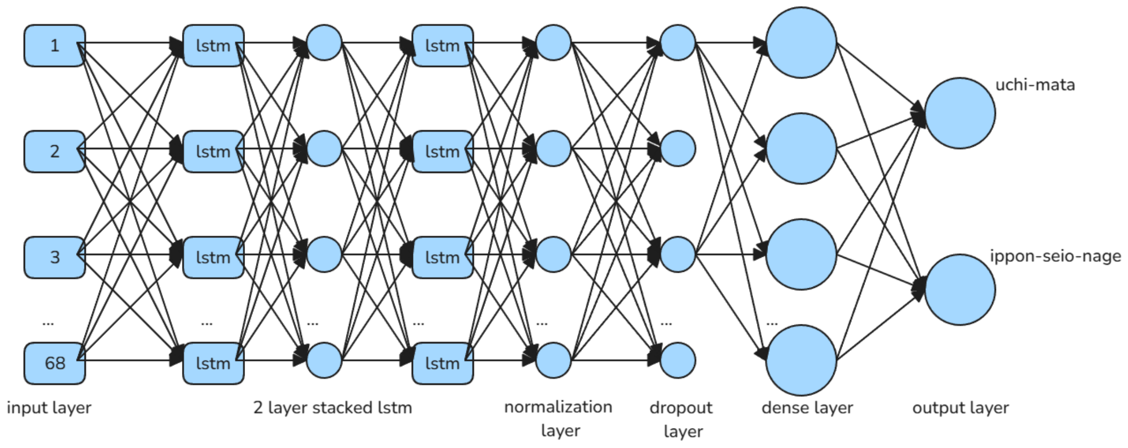


Figure 22 Diagram of throwTypeClassifier model

For the second model represented in Figure 22, a batch normalization layer has been added to improve training stability and speed by normalizing the output from the LSTM. The number of LSTM layers has been reduced to 2, which simplifies the model structure while maintaining the capacity to learn complex temporal features.

The output layer now consists of 2 neurons, one for each throw type (*ippon-seoi-nage* and *uchi-mata*). This change enables the model to output separate probabilities for each throw class, enhancing the model's ability to distinguish between these two types of throws.

Additionally, k-fold cross-validation is implemented to evaluate the model's performance more robustly by splitting the dataset into multiple training and validation sets, allowing for better generalization assessment. In this case, k=5, meaning the dataset is divided into five subsets. Each subset is used once as a validation set while the remaining four are used for training, leading to a comprehensive evaluation of the model's performance.

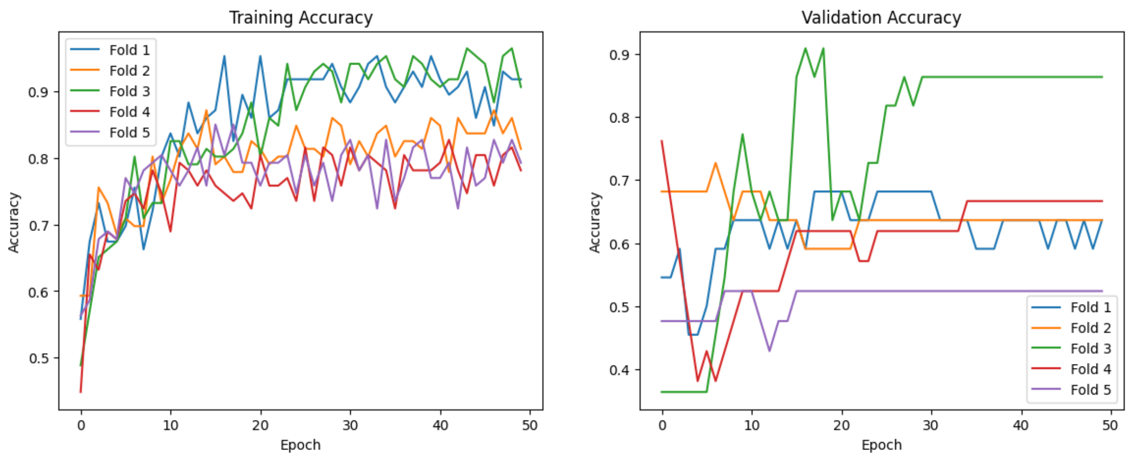


Figure 23 Graph of the evolution of accuracy for the 5 folds of throwTypeClassifier model

The training accuracy across the five folds shows a general upward trend throughout the epochs as shown in Figure 23. However, they show significant variability in validation accuracy, suggesting that the model's performance is influenced by the specific data split used for training and validation. Notably, fold 3 exhibits the best results, from epoch 10 to 25 it as a weird fluctuation where it goes from 65% to close to 90% and comes down again to 65% and then reaching around 85% accuracy around epoch 30 and maintaining that level throughout the remainder of the training process. This indicates that for this particular split, the model was able to effectively learn and generalize the features needed for classification.

Other folds show less pronounced improvements and a lower degree of stability. For example, Folds 1 and 2 start with rapid changes in accuracy but eventually stabilize around 60-65% by the end of training. Fold 4 displays moderate fluctuations, reaching about 68% accuracy, while Fold 5 struggles the most, finishing below the 60% mark. These discrepancies among the folds indicate variability in model generalization across different data splits, with some splits presenting more challenging or less representative data samples for the model to learn from effectively.

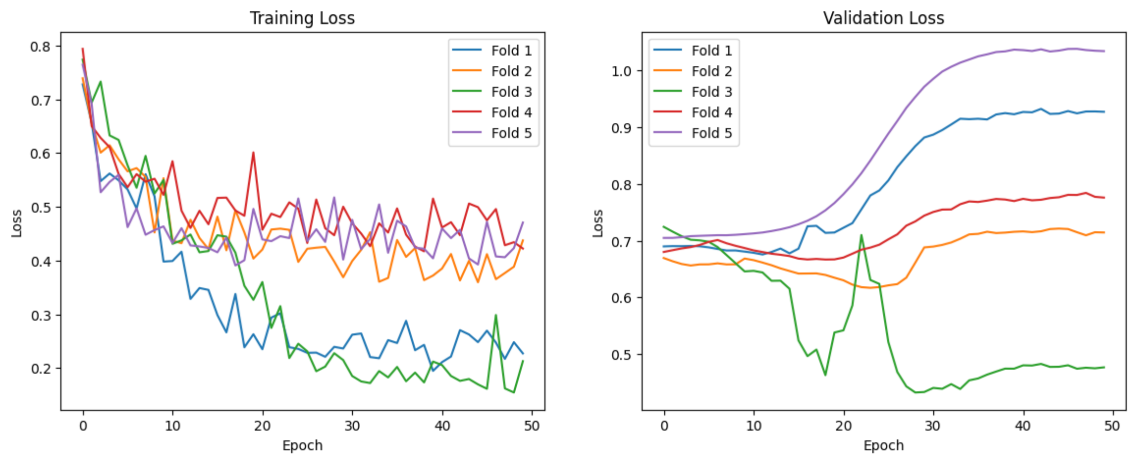


Figure 24 Graph of the evolution of loss for the 5 folds of throwTypeClassifier model

The training and validation loss graphs of Figure 24 reveal significant variability across the five folds, indicating differences in the model's learning and generalization capabilities depending on the data splits. In the training loss graph, all folds show a general downward trend, reflecting the model's ability to fit the training data over time. Fold 3 exhibits the lowest training loss, reaching around 0.2 by epoch 50, suggesting effective learning during training. Folds 1 also shows considerable reductions in training loss, stabilizing around 0.25, while folds 2, 4 and 5 demonstrate higher losses, indicating potentially less effective training or higher variability in the data.

The validation loss graph adds further context, revealing how well the model generalizes to unseen data. Fold 3 shows the lowest validation loss trajectory, dropping significantly around epoch 18, then fluctuating before stabilizing at approximately 0.45 by epoch 50, which aligns behavior in the accuracy graph. The rest of the folds their loss values only increased during training ending with values well above 0.6, which indicates that the model struggled to generalize effectively for these data splits, resulting in significant prediction errors on the validation set.

Ultimately, Fold 3 presents the best results in terms of both accuracy and loss, highlighting its effectiveness in distinguishing between the two types of judo throws.

	Accuracy	Loss
Train	0.9070	0.2127
Validation	0.8636	0.4766

Table 4 Final performance metrics for train and validation for throwTypeClassifier model

The results presented in Table 4 indicate that the model achieved a training accuracy of 90.70% with a training loss of 0.2127, and a validation accuracy of 86.36% with a validation loss of 0.4766. While these results are not as strong as those achieved by the isThrowLSTMClassifier, the lower performance of the throwTypeLSTMClassifier is understandable due to the increased complexity of distinguishing between different throw types rather than simply identifying whether a throw occurred.

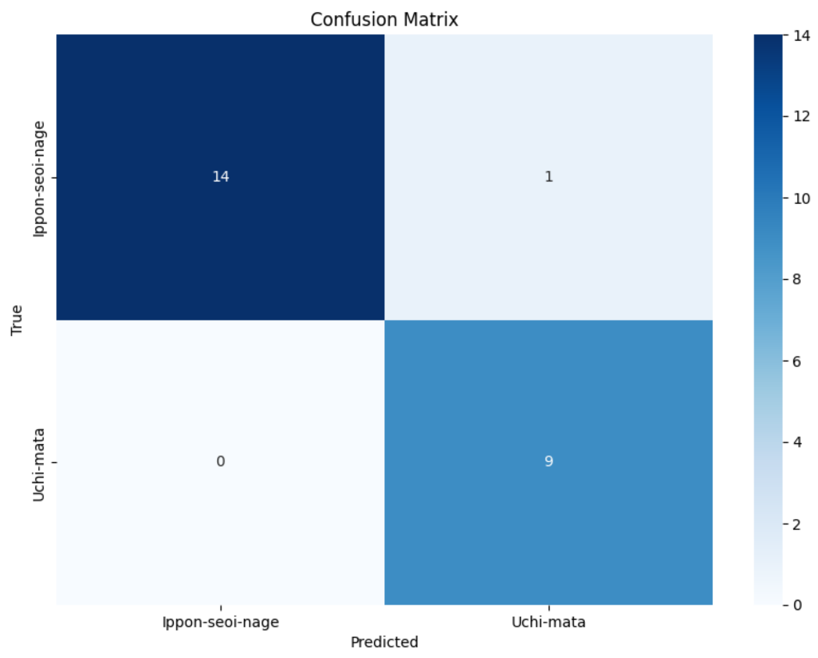


Figure 25 Confusion Matrix for throwTypeClassifier model

Based on Figure 25, the model classifies almost all instances of the throws correctly, only wrongly classifying 1 *ippon-seio-nage* as *uchi-mata*.

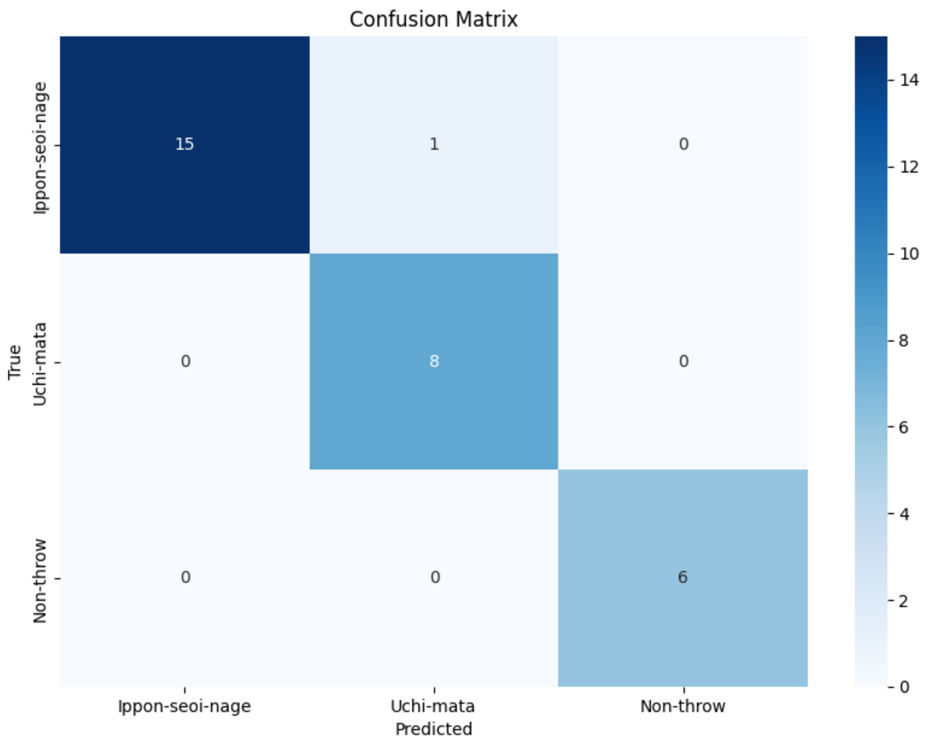


Figure 26 Confusion matrix of the isThrowClassifier model combined with throwTypeClassifier model

The confusion matrix of Figure 26 shows the validation data processed by the updated model using two separate LSTM networks. Compared to the initial version in Figure 18, which uses the same validation data, the updated approach demonstrates a noticeable improvement. The distinction between the two throw types is more accurate, reducing incorrect classifications from 8 to 1, while non-throws are classified with higher precision, improving from 2 errors to none. These results indicate that the revised model with two distinct LSTMs achieves better classification performance overall.

3.4 Evaluation

The evaluation phase assesses the model's generalization ability using a separate test dataset that was not part of the training or validation processes. This step is crucial to evaluate the model's performance and validate its robustness on completely unseen data.

The model's performance on the test dataset is presented in terms precision, recall and f1-score in Table 5, global accuracy in Table 6 and a confusion matrix in Figure 27.

	Precision %	Recall %	F1-Score
Ippon-seoi-nage	66.67	76.19	71.11
Uchi-mata	86.67	78.79	82.54
Non-Throws	83.33	83.33	83.33

Table 5 Final performance metrics for train and validation for the two-stage classification system

Global accuracy	79.17%

Table 6 Global accuracy for the two-stage classification system

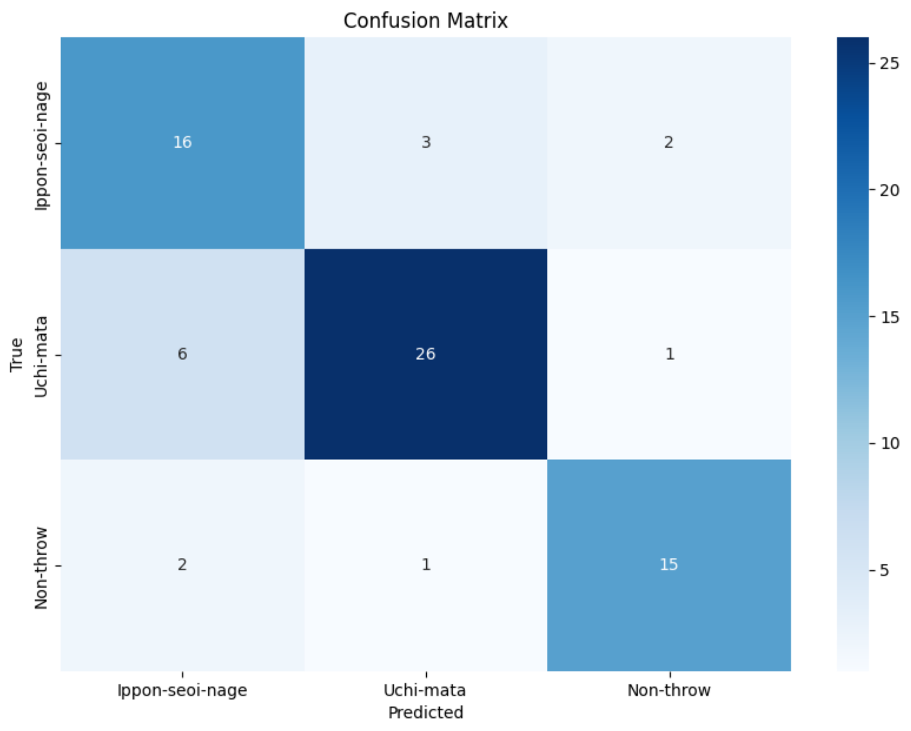


Figure 27 Confusion matrix for the two-stage classification system

The model achieved an overall accuracy of 79.17% on the test dataset, indicating good generalization to unseen data. This suggests that the model has learned meaningful features from the training data that are applicable to new instances. The performance varies across the different classes, with "*uchi-mata*" showing the highest precision at 86.67%, followed by "non-throws" at 83.33%, and "*ippon-seoi-nage*" at 66.67%.

Examining the confusion matrix provides deeper insights into the model's behavior. For "*ippon-seoi-nage*", out of 21 actual instances, 16 were correctly classified, with 3 misclassified as "*uchi-mata"* and 2 as "non-throws". "*Uchi-mata"* saw 26 correct classifications out of 33 instances, with 6 misclassified as "*ippon-seoi-nage*" and 1 as "non-throws". Non-Throws had 15 correct classifications out of 18 instances, with 2 misclassified as "*ippon-seoi-nage*" and 1 as "*uchi-mata"*.

The most common misclassification occurs between "*ippon-seoi-nage*" and "*uchi-mata"*, which is expected based on the results achieved during the training step and the increased complexity of the task. Additionally, there is a slight tendency for the model to misclassify some throw instances as "non-throws" (three cases in total), and a similar number of "non-throws" instances as throws.

Overall, the model demonstrates a balanced performance across the three categories. With the results highlighting that the use of separate LSTM networks contributed to better classification accuracy compared to earlier versions of the model.

4. Conclusion and future work

This research set out to investigate the classification of judo techniques using computer vision approaches, specifically addressing two key research questions: the identification of optimal pose detection models and the feasibility of accurate throw classification from video data.

Regarding the first research question about optimal pose detection models, "*What are the best models to detect the athletes' pose?*", the evaluation revealed that YOLOv8 and YOLO-NAS demonstrated the most promising results for both object detection and pose estimation tasks. These models proved to be the most effective in handling judo movements, though their performance was occasionally compromised during complex throwing sequences and occlusions. The selection of these models represents a significant step forward in addressing the unique challenges posed by judo's rapid movements and close-contact nature.

In addressing the second research question concerning throw classification accuracy, *"How accurately can you classify judo throws from video data?"*, the dual LSTM network approach achieved an encouraging accuracy of 79.17% on the test set. This result demonstrates the feasibility of automated judo throw classification while acknowledging room for improvement. The system successfully distinguished between different types of throws and non-throw actions, providing a solid foundation for future development in automated judo technique analysis.

Despite these achievements, several limitations emerged during the research. The pose estimation components, while advanced, still struggled with consistent prediction during critical moments of throws or when occlusions occurred, resulting in missing or incorrect key points that affected classification accuracy. The system's performance also showed sensitivity to environmental variations, particularly when multiple people appeared in the frame. Additionally, while sufficient for initial development, the current dataset's scope potentially limits the model's generalization capabilities across diverse scenarios and techniques.

Looking ahead, future work should focus on several key areas to build upon this foundation. A crucial next step would be the expansion and diversification of the dataset, incorporating more samples for existing throw types and additional techniques from diverse sources. This expansion would help capture a wider range of throwing styles and environmental conditions, enhancing the model's ability to generalize across different scenarios. The development of more sophisticated architectural approaches could also prove beneficial, particularly in handling the increased complexity of additional throw categories.

The rapid advancement of pose estimation technology offers promise for addressing current limitations naturally, potentially leading to more reliable classification results. Future research should focus on leveraging these improvements while addressing the specific challenges of judo technique analysis. This research establishes a strong foundation for automated judo technique analysis while highlighting the potential for future advancements in this field. As pose estimation technology continues to evolve and datasets grow more comprehensive, the accuracy and reliability of such systems will likely improve, bringing us closer to practical applications that can benefit the judo community as a whole.

Bibliographic references

[1] B. T. Naik, M. F. Hashmi, and N. D. Bokde, ‘A Comprehensive Review of Computer Vision in Sports: Open Issues, Future Trends and Research Directions’, 2022.

[2] Y. Pang, C. Zhang, Y. Wang, Q. Wang, and M. Wang, ‘Analysis of Computer Vision Applied in Martial Arts’, presented at the 2022 2nd International Conference on Consumer Electronics and Computer Engineering, ICCECE 2022, 2022, pp. 191–196. doi: 10.1109/ICCECE54139.2022.9712803.

[3] J. Stenum, K. M. Cherry-Allen, C. O. Pyles, R. D. Reetzke, M. F. Vignos, and R. T. Roemmich, ‘Applications of Pose Estimation in Human Health and Performance across the Lifespan’, *Sensors*, vol. 21, no. 21, Art. no. 21, Jan. 2021, doi: 10.3390/s21217315.

[4] J. vom Brocke, A. Hevner, and A. Maedche, ‘Introduction to Design Science Research’, 2020, pp. 1–13. doi: 10.1007/978-3-030-46781-4_1.

[5] R. Szeliski, ‘Computer Vision: Algorithms and Applications, 2nd Edition’, 2021.

[6] Simon J.D. Prince, *Understanding Deep Learning*. 2023.

[7] S. S. A. Zaidi, M. S. Ansari, A. Aslam, N. Kanwal, M. Asghar, and B. Lee, ‘A survey of modern deep learning based object detection models’, *Digital Signal Processing*, vol. 126, p. 103514, Jun. 2022, doi: 10.1016/j.dsp.2022.103514.

[8] Y.-C. Zhou, Z.-Z. Hu, K.-X. Yan, and J.-R. Lin, ‘Deep Learning-Based Instance Segmentation for Indoor Fire Load Recognition’, *IEEE Access*, vol. 9, pp. 148771–148782, Nov. 2021, doi: 10.1109/ACCESS.2021.3124831.

[9] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko, ‘End-to-End Object Detection with Transformers’, May 28, 2020, *arXiv*: arXiv:2005.12872. Accessed: Feb. 16, 2024. [Online]. Available: http://arxiv.org/abs/2005.12872

[10] S. Kato and S. Yamagiwa, ‘Predicting Successful Throwing Technique in Judo from Factors of Kumite Posture Based on a Machine-Learning Approach’, *Computation*, vol. 10, p. 175, Sep. 2022, doi: 10.3390/computation10100175.

[11] Jigoro Kano, *Kodokan Judo: The Essential Guide to Judo by Its Founder Jigoro Kano*. 2013.

[12] N. Nonaka *et al.*, ‘End-to-End High-Risk Tackle Detection System for Rugby’, in *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, Jun. 2022, pp. 3549–3558. doi: 10.1109/CVPRW56347.2022.00399.

[13] I. F. Nurahmadan, Jayanta, and I. W. W. Pradnyana, ‘Utilization of Pose Estimation and Multilayer Perceptron Methods in the Development of Taekwondo Martial Arts Independent Learning’, presented at the Proceedings - 3rd International Conference on Informatics, Multimedia, Cyber, and Information System, ICIMCIS 2021, 2021, pp. 267–272. doi: 10.1109/ICIMCIS53775.2021.9699268.

[14] E. Quinn and N. Corcoran, ‘The Automation of Computer Vision Applications for Real-Time Combat Sports Video Analysis’, presented at the 4th European Conference on the Impact of Artificial Intelligence and Robotics, ECIAIR 2022, 2022, pp. 162–171.

[15] V. Hudovernik and D. Skocaj, ‘Video-Based Detection of Combat Positions and Automatic Scoring in Jiu-jitsu’, presented at the MMSports 2022 - Proceedings of the 5th International ACM Workshop on Multimedia Content Analysis in Sports, 2022, pp. 55–63. doi: 10.1145/3552437.3555707.